HPSS

System Administration Guide

High Performance Storage System
Release 4.1.1

February 2000 (Revision 3)

HPSS System Administration Guide

Printed in the United States of America.

HPSS Release 4.1.1
February 2000 (Revision 3)

Table of Contents

HPSS System Administration Guide

List of Figures

List of Tables

Preface

HPSS System Administration Guide

This *High Performance Storage System (HPSS) System Administration Guide* provides the necessary information for successful installation, configuration, and day-to-day operation of HPSS. It is designed for both HPSS administration and operations personnel.

Administrators are normally responsible for initial installation and system configuration, as well as setting and managing site-specific HPSS policies. Operators are responsible for starting, monitoring, controlling, and stopping HPSS and its components during normal daily operation. Depending on the site, administrative and operational tasks may be performed by the same or different personnel.

General information in the *Guide* is presented in the early chapters. Progressively more detailed information concerning HPSS installation, configuration, maintenance, intervention, and problem diagnosis are in the later chapters.

The version of this guide is Release 4.1.1. It presents information on HPSS Release 4.1.1 only. Information on future releases of HPSS is not included. Refer to the Appendix M in this document for more information on the new capabilities and upgrade procedures for Release 4.1.1.

The *HPSS System Administration Guide* is one in a series of documents intended to provide a complete set of information about HPSS. Other documents currently include the *HPSS Error Messages Reference Manual*, the *HPSS Programmer's Reference Guide (*Volume 1 and 2), and the *HPSS User's Guide*. In addition "Help Files" are available as part of the various system administrator and operator windows that are used to setup and control the system.

For information on interpreting and responding to HPSS error messages during system operation, refer to the *HPSS Error Messages Reference Manual*. For information on client use of HPSS or information relevant to HPSS applications or systems programmers, refer to the *HPSS User's Guide* and the *HPSS Programmer's Reference Guide*.

The *HPSS System Administration Guide* is structured as follows:

Chapter 1: HPSS Basics — This chapter provides an introduction to the HPSS system. It provides a high-level summary of the HPSS components (objects, servers, infrastructure, interfaces, policies) and describes HPSS hardware platforms.

Chapter 2: HPSS Planning — This chapter describes the HPSS planning guidelines and considerations to help the administrator in effectively planning for an HPSS

	system. The chapter will guide the administrator in making the key decisions to be used during the installation, configuration, and operational phases of HPSS.
Chapter 3: HPSS Installation	This chapter provides instructions and supporting information for installing the HPSS software from the HPSS distribution media.
Chapter 4: HPSS Infrastructure Configuration	This chapter provides instructions and supporting information for the infrastructure configuration of HPSS.
Chapter 5: HPSS Configuration	This chapter provides instructions for creating the configuration data to be used by the HPSS servers. This includes creating the server configurations, defining the storage policies, defining the storage characteristics, creating the HPSS storage space, and defining additional UNIX configurations.
Chapter 6: HPSS Management	This chapter provides instructions and supporting information for most day-to-day or regularly-performed administrative and operational procedures for HPSS.
Chapter 7: DFS Configuration and Management	This chapter provides instructions for configuration and management of DFS/HPSS and other related components to support DFS interface.
Chapter 8: HPSS Special Intervention Procedures	This chapter provides instructions and supporting information for special operations that are not considered usual or regularly-performed procedures for HPSS.
Chapter 9: HPSS Problem Diagnosis and Resolution	This chapter provides advice for solving selected problems with HPSS infrastructure components, servers, and user interfaces. Information presented in this chapter can be used as a supplement to the *HPSS Error Messages Reference Manual*, which contains descriptions of specific logged error messages and suggestions for administrative action.
Appendix A: Glossary of Terms and Acronyms	This appendix explains terms and acronyms used throughout the text.
Appendix B: References	This appendix lists documents cited in the text as well as other reference materials.
Appendix C: HPSS Worksheets	This appendix provides worksheets to allow the user to record the essential planning decisions for HPSS.

Appendix D:	*Accounting Examples*	This appendix describes how to set up the gathering of accounting data at a customer site.
Appendix E:	*Installation Examples*	This appendix gives examples for the HPSS installation.
Appendix F:	*Infrastructure Configuration Examples*	This appendix gives examples for the HPSS infrastructure configuration.
Appendix G:	*DCE Cross Cell Administration*	This appendix provides information necessary for performing the DCE Cross Cell administration.
Appendix H:	*Additional SSM Information*	This appendix provides additional SSM information that may be of interest to an HPSS administrator.
Appendix I:	*HPSS Utility Manual Pages*	This appendix describes usage information for the HPSS utilities.
Appendix J:	*IBM 3494/3495 PVR Information*	This appendix provides information necessary for configuring and using the IBM 3494/3495 robots.
Appendix K:	*StorageTek PVR Information*	This appendix provides information necessary for configuring and using the StorageTek Silo.
Appendix L:	*ADIC Automatic Media Library Storage Systems Information*	This appendix provides information necessary for configuring and using the ADIC AML library.
Appendix M:	*HPSS Release 4.1.1*	This appendix provides information necessary for upgrading to HPSS Release 4.1.1.
Appendix N:	*Developer Acknowledgments*	

Chapter 1

HPSS Basics

1.1 Introduction

The High Performance Storage System (HPSS) is software that provides hierarchical storage management and services for very large storage environments. HPSS may be of interest in situations having present and future scalability requirements that are very demanding in terms of total storage capacity, file sizes, data rates, number of objects stored, and numbers of users. HPSS is part of an open, distributed environment based on OSF Distributed Computing Environment (DCE) products that form the infrastructure of HPSS. HPSS is the result of a collaborative effort by leading US Government supercomputer laboratories and industry to address very real, very urgent high-end storage requirements. HPSS is offered commercially by IBM Worldwide Government Industry, Houston, Texas. In terms of HPSS licensing, an instance of HPSS means a single operational HPSS system with a single HPSS Name Server.

HPSS provides scalable parallel storage systems for highly parallel computers as well as traditional supercomputers and workstation clusters. Concentrating on meeting the high end of storage system and data management requirements, HPSS is scalable and designed to store up to petabytes (10^{15}) of data and to use network-connected storage devices to transfer data at rates up to multiple gigabytes (10^{9}) per second.

HPSS provides a large degree of control for the customer site to manage their hierarchical storage system. Using configuration information defined by the site, HPSS organizes storage devices into multiple storage hierarchies. Based on policy information defined by the site and actual usage information, data are then moved to the appropriate storage hierarchy and to appropriate levels in the storage hierarchy.

1.2 HPSS Capabilities

A central technical goal of HPSS is to move large files between storage devices and parallel or clustered computers at speeds many times faster than today's commercial storage system software products, and to do this in a way that is more reliable and manageable than is possible with current systems. In order to accomplish this goal, HPSS is designed and implemented based on the concepts described in the following subsections.

Network-centered Architecture

The focus of HPSS is the network, not a single server processor as in conventional storage systems. HPSS provides servers and movers that can be distributed across a high performance network to provide scalability and parallelism. The basis for this architecture is the IEEE Mass Storage System Reference Model, Version 5.

Third Party Data Transfer

Closely related to the network-centered architecture is a design allowing data to be transferred directly from an intelligent disk or tape controller to a client. Once a transfer session is established, the actual data transfer takes place directly between the client and the storage device controller. Normally this is done using IPI-3 third party protocols but TCP/IP will also work, and other protocols are being evaluated. Significantly, transferred data is not buffered in memory of a large storage management computer. The intelligent control unit may be packaged with the device, as with the Maximum Strategy disk array, or it may be a low-cost UNIX-based processor running HPSS Mover code. An HPSS Mover may buffer data but the distributed nature of HPSS allows multiple Movers thus eliminating the bottleneck of a single, large centralized server.

High Data Transfer Rate

HPSS achieves high data transfer rates by eliminating overhead normally associated with data transfer operations. In general, HPSS servers establish transfer sessions but are not involved in actual transfer of data. For network-attached storage devices supporting IPI-3 third party transfer protocols, HPSS Movers deliver data at device speeds. For example, with a single HiPPI-attached disk array supporting IPI-3 third party protocols, HPSS transfers a single data stream at over 50MB/sec.

Parallel Operation Built In

The HPSS Application Program Interface (API) supports parallel or sequential access to storage devices by clients executing parallel or sequential applications. HPSS also provides a Parallel File Transfer Protocol. HPSS can even manage data transfers in a situation where the number of data sources and destination are different. Parallel data transfer is vital in situations that demand fast access to very large files.

A Design Based on Standard Components

HPSS runs on UNIX with no kernel modifications and is written in ANSI C. It uses the OSF Distributed Computing Environment (DCE) and Encina from Transarc Corporation as the basis for its portable, distributed, transaction-based architecture. These components are offered on many vendors' platforms. Source code is available to vendors and users for porting HPSS to new platforms. HPSS Movers may be ported to non-DCE platforms. The first implementation of HPSS has been done on the IBMAIX platforms.

Data Integrity Through Transaction Management

Transaction management and Kerberos security enable a reliable design that protects data both from unauthorized use and from corruption due to data without pointers or pointers without data.

A transaction is an atomic grouping of functions that either take place together, or none of them take place. Journaling makes it possible to back out any partially complete transactions if a failure occurs. Transaction technology is common in relational data management systems but not in storage systems. HPSS implements transactions through Transarc's Encina product. Transaction management is the key to maintaining reliability and security while scaling upward into a large distributed storage environment.

Multiple Hierarchies and Classes of Services

Most other storage management systems support simple storage hierarchies consisting of one kind of disk and one kind of tape. HPSS provides multiple hierarchies, which are particularly useful when inserting new storage technologies over time. As new disks, tapes, or optical media are added, new classes of service can be set up. HPSS files reside in a particular class of service which users select based on parameters such as file size and performance. A class of service is implemented by a storage hierarchy which in turn consists of multiple storage classes, as shown in Figure 1-2. Storage classes are used to logically group storage media to provide storage for HPSS files. A hierarchy may be as simple as a single tape, or it may consist of two or more levels of disk, disk array, and local tape. The user can even set up classes of service so that data from an older type of tape is subsequently migrated to a new type of tape. Such a procedure allows migration to new media over time without having to copy all the old media at once.

1.3 HPSS Components

The components of HPSS include files, filesets, junctions, virtual volumes, physical volumes, storage segments, metadata, servers, infrastructure, user interfaces, a management interface, and policies. Storage and file metadata are represented by data structures that describe the attributes and characteristics of storage system components such as files filesets, junctions, storage segments, and volumes. Servers are the processes that control the logic of the system and control movement of the data. The HPSS infrastructure provides the services that are used by all the servers for standard operations such as sending messages and providing reliable transaction management. User interfaces provide several different views of HPSS to applications with different needs. The management interface provides a way to administer and control the storage system and implement site policy.

These HPSS components are discussed below in Sections 1.3.1 through 1.3.6.

1.3.1 HPSS Files, Filesets, Volumes, Storage Segments and Related Metadata

The components used to define the structure of the HPSS name space are filesets and junctions. The components containing user data include bitfiles, physical and virtual volumes, and storage segments. Components containing metadata, describing the attributes and characteristics of files, volumes, and storage segments, include storage maps, classes of service, hierarchies, and storage classes.

- ***Files (Bitfiles).*** Files in HPSS, called bitfiles in deference to IEEE Reference Model terminology, are logical strings of bytes, even though a particular bitfile may have a structure imposed by its owner. This unstructured view decouples HPSS from any particular file

management system that host clients of HPSS might have. HPSS bitfile size is limited to 2 to the power of 64 minus 1 (2^{64} - 1).

Each bitfile is identified by a machine-generated name called a bitfile ID. It may also have a human readable name. It is the job of the HPSS Name Server (discussed in Section 1.3.2) to map a human readable name to a bitfile's bitfile ID. By separating human readable names from the bitfiles and their associated bitfile IDs, HPSS allows sites to use different Name Servers to organize their storage. There is, however, a standard Name Server included with HPSS.

- ***Filesets.*** A fileset is a logical collection of files that can be managed as a single administrative unit, or more simply, a disjoint directory tree. A fileset has two identifiers: a human readable name, and a 64-bit integer. Both identifiers are unique to a given DCE cell.

- ***Junctions.*** A junction is an object, managed by the HPSS Name Server, that links a path name to a fileset. Junctions provide the mechanism to link filesets into the HPSS name space.

- ***File Families.*** A file family is an attribute of an HPSS file used to group a set of files on a common set of tape virtual volumes. Release 4.1.1 supports grouping files only on tape volumes. In Release 4.1.1, families can only be specified by associating the family with a fileset. Then all files created in the fileset belong to the family. When a file is migrated from disk to tape, it is associated with the files. If no family is associated with a file, the file is migrated to the next available tape not associated with any family. If no tape virtual volume is associated with the family, a blank tape is reassigned from the default family. The family affiliation is preserved when tapes are repacked.

- ***Physical Volumes.*** A physical volume is a unit of storage media on which HPSS stores data. The media can be removable (e.g., cartridge tape, optical disk) or non-removable (magnetic disk). Physical volumes are not visible to the end user. The end user simply stores bitfiles into a logically unlimited storage space. HPSS, however, must implement this storage on a variety of types and quantities of physical volumes.

  The physical volume types currently supported by HPSS are IBM 3480, 3490, 3490E, and 3590; StorageTek 4480, 4490, Timberline, and Redwood; SCSI disk; SSA disk; and Maximum Strategies Inc. and IBM 9570 HiPPI-attached disk arrays.

- ***Virtual Volumes.*** A virtual volume is used by the Storage Server to provide a logical abstraction or mapping of physical volumes. A virtual volume may include one or more physical volumes. A virtual volume is primarily used inside of HPSS, thus hidden from the user, but its existence benefits the user by making the user's data independent of device characteristics. Virtual volumes are organized as strings of bytes up to 2^{64-1} bytes in length that can be addressed by an offset into the virtual volume.

- ***Storage Segments.*** A storage segment is an abstract storage object which is mapped onto a virtual volume. Each storage segment is associated with a storage class (defined below) and has a certain measure of location transparency. The Bitfile Server (discussed in Section 1.3.2) uses both disk and tape storage segments as its primary method of obtaining and accessing HPSS storage resources. Mappings of storage segments onto virtual volumes are maintained by the HPSS Storage Servers (Section 1.3.2).

- ***Storage Maps.*** A storage map is a data structure used by the Storage Server to manage the allocation of storage space on virtual volumes.

- ***Storage Classes.*** A storage class defines a set of characteristics and usage parameters to be associated with a particular grouping of HPSS virtual volumes. Each virtual volume and its

associated physical volumes belong to a single storage class in HPSS. Storage classes in turn are grouped to form storage hierarchies (see below). An HPSS storage class is used to logically group storage media to provide storage for HPSS files with specific intended usage, similar size and usage characteristics.

- ***Storage Hierarchies.*** An HPSS storage hierarchy defines the storage classes on which files in that hierarchy are to be stored. A hierarchy consists of multiple levels of storage, with each level representing a different storage class. Files are moved up and down the hierarchy via migrate and stage operations based on usage patterns, storage availability, and site policies. For example, a storage hierarchy might consist of a fast disk, followed by a fast data transfer and medium storage capacity robot tape system, which in turn is followed by a large data storage capacity but relatively slow data transfer tape robot system. Files are placed on a particular level in the hierarchy depending on the migration policy (discussed in Section 1.3.6) and staging operations. The staging and migrating of data is shown in Figure 1-1.

- ***Class of Service (COS).*** Each bitfile has an important attribute called COS. The COS defines a set of parameters associated with operational and performance characteristics of a bitfile. The COS results in the bitfile being stored in a storage hierarchy suitable for its anticipated and actual size and usage characteristics. Figure 1-2 shows the relationship between COS, storage hierarchies, and storage classes.

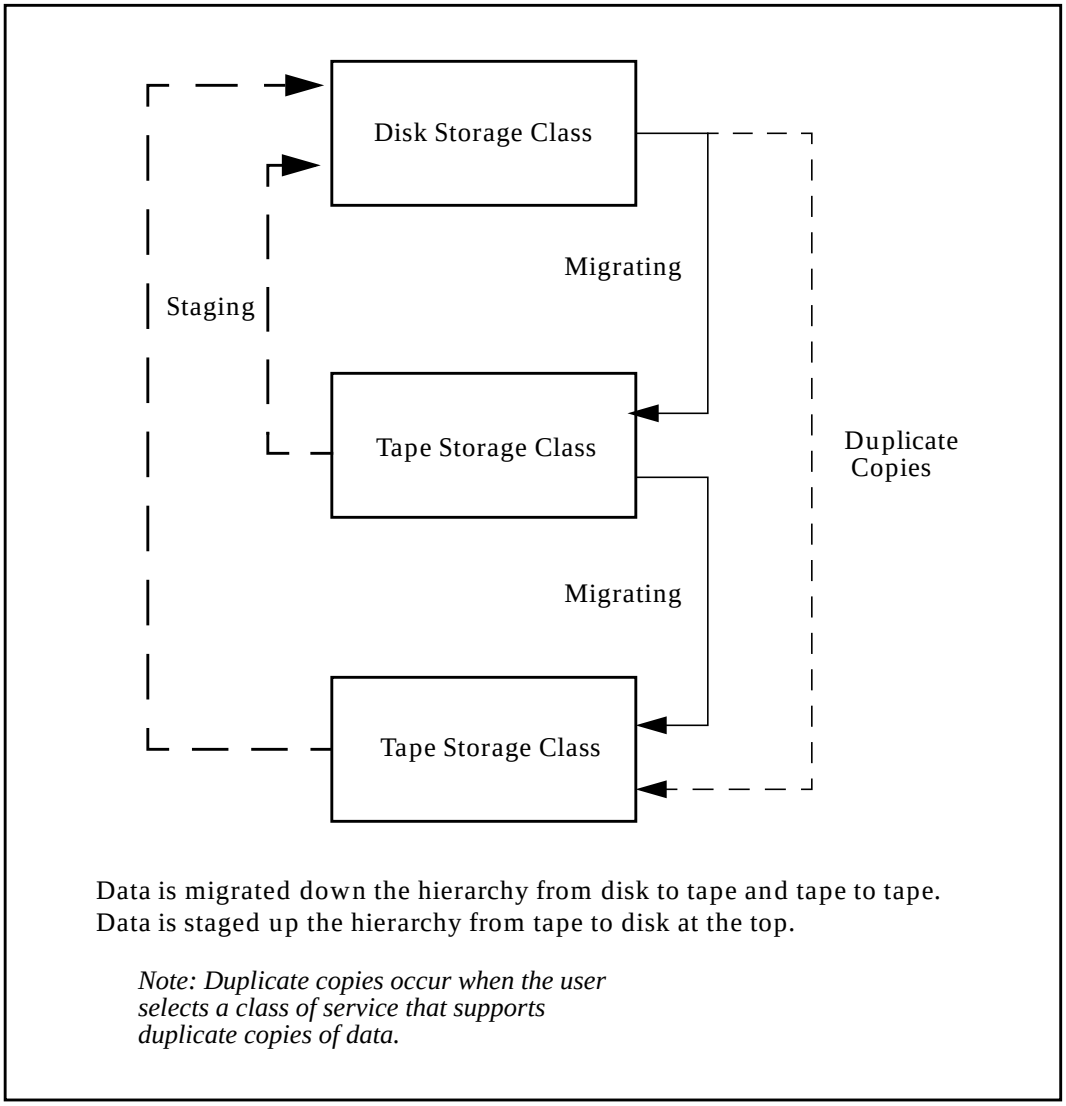


Figure 1-1 Migrate and Stage Operations

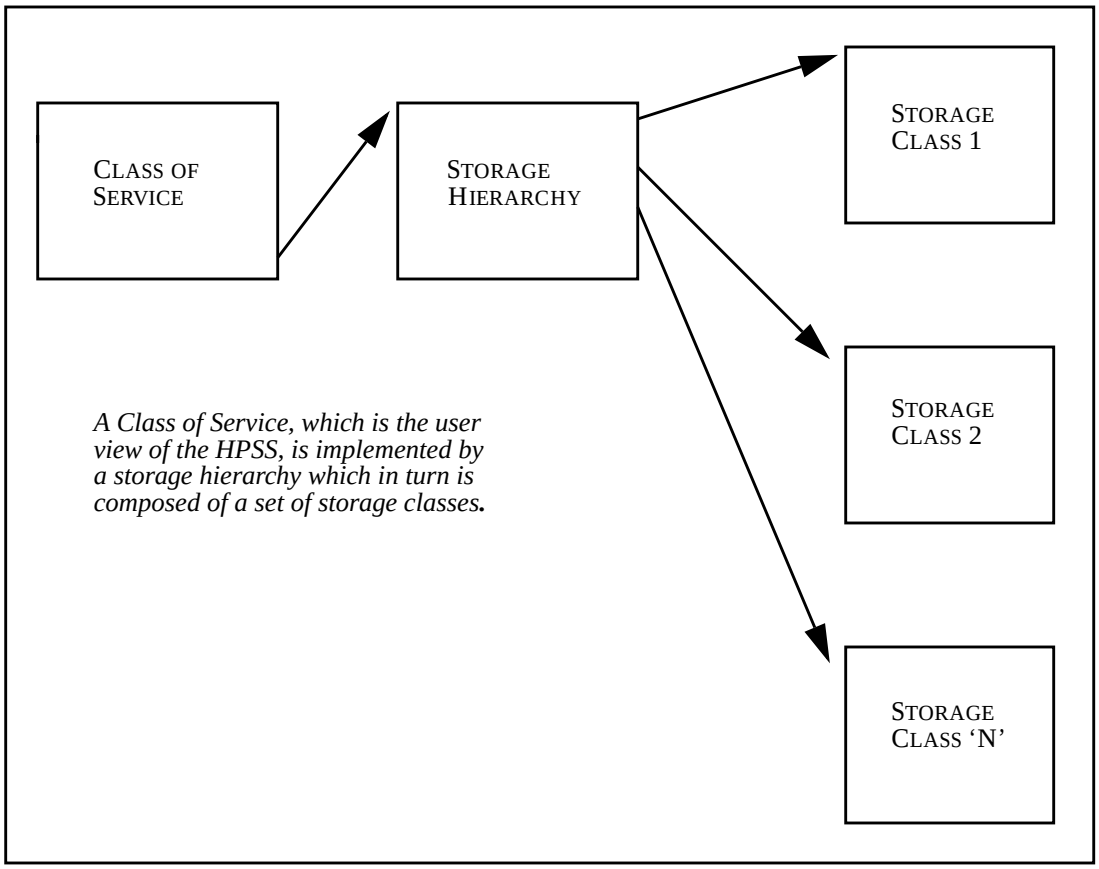


Figure 1-2 Relationship of HPSS Data Structures

1.3.2 HPSS Core Servers

HPSS servers include the Name Server, Bitfile Server, Migration/Purge Server, Storage Server, Location Server, DMAP Gateway, Physical Volume Library, Physical Volume Repository, Mover, Storage System Manager, and Non-DCE Client Gateway. Figure 1-3 provides a simplified view of the HPSS system. Each major server component is shown, along with the basic control communications paths (thin arrowed lines). The thick line reveals actual data movement. Infrastructure items (those components that "glue together" the distributed servers) are shown at the top of the cube in gray scale. These infrastructure items are discussed in Section 1.3.3. HPSS user interfaces (the clients listed in the figure) are discussed in Section 1.3.4.

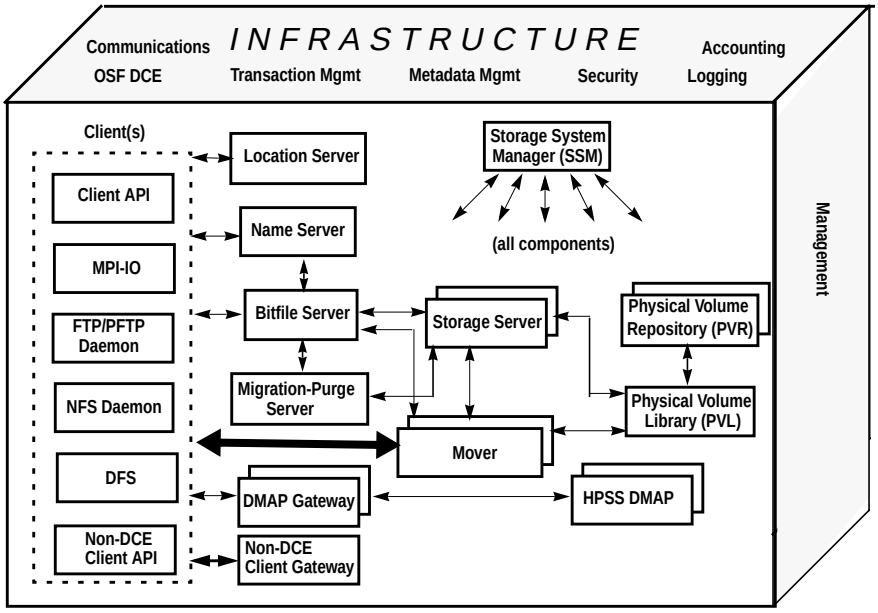


Figure 1-3 The HPSS System

- ***Name Server (NS).*** The NS translates a human-oriented name to an HPSS object identifier. Objects managed by the NS are files, filesets, directories, symbolic links, junctions and hard links. The NS also provides access verification to objects. The NS provides a Portable Operating System Interface (POSIX) view of the name space. This name space is a hierarchical structure consisting of directories, files, and links. Filesets allow collections of NS objects to be managed as a single administrative unit. Junctions are used to link filesets into the HPSS name space.

- ***Bitfile Server (BFS).*** The BFS provides the abstraction of logical bitfiles to its clients. A bitfile is identified by a BFS-generated name called a bitfile ID. Clients may reference portions of a bitfile by specifying the bitfile ID and a starting address and length. The reads and writes to a bitfile are random, and BFS supports the notion of holes (areas of a bitfile where no data has been written). The BFS supports parallel reading and writing of data to bitfiles. The BFS communicates with the storage segment layer interface of the Storage Server (see below) to support the mapping of logical portions of bitfiles onto physical storage devices. The BFS supports the migration, purging, and staging of data in a storage hierarchy.

- ***Migration/Purge Server (MPS).*** The MPS helps the local site implement its storage management policies by managing the placement of data on HPSS storage media using site-defined policy mechanisms. By issuing appropriate commands to the Bitfile Server and Storage Servers, MPS copies data to lower levels in the hierarchy (this is called data migration). The purge component of MPS releases space for data that has been copied successfully from disk storage classes. This is done by issuing appropriate purge commands to the Bitfile Server based on a site-defined purge policy. In addition, MPS can be directed by site policy to create duplicate copies of data when data is being migrated from disk. This is done by copying the data to one or more lower levels in the storage hierarchy.

There are two types of migration: disk migration and tape migration. The main purpose of disk migration is to free up the disk storage. The main purpose of tape migration is to free up tape volumes — not just migrate bitfiles. Disk migration contains two functions: migration and purge. The migration selects the qualified bitfiles or bitfile segments and copies these bitfiles to the next lower storage level defined in the hierarchy. The purge later frees the space currently utilized by the bitfiles on the disk storage. In tape migration, the active bitfiles in the target virtual volumes are moved laterally to the free tape volumes in the same storage level. The inactive bitfiles in the target virtual volumes are migrated to the free tape volumes in the next storage lower level.

The main purpose of duplicate copies of bitfiles is to allow users to automatically have backup copies of their critical data.

The MPS is set up to run the migration periodically using the time interval specified in the migration policy (see Section 1.3.6). In addition, the server will start the purge run automatically if the free space of a storage class is below the percentage specified in the purge policy.

- ***Storage Server (SS).*** The Storage Servers provide a hierarchy of storage objects: storage segments, virtual volumes, and physical volumes. The Storage Servers translates storage segment references into virtual volume references and then into physical volume references, handle s the mapping of physical resources into striped virtual volumes to allow parallel I/O to that set of resources, and schedules the mounting and dismounting of removable media through the Physical Volume Library (see below).

- ***Location Server (LS).*** The Location Server acts as an information clearinghouse to its clients to enable them to locate servers and gather information from both local and remote HPSS systems. It performs two main functions: Its primary function is to allow a client to determine a server's location (its CDS pathname) by knowing the server's unique identifier, its object UUID. This allows a client to contact the appropriate server, usually the Bitfile Server or the Name Server. The second function is to aid the clients in determining what Class of Service to select when creating bitfiles and to maintain storage statistics for all defined Classes of Service.

- ***DMAP Gateway (DMG)***. The DMAP Gateway acts as a conduit and translator between DFS and HPSS servers. It translates calls between DFS and HPSS, migrates data from DFS into HPSS, and validates data in DFS and HPSS. In addition, it maintains records of all DFS and HPSS filesets and their statistics.

- ***Physical Volume Library (PVL).*** The PVL manages all HPSS physical volumes. It is in charge of mounting and dismounting sets of physical volumes, allocating drive and cartridge resources to satisfy mount and dismount requests, providing a mapping of physical volume to cartridge and of cartridge to Physical Volume Repository (PVR), and issuing commands to PVRs to perform physical mount and dismount actions. A requirement of the PVL is the support for atomic mounts of sets of cartridges for parallel access to data. Atomic mounts are implemented by the PVL, which waits until all necessary cartridge resources for a request are available before issuing mount commands to the PVRs.

- ***Physical Volume Repository (PVR).*** The PVR manages all HPSS cartridges. Clients (e.g., the PVL) can ask the PVR to mount and dismount cartridges. Clients can also query the status and characteristics of cartridges. Every cartridge in HPSS must be managed by exactly one PVR. Multiple PVRs are supported within an HPSS system. Each PVR is typically configured to manage the cartridges for one robot utilized by HPSS.

The current HPSS release provides the following types of PVR:

- STK PVR – Manages the StorageTek 4400 Automated Cartridge Store, which mounts, dismounts, and manages 3480/3490 cartridges for a set of STK 4480/4490 drives.
- IBM 3494/3495 PVR – Manages the two IBM tape robots, which manage 3480 form factor cartridges. Form factor cartridges are square 1/2-inch tapes. The cartridges may be for IBM 3480, 3490, or 3590. The robots, while physically very different, are managed through identical interfaces.
- AML PVR – Manages the ADIC Automated Media Library robot which can handle a variety of cartridge types. Per ADIC the AML/E, AML/J and AML/2 robot interfaces are identical.
- Operator PVR – Manages a set of cartridges that are not under the control of a robotic device. These cartridges are mounted on a set of drives by operators.

- ***Mover (MVR).*** The purpose of the Mover is to transfer data from a source device to a sink device. A device can be a standard I/O device with geometry (e.g., tape, disk) or a device without geometry (e.g., network, memory). The MVR's client (typically the SS) describes the data to be moved and where the data is to be sent. It is the MVR's responsibility to actually transfer the data, retrying failed requests and attempting to optimize transfers. The MVR supports transfers for disk devices, tape devices, third-party transfers via the IPI-3 protocol (allowing data to be sent between a storage device and client without flowing through the controlling Mover) and a mover protocol that can be used as a lightweight coordination and flow control mechanism for large transfers.

- ***Storage System Management (SSM).*** SSM roles cover a wide range, including aspects of configuration, initialization, and termination tasks. The SSM monitors and controls the resources (e.g., servers) of the HPSS storage system in ways that conform to management policies of a given customer site. Monitoring capabilities include the ability to query the values of important management attributes of storage system resources and the ability to receive notifications of alarms and other significant system events. Controlling capabilities include the ability to start up and shut down servers and the ability to set the values of management attributes of storage system resources and storage system policy parameters. Additionally, SSM can request that specific operations be performed on resources within the storage system, such as adding and deleting logical or physical resources. Operations performed by SSM are usually accomplished through standard HPSS Application Program Interfaces (APIs).

  SSM has three components: (1) the System Manager, which communicates with all other HPSS components requiring monitoring or control, (2) the Data Server, which provides the bridge between the System Manager and the Graphical User Interface (GUI), and (3) the GUI itself, which includes the Sammi Runtime Environment and the set of SSM windows.

- ***Non-DCE Client Gateway (NDCG).*** NDCG provides an interface into HPSS for client programs running on systems without either DCE or Encina. By linking the Non-DCE Client API library, instead of the usual Client API library, all API calls are routed through the NDCG. The API calls are then executed by the NDCG, and the results are returned to the client application. Note that the NDCG must run on a system with DCE and Encina, while the client application using the Non-DCE Client API does not have this restriction.

1.3.3 HPSS Infrastructure

The HPSS infrastructure items (see Figure 1-3) are those components that "glue together" the distributed servers. While each HPSS server component provides some explicit functionality, they

must all work together to provide users with a stable, reliable, and portable storage system. The HPSS infrastructure components common among servers that tie servers together are discussed below.

- ***Distributed Computing Environment (DCE).*** HPSS uses the Open Software Foundation's Distributed Computing Environment (OSF DCE) as the basic infrastructure for its architecture and high-performance storage system control. DCE was selected because of its wide adoption among vendors and its near industry-standard status. HPSS uses the DCE Remote Procedure Call (RPC) mechanism for control messages and the DCE Threads package for multitasking. The DCE Threads package is vital for HPSS to serve large numbers of concurrent users and to enable multiprocessing of its servers. HPSS also uses DCE Security and Cell Directory services.

  All HPSS servers, with the exception of the MVR and the logging services (see below), communicate requests and status (control information) via RPCs. *HPSS does not use RPCs to move user data.* RPCs provide a communication interface resembling simple, local procedure calls.

- ***Transaction Management.*** Requests to perform actions, such as creating bitfiles or accessing file data, result in client-server interactions between software components. The problem with distributed servers working together on a common job is that one server may fail or not be able to do its part. When such an event occurs, it is often necessary to abort the job by backing off all actions made by all servers on behalf of the job.

  Transactional integrity to guarantee consistency of server state and metadata is required in HPSS in case a particular component fails. As a result, a product named Encina, from Transarc Corporation, was selected to serve as the HPSS transaction manager. This selection was based on functionality and vendor platform support. Encina provides begin-commit-abort semantics, distributed two-phase commit, and nested transactions. It provides HPSS with an environment in which a job or action that requires the work of multiple servers either completes successfully or is aborted completely within all servers.

- ***Metadata Management.*** Each HPSS server component has system state and resource data (metadata) associated with the objects it manages. Each server with non-volatile metadata requires the ability to reliably store its metadata. The metadata management performance must also be able to scale as the number of object instances grow. In addition, access to metadata by primary and secondary keys is required.

  The Encina Structured File Server (SFS) product serves as the HPSS Metadata Manager (MM). SFS provides B-tree clustered file records, relative sequence file records, record and field level access, primary and secondary keys, and automatic byte ordering between machines. SFS is also fully integrated with the Encina's transaction management capabilities. As a result, SFS provides transactional consistency and data recovery from transaction aborts. An HPSS component called the Metadata Monitor (MMON) provides monitoring of the HPSS Encina SFS, including generating notifications when configurable metadata space usage thresholds are exceeded.

- ***Security.*** HPSS software security provides mechanisms that allow HPSS components to communicate in an authenticated manner, to authorize access to HPSS objects, to enforce access control on HPSS objects, and to issue log records for security-related events. The security components of HPSS provide authentication, authorization, enforcement, audit, and security management capabilities for the HPSS components. Customer sites may use the default security policy delivered with HPSS or define their own security policy by implementing their own version of the security policy module.

- *Authentication* — is responsible for guaranteeing that a principal (a customer identity) is the entity that is claimed, and that information received from an entity is from that entity. An additional mechanism is provided to allow authentication of File Transfer Protocol (FTP) users through a site-supplied policy module.
- *Authorization* — is responsible for enabling an authenticated entity access to an allowed set of resources and objects. Authorization enables end user access to HPSS directories and bitfiles.
- *Enforcement* — is responsible for guaranteeing that operations are restricted to the authorized set of operations.
- *Audit* — is responsible for generating a log of security-relevant activity. HPSS audit capabilities allow sites to monitor HPSS authentication, authorization, and file security events. File security events include file creation, deletion, opening for I/O, and attribute modification operations.
- *Security management* — allows the HPSS administrative component to monitor the underlying DCE security services used by the HPSS security subsystem.

HPSS components that communicate with each other maintain a joint security context. The security context for both sides of the communication contains identity and authorization information for the peer principals as well as an optional encryption key. The security context identity and authorization information is obtained using DCE security and RPC services.

Access to HPSS server interfaces is controlled through an Access Control List (ACL) mechanism. Access for HPSS bitfile data is provided through a distributed mechanism whereby a user's access permissions to an HPSS bitfile are specified by the HPSS bitfile authorization agent, the Name Server. These permissions are processed by the bitfile data authorization enforcement agent, the Bitfile Server. The integrity of the access permissions is certified by the inclusion of a checksum that is encrypted using the security context key shared between the HPSS Name Server and Bitfile Server.

- ***Logging.*** A logging infrastructure component in HPSS provides an audit trail of server events. Logged data includes alarms, events, requests, security audit records, status records, and trace information. Servers send log messages to a Log Client (a server executing on each hardware platform containing servers that use logging). The Log Client, which may keep a temporary local copy of logged information, communicates log messages to a central Log Daemon, which in turn maintains a central log. Depending on the type of log message, the Log Daemon may send the message to the SSM for display purposes. When the central HPSS log fills, messages are sent to a secondary log file. A configuration option allows the filled log to be automatically archived to HPSS. A delog function is provided to extract and format log records. Delog options support filtering by time interval, record type, server, and user.

- ***Accounting.*** The primary purpose of the HPSS accounting system is to provide the means to collect information on usage in order to allow a particular site to charge its users for the use of HPSS resources.

  For every account index, the storage usage information is written out to an ASCII text file. It is the responsibility of the individual site to sort and use this information for subsequent billing based on site- specific charging policies. For more information on the HPSS accounting policy, refer to Section 1.3.6.

1.3.4 HPSS User Interfaces

As indicated in Figure 1-3, HPSS provides the user with a number of transfer interfaces as discussed below.

- ***File Transfer Protocol (FTP).*** HPSS provides an industry-standard FTP user interface. Because standard FTP is a serial interface, data sent to a user is received serially. This does not mean that the data within HPSS is not stored and retrieved in parallel; it simply means that the FTP Daemon within HPSS must consolidate its internal parallel transfers into a serial data transfer to the user. HPSS FTP performance in many cases will be limited not by the speed of a single storage device, as in most other storage systems, but by the speed of the data path between the HPSS FTP Daemon and the user's FTP client.

- ***Network File System (NFS) V2.*** The NFS V2 server interface for HPSS provides transparent access to HPSS name space objects and bitfile data for client systems through the NFS V2 service. The NFS implementation consists of the server interface for NFS V2 clients and the NFS mount service for HPSS, plus server support functions that are not accessible to NFS clients. The HPSS NFS service will work with any industry-standard NFS V2 client.

- ***Parallel FTP (PFTP).*** The PFTP supports standard FTP commands plus extensions and is built to optimize performance for storing and retrieving files from HPSS by allowing data to be transferred in parallel across the network media. The parallel client interfaces have a syntax similar to FTP but with some extensions to allow the user to transfer data to and from HPSS across parallel communication interfaces established between the FTP client and the HPSS Movers. This provides the potential for using multiple client nodes as well as multiple server nodes. PFTP supports transfer either via TCP/IP or IPI-3 over HiPPI. In either case, the FTP client communicates directly with HPSS Movers to transfer data at rates limited only by the underlying communications hardware and software.

- ***Client Application Program Interface (Client API).*** The Client API is an HPSS-specific programming interface that mirrors the POSIX.1 specification where possible to provide ease of use to POSIX application programmers. Additional APIs are also provided to allow the programmer to take advantage of the specific features provided by HPSS (e.g., storage/access hints passed on file creation and parallel data transfers). The Client API is a programming level interface. It supports file open/create and close operations; file data and attribute access operations; file name operations; directory creation, deletion, and access operations; and working directory operations. HPSS users interested in taking advantage of parallel I/O capabilities in HPSS can add Client API calls to their applications to utilize parallel I/O. For the specific details of this interface see *HPSS Programmer's Reference Guide*, Volume 1, Release 4.1.1.

- ***Non-DCE Client Application Program Interface (Non-DCE Client API).*** The Non-DCE Client API is a programming interface that allows the client program the option of running on a platform that does not support DCE and Encina. This API does not call HPSS directly. Instead, it sends network messages to the Non-DCE Client Gateway, which runs on the target HPSS system. The NDCG then performs the requested Client API calls for the client and returns the results. The Non-DCE Client API has the same functionality as the standard Client API with the following exceptions: it does not support ACL functions, it does not support transactional processing, and it does not implement any security. Security is to be added in Release 4.2.

- ***MPI-IO Application Programming Interface (MPI-IO API)***. The MPI-IO API is a subset of the MPI-2 standard. It gives applications written for a distributed memory programming model an interface that offers coordinated access to HPSS files from multiple processes. These

processes can read and write data from a single file in parallel using HPSS third-party transfer facilities. The interface also lets applications specify discontiguous patterns of access to files and memory buffers using the same "datatype" constructs that the Message-Passing Interface (MPI) offers. For the specific details of this interface, see the *HPSS Programmer's Reference Guide*, Volume 1, Release 4.1.1.

- ***Distributed File System (DFS***. Distributed file system services are provided by optionally allowing HPSS to interface with Transarc's DFSTM. DFS is a highly scalable distributed file system that provides a uniform view of file data to all users through a global name space. DFS uses the DCE concept of cells, and allows data access and authorization between clients and servers in different cells. DFS uses the Episode physical file system, although it can use other native file systems, such as UFS.

  A standard interface, XDSM, is used to couple DFS with HPSS. The standard was previously called DMAPI and originated in the mid-nineties from the Data Management Interfaces Group (DMIG). XDSM is a low-level interface to the physical file system. It provides a Data Management Application (DMAP) with the ability to store important attribute information with a file and allows for the generation of notifications to the DMAP on occurrence of various file system operations. XDSM enables the DMAP to control disk storage by allowing the DMAP to move disk-resident file data to tertiary storage systems and vice-versa. The architecture used to integrate HPSS with the Episode file system is shown in Figure 1.4.

  The integrated HPSS/DFS system provides transparent, automatic archiving and caching of data between DFS and HPSS, supports partially-resident file data in both DFS and HPSS, allows changes made to a file or directory through DFS interfaces to be visible through HPSS interfaces and vice versa, and continues to provide single file transfer rates over 1 GB/sec.

  If a site elects to use DFS, two additional HPSS components must be configured into the HPSS system: the HPSS/DMAP (HDM) and the DMAP Gateway. In addition, the XDSM implementation for DFS, called the DFS Storage Management Toolkit (DFS SMT) is required.

  DFS SMT is a fully compliant implementation of the standard XDSM specification. In addition, it provides optional features; persistent opaque DM attributes, persistent event masks, persistent managed regions, non-blocking lock upgrades and the ability to scan for objects with a particular attribute. This component maintains event queues, and the meta data describing the events for which various file systems have registered. It is responsible for receiving events and dispatching them to the DMAP.

  HDM is responsible for initiating and coordinating name and data space interactions between Episode and HPSS. It catches and processes desired name and data space events generated by Episode sent through the DFS SMT; migrates file data from Episode to HPSS; purges unneeded Episode file data after the data migrates to HPSS; and processes requests originating in HPSS interfaces that require either Episode name or data resources. HDM resides on the same machine as the disk(s) managed by the Episode file system.

  DMAP Gateway is a conduit between HDM and HPSS. HPSS servers use DCE/RPCs, the Encina transaction manager, and Transarc's SFS for metadata managing HPSS objects. These resources may not be available for DMAPs supporting file systems other than DFS. The DMAP Gateway translates requests between XDR and sockets and DCE/TRPC/Encina, depending on where the client request originates. In addition, the DMAP Gateway keeps statistics on fileset requests and internal DMAP Gateway resource utilization. Connections between the HDM and Gateway are mutually authenticated.

1.3.5 HPSS Management Interface

HPSS provides a powerful SSM administration and operations GUI through the use of the Sammi product from Kinesix Corporation. Detailed information about Sammi can be found in the *Sammi Runtime Reference*, *Sammi User's Guide*, and *Sammi System Administrator's Guide.*

SSM simplifies the management of HPSS by organizing a broad range of technical data into a series of easy-to-read graphic displays. SSM allows monitoring and control of virtually all HPSS processes and resources from windows that can easily be added, deleted, moved, or overlapped as desired.

1.3.6 HPSS Policy Modules

There are a number of aspects of storage management that probably will differ at each HPSS site. For instance, sites typically have their own guidelines or policies covering the implementation of accounting, security, and other storage management operations. In order to accommodate site-specific policies, HPSS has implemented flexible interfaces to its servers to allow local sites the freedom to tailor management operations to meet their particular needs.

HPSS policies are implemented using two different approaches. Under the first approach—used for migration, purge, and logging policies—sites are provided with a large number of parameters that may be used to implement local policy. Under the second approach—used for security policy—HPSS communicates information through a well-defined interface to a policy software module that can be completely replaced by a site. Under both approaches, HPSS provides a default policy set for users.

- ***Migration Policy***. The migration policy defines the conditions under which data is copied from one level in a storage hierarchy to one or more lower levels. Each storage class that is to have data copied from that storage class to a lower level in the hierarchy has a migration policy associated with it. The MPS uses this policy to control when files are copied and how much data is copied from the storage class in a given migration execution. Migration runs are started automatically by the MPS based upon parameters in the migration policy.

- ***Purge Policy.*** The purge policy defines the conditions under which data that has already been migrated from a disk storage class can be deleted. Purge applies only to disk storage classes. Each disk storage class which has a migration policy should also have a purge policy. Purge runs are started automatically by the MPS based upon parameters in the purge policy.

- ***Logging Policy.*** The logging policy controls the types of messages to log. On a per server basis, the message types to write to the HPSS log may be defined. In addition, for each server, options to send Alarm, Event, or Status messages to SSM may be defined.

- ***Security Policy.*** In pre 4.1 releases, the security policy allowed a site the option of using the delivered security policy or using their own security mechanism by implementing a security module. Site security policy defines the authorization and access controls to be used for client access to HPSS. Site policy managers were developed for controlling access from FTP and/or Parallel FTP using either **Ident** or **Kerberos** credentials. Effective with HPSS release 4.1, these access methods are supported by request using the **hpss_pftpd_amgr** and an appropriate authentication manager. The Policy Manager is no longer supported.

HPSS server authentication and authorization use DCE authentication and authorization mechanisms. Each HPSS server has configuration information that determines the type and level of DCE security services available / required for the individual server. HPSS software uses DCE services to determine a caller's identity via credentials passed by the caller to the server. Once the identity and authorization information has been obtained, each HPSS server grants / denies the caller's request based on the access control list (ACLs) attached to the Security object in the server's Cell Directory Service (CDS) entry. Access to the interfaces that modify a server's internal metadata, generally require control permission. HPSS security is only as good as the security employed in the DCE cell!

HPSS provides facilities for recording information about authentication and object (file / directory) creation, deletion, access, and authorization events. The security audit policy for each server determines the records that each individual server will generate. All servers can generate authentication records, while only the Name and Bitfile Servers generate other object event records.

- ***Accounting Policy.*** The accounting policy provides runtime information to the accounting program. Accounting style is indicated by the setting of the **GECOS** field in the DCE registry or by the HPSS.gecos Extended Registry Attribute (ERA) in the DCE Registry.

  The two types of accounting are site- style and UNIX-style. The site-style approach is the traditional type of accounting in use by most mass storage systems. Each site will have a site-specific table (Account Map) that correlates the HPSS account index number with their local account charge codes. The UNIX-style approach allows a site to use the user identifier (UID) for the account index. The UID is passed along in UNIX-style accounting just as the account index number is passed along in site-style accounting. The **hpss_Chown** API or FTP **quote site chown** command can be used to assign a file to a new owner.

  The HPSS accounting policy also includes an **Accounting Style** field, currently unused, that will indicate whether the site is doing site-style accounting or UNIX-style accounting. Future releases of HPSS may use this field to control the accounting style selection and to guarantee consistency of the registry based on the style selection.

- ***Location Policy.*** The location policy defines how Location Servers at a given site will perform, especially in regards to how often server location and storage statistic information is updated. All local, replicated Location Servers update information according to the same policy.

1.4 HPSS Hardware Platforms

HPSS was designed with portability and use of industry standards as prime requirements. HPSS should be portable to any server platform having DCE and Encina. Source code licenses are being made available so that HPSS software can be ported to other computer manufacturers' platforms and resold by other vendors.

Client Platforms Supported

HPSS provides FTP and NFS V2 client access to any hardware platform running standard FTP and NFS V2 clients. The Parallel FTP currently has been ported to IBM AIX, Sun Solaris, Cray UniCOS,

Intel Paragon and Teraflop, Digital Unix, NEC SX-4, Hewlett-Packard HPUX, Silicon Graphics IRIX (32 Bit) and Linux for Intel and Alpha platforms.

A full-function Client API is available presently on IBM AIX platforms. However, it can be ported to any platform with UNIX, DCE and Encina.

The Non-DCE Client API is currently available for AIX and has been ported to IRIX.

The PFTP and Client API source code for the non-AIX platforms listed above is not on the HPSS distribution tape, but is available to HPSS customers by special arrangement at no additional fee. Maintenance of the PFTP and Client API software on these platforms is the responsibility of the customer, unless a support agreement is negotiated with IBM. Contact IBM for information on how to obtain the software mentioned above.

To utilize the DFS interface, a machine where the DFS server will be run is required. Although DFS is available on a number of platforms, the implementation of DFS that includes DFS SMT is only available on AIX. The DFS client may run on any platform that supports DFS.

Server and Mover Platforms Supported

HPSS currently requires at least one AIX machine for the core server components. The core server machine must include DCE and Encina as prerequisites and must have sufficient processing power and memory to handle the work load needed by HPSS.

HPSS is a distributed system. The main component that is replicated is the Mover, used for logical network attachment of storage devices. The Mover is currently supported in two modes of operation. In the first mode, the Mover is supported on AIX and Solaris platforms, which require DCE and Encina services. In the second mode, a portion of the Mover runs on the AIX or Solaris platform (again requiring DCE and Encina), but the portion of the Mover that manages HPSS devices and transfers data to/from clients runs on platforms that do not require DCE or Encina services. This second portion of the Mover that handles data transfers is supported on AIX, IRIX and Solaris.

Chapter 2

HPSS Planning

2.1 Overview

This chapter provides HPSS planning guidelines and considerations to help the administrator effectively plan, and make key decisions about, an HPSS system. Topics include:

- Requirements and Intended Usages for HPSS (Section 2.2)
- Prerequisite Software Considerations (Section 2.3)
- Hardware Considerations (Section 2.4)
- HPSS Interface Considerations (Section 2.5)
- HPSS Server Considerations (Section 2.6)
- Storage Policy Considerations (Section 2.7)
- Storage Characteristics Considerations (Section 2.7.6)
- HPSS Sizing Considerations (Section 2.9)
- HPSS Performance Considerations (Section 2.10)

The planning process for HPSS must be done carefully to ensure that the resulting system satisfies the site's requirements and operates in an efficient manner. We recommend that the administrator read the entire document before planning the system. In addition, we recommend that the administrator use the planning worksheets provided in Appendix C to record all planning decisions and also use the worksheets during installation and configuration.

The following paragraphs describe the recommended planning steps for the HPSS installation, configuration, and operational phases.

2.1.1 HPSS Installation Planning

The following planning steps must be carefully considered for the HPSS installation phase:

1. Identify the installation node to install the entire contents of the HPSS distribution media.

2. By default, HPSS files will be installed in the **/usr/lpp/hpss** directory. Ensure there is sufficient disk space on the installation node to install the HPSS files. Refer to Section 2.9.3.1 for more information to determine the disk space needed for the HPSS installation.

3. While HPSS is running, it requires additional disk space to store operational information such as log files, reports, etc. in the **/var/hpss** directory. Ensure that there is sufficient disk space available for HPSS servers to store these files. Refer to 2.9.3.2 for more information on the disk space required to support HPSS operationally.

4. Define the access policy for the HPSS software. That is, identify which UIDs will be used to bring up the HPSS servers and daemons. In addition, determine which UIDs will be allowed to bring up an SSM session. Based on these decisions, define the file ownerships and permissions for the to be installed HPSS files. Refer to 3.4.1 for more information on the recommended HPSS file ownerships and permissions.

Use the HPSS Installation worksheets in Appendix C to document the planning decisions to later use during the installation phase.

2.1.2 HPSS Configuration Planning

The following planning steps must be carefully considered for the HPSS infrastructure configuration and the HPSS configuration phases:

1. Identify the site's storage requirements and policies, such as the initial storage system size, anticipated growth, usage trends, average file size, expected throughput, and backup policy. Refer to Section 2.2 for more information.

2. Define the architecture of the entire HPSS system to satisfy the above requirements. Document the HPSS architecture in the HPSS Architecture Planning worksheet in Appendix C. The planning should:

   - Identify the nodes to be configured as part of the HPSS system.

   - Identify the disk and tape storage devices to be configured as part of the HPSS system and the nodes / networks to which each of the devices will be attached. Storage devices can be assigned to a number of nodes to allow data transfers to utilize the devices in parallel without being constrained by the resources of a single node. This capability also allows the administrator to configure the HPSS system to match the device performance with the network performance used to transfer the data between the HPSS Movers and the end users (or other HPSS Movers in the case of internal HPSS data movement for migration and staging). Refer to Section 2.4 for more discussions on the storage devices and networks supported by HPSS.

   - Identify the HPSS servers to be configured and the node where each of the servers will run. Use the anticipated growth information to determine the number of Tape Storage Servers to configure. While it may be administratively easier to have one Tape Storage Server managing all tape storage classes, it will be very difficult to add another Tape Storage Server later to balance the workload as the tape system grows very large. Refer to Section 2.6 for more discussions on the HPSS server configuration.

   - Identify the HPSS user interfaces (e.g. FTP, PFTP, NFS, DFS) to be configured and the nodes where the components of each user interface will run. Refer to Section 2.5 for more discussion on the user interfaces supported by HPSS.

3. Ensure that the prerequisite software has been installed and properly configured to satisfy the identified HPSS architecture. Refer to Section 2.3 for more information on the HPSS prerequisite software requirements.

4. Determine the SFS space needed to satisfy the requirements of the HPSS system. In addition, verify that there is sufficient disk space to configure the space needed by SFS. Refer to Section 2.9.2 for more discussions of SFS sizing required by HPSS.

5. Verify that each of the identified nodes has sufficient resources to handle the work loads and resources to be imposed on the node. Refer to Section 2.9.3 for more discussions on the system resource requirements.

6. Define the HPSS storage characteristics and create the HPSS storage space to satisfy the site's requirements:

    - Define the HPSS file families. Refer to Section 2.8.4 for more information about configuring families.

    - Define filesets and junctions. Refer to Sections 6.5 and 7.4 for more information.

    - Define the HPSS storage classes. Refer to Section 2.8.1 for more information on the storage class configuration.

    - Define the HPSS storage hierarchies. Refer to Section 2.8.2 for more information on the storage hierarchy configuration.

    - Define the HPSS classes of service. Refer to Section 2.8.3 for more information on the class of service configuration.

    - Define the migration and purge policy for each storage class. Refer to Section 2.7.1 and Section 2.7.2 for more information on the Migration Policy and Purge Policy configuration, respectively.

    - Determine the HPSS storage space needed for each storage class. Refer to Section 2.9.1 for more information on the HPSS storage space considerations.

    - Identify the disk and tape media to be imported into HPSS to allow for the creation of the needed storage space.

7. Define the location policy to be used. Refer to Section 2.7.6 for more information.

8. Define the accounting policy to be used. Refer to Section 2.7.3 for more information on the Accounting Policy configuration.

9. Define the logging policy for the each of the HPSS servers. Refer to Section 2.7.5 for more information on the Logging Policy configuration.

10. Define the security policy for the HPSS system. Refer to Section 2.7.4 for more information on the Security Policy for HPSS.

Use the HPSS Infrastructure Configuration worksheet in Appendix C to document the planning decisions to later use during the HPSS configuration phases.

2.1.3 HPSS Operational Planning

The following planning steps must be carefully considered for the HPSS operational phase:

1. Define the site policy for the HPSS users and SSM users.

   - Each HPSS user who uses the storage services provided by HPSS should be assigned an Accounting ID and one or more appropriate Classes of Service (COS) to store files.

   - Each SSM user (usually an HPSS administrator or an operator) should be assigned an appropriate SSM security level. The SSM security level defines what functions each SSM user can perform on HPSS through SSM. Refer to Appendix H for more information on setting up the security level for an SSM user.

2. Define the site policy and procedure for repacking and reclaiming HPSS tape volumes. This policy should consider how often to repack and reclaim HPSS volumes in the configured tape storage classes. In addition, the policy must consider when the repack and reclaim should be performed to minimize the impact on the HPSS normal operations. Refer to Section 6.4.7 and Section 6.4.8 for more information on repacking and reclaiming tape volumes.

3. Define the site policy and procedure for generating the accounting reports. This policy should consider how often an accounting report needs to be generated, how to use the accounting information from the report to produce the desired cost accounting, and whether the accounting reports need to be archived. Refer to Section 2.7.3 and Section 6.8 for more information on defining an Accounting Policy and generating accounting reports.

4. Verify that the number of configured Tape Storage Servers is sufficient for the anticipated tape system growth. While it may be administratively easier to have one Tape Storage Server managing all tape storage classes, it will be very difficult to add another Tape Storage Server later to balance the workload as the tape system grows very large.

2.2 Requirements and Intended Usages for HPSS

This section provides some guidance for the administrator to identify the site's requirements and expectation of HPSS. Issues such as the amount of storage needed, access speed and data transfer speed, typical usage, security, expected growth, data backup, and conversion from an old system must be factored into the planning of a new HPSS system.

2.2.1 Storage System Capacity

The amount of HPSS storage space the administrator must plan for is the sum of the following factors:

- The amount of storage data from anticipated new user accounts.

- The amount of new storage data resulting from normal growth of current accounts.

- The amount of storage space needed to support normal storage management such as migration and repack.
- The amount of storage data to be duplicated.

Another component of data storage is the amount of metadata space needed for user directories and other HPSS metadata. Much of this data must be duplicated (called “mirroring”) in real time on separate storage devices. Refer to Section 2.9.1 and Section 2.9.2 for more information on determining the needed storage space and metadata space.

2.2.2 Required Throughputs

To determine the required or expected throughput for the various types of data transfers that the users will perform. Some users want quick access to small amounts of data. Other users have huge amounts of data they want to transfer quickly, but are willing to wait for tape mounts, etc. In all cases, plan for peak loads that can occur during certain time periods. These findings must be used to determine the type of storage devices and network to be used with HPSS to provide the needed throughput.

2.2.3 Load Characterization

Understanding the kind of load users are putting on an existing file system provides input that can be used to configure and schedule the HPSS system. What is the distribution of file sizes? How many files and how much data is moved in each category? How does the load vary with time (e.g., over a day, week, month)? Are any of the data transfer paths saturated?

Having this storage system load information helps to configure HPSS so that it can meet the peak demands. Also based on this information, maintenance activities such as migration, repack, and reclaim can be scheduled during times when HPSS is less busy.

2.2.4 Usage Trends

To configure the system properly the growth rates of the various categories of storage, as well as the growth rate of the number of file accessed and data moved in the various categories, must be known. Extra storage and data transfer hardware must be available if the amount of data storage and use are growing rapidly.

2.2.5 Duplicate File Policy

The policy on duplicating critical files that a site uses impacts the amount of data stored and the amount of data moved. If all user files are mirrored, the system will require twice as many tape devices and twice as much tape storage. If a site let the users control their own duplication of files, the system may have a smaller amount of data duplicated depending on user needs. Users can be given control over duplication of their files by allowing them a choice between hierarchies which provide duplication and hierarchies which do not. Note that only files on disk can be duplicated to tapes and only if their associated hierarchies are configured to support multiple copies.

2.2.6 Charging Policy

HPSS does not do the actual charging of users for the use of storage system resources. Instead, it collects information that a site can use to implement a charging policy for HPSS use. The amount charged for storage system use also will impact the amount of needed storage. If there is no charge for storage, users will have no incentive to remove files that are outdated and more data storage will be needed.

2.2.7 Security

The process of defining security requirements is called developing a site security policy. It will be necessary to map the security requirements into those supported by HPSS. HPSS authentication, authorization, and audit capabilities can be tailored to a site needs.

Authentication and authorization between HPSS servers is done through use of DCE cell security authentication and authorization services. By default, servers are authenticated using the DCE secret authentication service, and authorization information is obtained from the DCE privilege service. The default protection level is to pass authentication tokens on the first remote procedure call to a server. The authentication service, authorization service, and protection level for each server can be configured to raise or lower the security of the system. Two cautions should be noted: (1) raising the protection level to packet integrity or packet privacy will require additional processing for each RPC, and (2) lowering the authentication service to none effectively removes the HPSS authentication and authorization mechanisms.This should only be done in a trusted environment.

Each HPSS server authorizes and enforces access to its interfaces through access control lists attached to an object (named Security) that is contained in its CDS directory. To be able to modify server state, control access is required. Generally, this is only given to the DCE principal associated with the HPSS system administrative component. Additional DCE principals can be allowed or denied access by setting permissions appropriately. See each server advanced configuration for permissions required. If the authorization by name service is used, only user type ACL entries should be specified.

Security auditing in each server may be configured to record all, none, or some security event. Some sites may choose to log every client connection; every bitfile creation, deletion, and open; and every file management operation. Other sites may choose to log only errors. See the security information fields in the general server configuration (Section 5.3.1) for more details.

User access to HPSS interfaces depend on the interface being used. Access through DFS and the native Client API uses the DCE authentication and authorization services described above. Access through NFS is determined based on how the HPSS directories are exported. Refer to Appendix I for more information on NFS exports and the **nfsmap** utility (Section I.40). FTP or Parallel FTP access may utilize an FTP password file or may utilize the DCE Registry. Additional FTP access is available using Ident, Kerberos GSS credentials, or DCE GSS credentials. The **Ident** and GSS authentication methods require running the hpss_pftpd_amgr server and an associated authentication manager in place of the standard hpss_pftpd. Refer to FTP documentation for additional details.

2.2.8 Cross Cell Access

DCE provides facilities for communication between multiple DCE cells (domains) referred to as “Cross Cell”. HPSS provides features to take advantage of the Cross Cell features. This provides

authentication enhancements and scalability opportunities. The procedures for inter-connecting DCE cells are outlined in Appendix G.

HPSS DFS facilities will be able to take advantage of these Cross Cell features.

The Generic Security Service (GSS) FTP and Parallel FTP applications can take advantage of the Cross Cell features for authentication and authorization. This requires using the hpss_pftpd_amgr server and an associated authentication manager in place of the standard hpss_pftpd. The GSSFTP from MIT Kerberos, the krb5_gss_pftp_client applications may use the Cross Cell features (subject to certain caveats! – See FTP documentation for details.)

Remote HPSS services may execute in different DCE cells or in the same DCE cell that contains the base HPSS servers. Thus it is feasible to execute a Startup Daemon, a Log Client, a Mover, and potentially other HPSS servers (PVRs, Storage Servers, etc) on a remote system (which may be executing a completely separate HPSS).

Only one Bitfile Server, one Name Server, and one PVL is allowed within a single HPSS system! A single HPSS Server cannot be part of two distinct HPSS systems since each server can respond to only one HPSS System Manager.

ACLs on HPSS CDS Objects may need to be added for appropriate foreign_user and/or foreign_group entries.

CDS paths may need to be modified between servers to specify full paths (/…/cellname/blah) rather than cell-relative paths (/.:/blah).

All servers in a specific HPSS MUST specify the same serverconfig file.

Separate rc.hpss (rc.hpss_othercell) and hpss_env files (hpss_env_othercell) should be specified on the remote host.

Log Clients on remote cells need to specify a unique HPSSLOG_SHMKEY in the hpss_env_othercell file. Failure to do this will result in inconsistent messages going to the Log Daemon (Messages for servers in the remote HPSS may appear to the wrong Log Daemon, local logs, etc.)

2.3 Prerequisite Software Considerations

This section defines the prerequisite software requirements for HPSS. These software products must be purchased separately from HPSS and installed prior to the HPSS installation and configuration.

2.3.1 Overview

A summary of the products required to run HPSS is described in the following subsections.

2.3.1.1 DCE

HPSS uses the Distributed Computing Environment (DCE) and operates within a DCE cell. HPSS requires at least one (1) *DCE Security Server* and at least one (1) *DCE Cell Directory Server.*

An additional Security Server can be added into the configuration on a different machine to provide redundancy and load balancing, if desired. The same is true for the Cell Directory Server.

DCE base (or client) software services are required on the HPSS client machines (except for FTP, PFTP, NFS client machines).

For U.S. sites, assuming some level of encryption is desired for secure DCE communication, the *DCE Data Encryption Standard (DES) library routines* are required. For non-U.S. sites or sites desiring to use non-DES encryption, the *DCE User Data Masking Encryption Facility* is required. Note that if either of these products are ordered, install them on all nodes containing any subset of DCE and / or Encina software.

If the DCE cell used for HPSS is expected to communicate with other DCE cells, the System Administrator should review the DCE documentation and determine if the *DCE Global Directory Service* and *DCE Global Directory Client* will be needed.

For AIX 4.2.1 environments, customers must acquire DCE Version 2.1 for machines running HPSS servers. In addition, customers must acquire the *Getting Started with DCE for Application Developers, Version 2.1* for each machine that will either compile / link HPSS software or compile / link applications that use the HPSS client API library. If running DFS is desired, customer also must acquire DCE Version 2.2 for machines running DFS HDM servers.

For AIX 4.3.2 environments, customer must acquire DCE Version 2.2 for running both HPSS servers and DFS HDM servers.

Specific DCE product information regarding individual version of operating systems can be found later in this section.

2.3.1.2 Encina

TXSeries 4.2 can now be run with DCE Version 2.2.

HPSS uses the Encina distributed transaction processing software developed by Transarc Corporation, including the Encina Structured File Server (SFS) to manage all HPSS metadata. HPSS Release 4.1.1 must be run with TX-Series 4.2 version. Sites with earlier versions of Encina must upgrade their software before running with HPSS 4.1.1.

The *Encina Base* software consists of the following three components:

- Encina Server
- Encina Client
- Encina Structured File Server

The following nodes must have, at a minimum, the *Encina Base* software installed:

- Nodes that run Encina SFS

- Nodes that run one or more HPSS servers (with the exception of nodes that run only the non-DCE/Encina HPSS Mover)

- Nodes that run an end-user client application that links with the HPSS Client API library

A node that only performs an "**ftp**" to another machine to get into HPSS does not require any DCE client or Encina client software license. Also, a node that does an NFS mount to HPSS do not require DCE and Encina client software. If DCE Kerberos authentication is desired over NFS, DCE client software is required.

Encina cannot run on nodes that run DFS HDM servers due to the DCE version 2.2 requirement.

The main I/O from Encina SFS is in the transaction log, the actual SFS data files, and media log archiving. When deciding on the size and number of disks for metadata, keep in mind the following:

1. At a minimum, the transaction log and SFS data files should be on different disks.

2. Ideally, several physical disks should be available for SFS data files (NS data can be on one, BFS data on another, etc.).

3. The media log archiving is simply a running backup of the transaction log. The transaction log is wrap-around, but the media log archives are not. A large amount of data is generated by these archives and must routinely be backed up and removed. A separate disk should be devoted to the log archives, if possible.

For reliability and availability reasons, the transaction log should be mirrored. The SFS data volumes should also be mirrored if possible. For performance and reliability reasons, mirroring should be done using separate physical devices (as opposed to the AIX logical volume being mirrored to the same physical drive).

Section 2.11 which contains guidelines for proper maintenance of the SFS metadata. The policies described should be fully understood and implemented to protect the HPSS metadata. Failure to follow these policies can lead to unrecoverable data loss.

2.3.1.3 Sammi

HPSS uses Sammi, a graphical user environment product developed by the Kinesix Corporation, to implement and provide the SSM graphical user interface. To be able to use SSM to configure, control, and monitor HPSS, one or more Sammi licenses must be purchased from Kinesix. The number of licenses needed depends on the site's anticipated number of SSM users who may be logged onto SSM concurrently.

For the current release of HPSS, the HPSS Server Sammi License (Part Number 01-0002100-A) and, optionally, the HPSS Client Sammi License (Part Number 01-0002200-B) available from the Kinesix Corporation are required. The Sammi software must be installed separately prior to the HPSS installation. In addition, the Sammi license(s) for the above components must be obtained from Kinesix and set up as described in Section 3.4.3 before running Sammi.

In addition to the purchase of the Sammi licenses, the following system resources are required to support Sammi:

1. Operating System and System Software Requirements:
    - AIX 4.2.1 or AIX 4.3.2
    - TCP/IP and Sun NFS/RPC
    - X Window System (X11 Release 5) and Motif
2. System Space Requirements:
    - 32 MB of memory
    - 40 MB of swap space
3. Hardware Requirements:
    - A graphic adapter with at least 256 colors
    - A monitor with a suggested resolution of 1280 x 1024 pixels
    - A mouse (two or three buttons)

2.3.1.4 Miscellaneous

- Sites needing to compile the HPSS FTP source code must have the **yacc** compiler.
- Sites needing to compile the HPSS NFS Daemon source code must have the AIX NFS client software.
- Sites needing to compile the HPSS Generic Security Service (GSS) security code must have the X.500 component of DCE.
- Sites needing to compile the HDM code, build the cs/xdr/dmapi libraries, or run the SFS Backup scripts must have the **perl** compiler installed (**perl 5.003** or later).

2.3.2 Prerequisite Summary for HPSS Server/Mover Machine - AIX

HPSS Release 4.1.1 supports two configuration scenarios based on the preference of AIX 4.2.1 or AIX 4.3.2. The prerequisite for these configurations are described in Section 2.3.2.1 and Section 2.3.2.2.

2.3.2.1 AIX 4.2.1

1. AIX 4.2.1 (IBM Program Number 5765-C34).
2. IBM DCE for AIX Version 2.1 (PTF Set 27 or later) must be installed on machine(s) where HPSS servers are to be run. The following DCE components are required:
    - IBM Directory and Security Server for AIX, Version 4 (IBM Program Number 5765-639), or equivalent.

- Getting Started with DCE for Application Developers, Version 2.1 (IBM Program Number 5765-532). Required on all nodes compiling or linking any HPSS software.

- DCE Data Encryption Standard Version 2.1 (IBM Program Number 5765-D13), used if data encryption is desired.

3. If DFS is desired, a separate machine for running the DFS HDM servers is required. This machine must be running with DCE for AIX Version 2.2 (IBM Program Number 5765-D14, PTF Set 5 or later)

4. TXSeries 4.2 for AIX (Transarc Product Number ENC-MMK-V25AIX-1, patch level 8 or later).

5. HPSS Server Sammi License (Part Number 01-0002100-A) from Kinesix Corporation for each HPSS license which usually consists of a production and a test system. In addition, an HPSS Client Sammi License (Part Number 01-0002200-B) is required for each additional, concurrent SSM user. Refer to Section 2.3.1.3 for more information on Sammi prerequisite and installation requirements. This is only needed if Sammi is to be run on the machine.

6. High Performance Parallel Interface Drivers Group (HiPPI/6000) (IBM Program Number 5765-551), if HiPPI is required.

2.3.2.2 AIX 4.3.2

1. AIX 4.3.2 (IBM Program Number 5765-C34)

2. DCE for AIX Version 2.2 (IBM Program Number 5765-D14, PTF Set 5 or later). HPSS servers and DFS HDM servers can be run on the same machine or on separate machines, as desired.

3. TXSeries 4.2 for AIX (Transarc Product Number ENC-MMK-V25AIX-1, patch level 8 or later).

4. HPSS Server Sammi License (Part Number 01-0002100-A) from Kinesix Corporation for each HPSS license which usually consists of a production and a test system. In addition, an HPSS Client Sammi License (Part Number 01-0002200-B) is required for each additional, concurrent SSM user. Refer to Section 2.3.1.3 for more information on Sammi prerequisite and installation requirements. This is only needed if Sammi is to be run on the machine.

5. High Performance Parallel Interface Drivers Group (HiPPI/6000) (IBM Program Number 5765-551), if HiPPI is required.

2.3.3 Prerequisite Summary for HPSS Mover Machine - IRIX

1. IRIX 6.4.1 (SGI Program Number SC4-IRIX-6.4.1, with latest/recommended patch set).

2. HiPPI drivers (SGI Program Number SC4-XHIPOR-3.1), if HiPPI network support is required.

2.3.4 Prerequisite Summary for HPSS Mover/Client API Machine - Solaris

1. Solaris 2.6.
2. DCE for Solaris 2.0, unless non-DCE Mover only machine.
3. Encina for Solaris 4.2, unless non-DCE Mover only machine.
4. HiPPI drivers, if HiPPI network support is required.

2.3.5 Prerequisite Summary for Non-DCE Client Machine

1. AIX 4.2.1 or AIX 4.3.2 (IBM Program Number 5765-C34), IRIX 6.4 or Solaris 2.6.

2.4 Hardware Considerations

This section describes the hardware infrastructure needed to operate HPSS and considerations about infrastructure installation and operation that may impact HPSS.

2.4.1 Network Considerations

Because of its distributed nature and high-performance requirements, an HPSS system is highly dependent on the networks providing the connectivity among the HPSS servers, SFS servers, and HPSS clients.

For control communications (i.e., all communications except the actual transfer of data) among the HPSS server and HPSS clients, HPSS supports networks that provide TCP/IP. Since control requests and replies are relatively small in size, a low-latency network usually is well suited to handling the control path.

The data path is logically separate from the control path and may also be physically separate (although this is not required). For the data path, HPSS supports the same TCP/IP networks as those supported for the control path, as well as IPI-3 over HiPPI. For supporting large data transfers, the latency of the network is less important than the overall data throughput.

HPSS also supports a special data path option that may indirectly affect network planning because they may off-load or shift some of the networking load. This option uses shared memory data transfer method, which provides for intra-machine transfers between either Movers or Movers and HPSS clients directly via a shared memory segment.

The DCE RPC mechanism used for HPSS control communications can be configured to utilize a combination of TCP/IP and User Datagram Protocol (UDP)/IP (i.e., one of the two protocols or both of the protocols). However, during the development and testing of HPSS, it was discovered that using TCP/IP would result in increasing and unbounded memory utilization in the servers over time (which would eventually cause servers to terminate when system memory and paging space resources were exhausted). Because of this behavior when using TCP/IP for the DCE RPC mechanism, the HPSS servers should only utilize UDP/IP for control communications. The default HPSS installation/configuration process will enable only UDP/IP for DCE RPC communications

with the HPSS servers. See Section 4.3 for further details (specifically the environment variable **RPC_SUPPORTED_PROTSEQS**).

The DCE RPC mechanism, by default, will use all available network interfaces on nodes that have multiple networks attached. In cases where one or more nodes in the DCE cell is attached to multiple networks, it is required that each node in the DCE cell be able to resolve any other network address via the local IP network routing table. The environment variable **RPC_UNSUPPORTED_NETIFS** may be used to direct DCE to ignore certain network interfaces, especially if those interfaces are not accessible from other nodes in the cell. For example, to instruct DCE to ignore a local HiPPI interface as well as a second ethernet interface, **RPC_UNSUPPORTED_NETIFS** could be set to "**`hp0:en1`**". Note that this must be done prior to configuring DCE. If used, this environment variable should be set system wide by placing it in **`/etc/environment`** on the local machine.

2.4.2 Tape Robots

All HPSS PVRs are capable of sharing a robot with other tape management systems but care must be taken when allocating drives among multiple robot users. If it is necessary to share a drive between HPSS and another tape management system, the drive can be configured in the HPSS PVR but left in the LOCKED state until it is needed. When needed by HPSS, the drive should be set to UNLOCKED by the HPSS PVR and should not be used by any other tape management system while in this state. This is critical because HPSS periodically polls all of its unlocked drives even if they are not currently mounted or in use.

Generally, only one HPSS PVR is required per robot. However, it is possible for multiple PVRs to manage a single robot in order to provide drive and tape pools within a robot. The drives in the robot must be partitioned among the PVRs and no drive should be configured in more than one PVR. Each tape is assigned to exactly one PVR when it is imported into the HPSS system and will only be mounted in drives managed by that PVR.

2.4.2.1 STK

The STK PVR must be able to communicate with STK's ACSLS server. HPSS supports ACSLS versions 4 and 5. For the PVR to communicate with the ACSLS server, it must have a TCP/IP connection to the server (e.g. Ethernet) and STK's SSI software must be running on the machine with the PVR. Multiple STK Silos can be connected via pass through ports and managed by a single ACSLS server. This collection of robots can be managed by a single HPSS PVR.

2.4.2.2 IBM 3494/3495

The 3494/3495 PVR supports BMUX, Ethernet, and RS-232 (TTY) attached robots. If appropriately configured, multiple robots can be accessible from a single machine.

2.4.2.3 ADIC AML

The Distributed AML Server (DAS) client components on the AIX workstations must be able to communicate (via a TCP/IP connected network) with DAS Client components on the machine controlling the robot in order to request DAS services.

2.4.2.4 Operator Mounted Drives

An Operator PVR is used to manage a homogeneous set of manually mounted drives. Tape mount requests will be displayed on an SSM screen.

2.4.3 Disk Devices

HPSS supports both locally attached devices connected to a single node via a private channel and HiPPI attached disk devices.

Locally attached disk devices that are supported include those devices attached via either SCSI, SSA or Fibre Channel. For these devices, operating system disk partitions of the desired size must be created (e.g., AIX logical volumes), and the raw logical volume device name must be used when creating the Mover Device configuration (see Section 5.7 for details on configuring storage devices).

AIX logical volumes can be greater than 2 GB.

HiPPI attached disk devices that are supported are the Maximum Strategies Inc. and IBM 9570 disk arrays. These devices provide third-party data transfer capabilities (i.e., data flows directly between the device and the client nodes, without passing through the Mover) via IPI-3 over HiPPI. The device definition is contained in IPI-3 configuration files, which are described in Section 5.8.5.

2.4.4 Tape Devices

The tape devices / drives supported by HPSS are listed below, along with the supported device host attachment methods and robotics supported for each device.

- IBM 3480 and 3490 devices are supported via a BMUX attachment, IBM 3490E devices are supported via either a BMUX or SCSI attachment, and IBM 3590 devices are supported via a SCSI attachment. All the devices are supported in the IBM 3494 / 3495 tape libraries. 3590 devices are also supported in the StorageTek 4400 and ADIC AML tape libraries.

- StorageTek 4480, 4490, 9840, Redwood, and Timberline devices are supported via a SCSI attachment and are supported in the StorageTek 4400 tape libraries. IBM 3590 devices are also supported when located in the StorageTek 4400 tape libraries.

- Ampex DST-312 and DST-314 devices are supported via SCSI attachment and are supported in the ADIC AML tape libraries.

2.4.4.1 Multiple Media Support

Starting with the 4.1.1.1 patch, HPSS supports multiple types of media for certain drives. Listed in the following table is a preference list for each media type that can be mounted on more than one drive type. When the PVL starts, it determines the drive type that each type of media may be mounted on. It makes these decisions by traversing each media type's list and using the first drive type from the list that it finds configured in the system. So, looking at the table, it can be determined that a single-length 3590E tape will mount on a double-length 3590E drive if and only if there are no single-length 3590E drives configured in the system.

Table 2-1

Cartridge Type	Drive Preference List
AMPEX DST-312	AMPEX DST-312 AMPEX DST-314
AMPEX DST-314	AMPEX DST-314
Single-Length 3590	Single-Length 3590 Double-Length 3590 Single-Length 3590E Double-Length 3590E
Double-Length 3590	Double-Length 3590 Double-Length 3590E
Single-Length 3590E	Single-Length 3590E Double-Length 3590E
Double-Length 3590E	Double-Length 3590E

Note that the PVL's choices are made at startup time, and are not made on a mount-to-mount basis. Therefore a single-length 3590E cartridge will never mount on a double-length 3590E drive if a single-length 3590E drive was configured in the system when the PVL was started.

2.5 HPSS Interface Considerations

This section describes the user interfaces to HPSS and the various considerations that may impact the use and operation of HPSS.

2.5.1 Client API

The HPSS Client API provides a set of routines that allow clients to access the functions offered by HPSS. The API consists of a set of calls that are comparable to the file input/output interfaces defined by the POSIX standard (specifically ISO/IEC 9945-1:1990 or IEEE Standard 1003.1-1990), as well as extensions provided to allow access to the extended capabilities offered by HPSS.

The Client API is built on top of DCE and Encina (which provide threading, transactional RPCs, and security) and must be run on a platform capable of supporting DCE and Encina client programs. To access HPSS from client platforms that do not support DCE and Encina clients, the FTP, Parallel FTP, NFS, and Non-DCE Client API interfaces can be used.

The Client API allows clients to specify the amount of data to be transferred with each request. The amount requested can have a considerable impact on system performance and the amount of metadata generated when writing directly to a tape storage class. See Sections 2.7.6 and 2.10 for further information.

The details of the Application Programming Interface are described in the *HPSS Programmer's Reference Guide.*

2.5.2 Non-DCE Client API

The Non-DCE Client API provides the same user functional calls as the Client API for client applications who are running on platforms without DCE or Encina with the following exceptions: ACL calls are not supported, calls are not transactional, and no security is currently implemented. Security is planned for Release 4.2. In order to use the NDAPI, the client application must link the Non-DCE Client API library, and the target HPSS system must have a Non-DCE Client Gateway server configured and running. The application's calls to the library are then sent over the network to the NDCG who executes the appropriate Client API calls, returning the results to the client application.

2.5.3 FTP

HPSS provides an FTP server that supports standard FTP clients. Extensions are also provided to allow additional features of HPSS to be utilized and queried. Extensions are provided for specifying Class of Service to be used for newly created files, as well as directory listing options to display Class of Service and Accounting Code information. In addition, the **chgrp**, **chmod**, and **chown** commands are supported as **quote site** options.

The FTP server is built on top of the Client API and must be run on a machine that supports DCE and Encina clients. Note that FTP clients can run on computers that do not have DCE and Encina installed.

The configuration of the FTP server allows the size of the buffer to be used for reading and writing HPSS files to be specified. The buffer size selected can have a considerable impact on both system performance and the amount of metadata generated when writing directly to a tape Storage Class. See Sections 2.7.6 and 2.10 for further information.

The GSSFTP from MIT is supported if the appropriate HPSS FTP Daemon and related processes are implemented. This client provides credential-based authentication and "Cross Cell" authentication to enhance security and "password-less" FTP features.

Refer to the *HPSS User's Guide* for details of the FTP interface.

2.5.4 Parallel FTP

The FTP server also supports the new HPSS Parallel FTP (PFTP) protocol, which allows the PFTP client to utilize the HPSS parallel data transfer mechanisms. This provides the capability for the client to transfer data directly to the HPSS Movers (i.e., bypassing the FTP Daemon), as well as the capability to stripe data across multiple client data ports (and potentially client nodes). Data transfers are supported via either TCP/IP or IPI-3 over HiPPI. (If the IPI-3 is to be supported, the PFTP client must include the IPI-3 specific option.) Support is also provided for performing partial file transfers.

The PFTP protocol is supported by the HPSS FTP Daemon. Refer to Section 5.8.3 for configuration information. No additional configuration of the FTP Daemon is required to support PFTP clients.

The client side executables for PFTP are **pftp_client** and **pftp_client_ipi3**. **pftp_client** supports only TCP based transfers. **pftp_client_ipi3** supports both TCP transfers and IPI-3 transfers over HiPPI. Because both client executables are supersets of standard FTP, standard FTP requests can be issued as well as the PFTP extensions.

The "**krb5_gss_pftp_client**" and MIT GSSFTP clients are supported by the **hpss_pftpd_amgr** and the **auth_krb5gss** Authentication Manager. These clients provide credential-based authentication and "Cross Cell" authentication for enhanced security and "password-less" FTP features.

Refer to the *HPSS User's Guide* for details of the PFTP interface.

2.5.5 NFS

The HPSS NFS interface implements the Network File System (NFS) Version 2 Protocol for access to HPSS name space objects and bitfile data. The NFS protocol was developed by Sun Microsystems to provide transparent remote access to shared file systems over local area networks. Because the NFS designers wanted a robust protocol that was easy to port, NFS is implemented as a stateless protocol. This allows use of a connectionless networking transport protocol (UDP) that requires much less overhead than the more robust TCP. As a result, client systems must time out requests to servers and retry requests that have timed out before a response is received. Client time-out values and retransmission limits are specified when a remote file system is mounted on the client system.

The two main advantages of using NFS instead of a utility like FTP are (1) files can be accessed and managed through standard system mechanisms without calling a special program or library to translate commands, and (2) problems associated with producing multiple copies of files can be eliminated because files can remain on the NFS server. The primary disadvantages of NFS are the 2 GB file size limitations of the Version 2 protocol, the fact that UDP does not provide data integrity capabilities, and the data transfer performance due to the limitation of sending data via the RPC mechanism. In general, NFS should not be the interface of choice for large HPSS data transfers. NFS is recommended for enabling functionality not provided through other interfaces available to the client system.

Because of the distributed nature of HPSS and the potential for data being stored on tertiary storage, the time required to complete an NFS request may be greater than the time required for non-HPSS NFS servers. The HPSS NFS server implements caching mechanisms to minimize these delays, but time-out values (**timeo** option) and retransmission limits (**retrans** option) should be adjusted accordingly. A time-out value of no less than 10 and a transmission limit of no less than 3 (the default) are recommended.

Refer to the *HPSS User's Guide* for details of the NFS interface.

2.5.6 MPI-IO API

The HPSS MPI-IO API provides access to the HPSS file system through the interfaces defined by the MPI-2 standard (*MPI-2: Extensions to the Message-Passing Interface*, July, 1997).

The MPI-IO API is layered on top of a host MPI library. The characteristics of a specific host MPI are designated through the **include/mpio_MPI_config.h**, which is selected from one of the set of supported host MPIs when HPSS is configured, by settings in the **Makefile.macros**. The HPSS system administrator should choose the appropriate configuration to use when building the MPI-IO subsystem. We currently support MPI-IO API for the following host MPIs:

IBM's Parallel Environment MPI, versions 2.4.0.8 and 2.3.0.6

Sun HPC MPI, version 4.0

ANL MPICH, version 1.1

Configuration files for other versions of MPI may be created by modifying one of the existing configuration files. Each configuration file enables or disables use of the host MPI implementation of other MPI-2 features on which the HPSS MPI-IO API depends. For MPI-1 implementations (such as MPICH 1.1), no host MPI-2 features are available for use. As MPI vendors' implementations provide more MPI-2 features, HPSS MPI-IO will switch from using its own versions to using the host MPI.

The host MPI library must support multithreading. Specifically, it must permit multiple threads within a process to issue MPI calls concurrently, subject to the limitations described in the MPI-2 standard.

The threads used by MPI-IO must be DCE-compatible threads. Threaded applications must be loaded with the appropriate threads libraries.

Files read and written through the HPSS MPI-IO can also be accessed through the HPSS Client API, FTP, Parallel FTP, or NFS interfaces. So even though the MPI-IO subsystem does not offer all the migration, purging, and caching operations that are available in HPSS, parallel applications can still do these tasks through the HPSS Client API or other HPSS interfaces.

The details of the MPI-IO API are described in the *HPSS Programmer's Reference Guide*, Volume 1, Release 4.1.

2.5.7 DFS

DFS is offered by the Open Software Foundation (now the Open Group) as part of DCE. DFS is a distributed file system that allows users to access files using normal Unix utilities and system calls, regardless of the files location. This transparency is one of the major attractions of DFS. The advantage of DFS over NFS is that it provides greater security and allows files to be shared globally between many sites using a common name space.

HPSS provides two options for controlling how DFS files are saved to HPSS: archived and mirrored. The archived option gives users the impression of having an infinitely large DFS file system that performs an near-native DFS speeds. This option is well suited to storing large numbers of small files. However, files stored to HPSS using this option can only be accessed through DFS interfaces, and cannot be accessed with HPSS utilities, such as parallel FTP. Therefore, the performance for data transfers is limited to DFS speeds.

The mirrored option gives users the impression of having a single, common (mirrored) name space where objects have the same path names in DFS and HPSS. With this option, large files can be stored quickly on HPSS, then analyzed at a more leisurely pace from DFS. On the other hand, some operations, such as file creates, perform slower when this option is used, as compared to when the archived option is used.

HPSS and DFS define disk partitions differently from one another. In HPSS, the option for how files are stored is associated with a fileset. Recall that in DFS, multiple filesets may reside on an aggregate, therefore, in DFS the option applies to all filesets on a given aggregate.

To use the DFS / HPSS interface on an aggregate, the aggregate must be on a processor that has Transarc's DFS SMT kernel extensions installed. These extensions are available for Sun Solaris and IBM AIX platforms. Once an aggregate has been set up, end users can access filesets on the aggregate from any machine that supports DFS client software, including PCs. The wait / retry logic

in DFS client software was modified to account for delays when data is staged from HPSS. If these changes have not been incorporated into the DFS client software, occasional long delays for IO requests may occur.

HPSS servers and DFS both use Encina as part of their infrastructure. Since the DFS and HPSS release cycles to support the latest version of Encina may differ significantly, running the DFS server on a different machine from the HPSS servers is recommended.

Refer to the *HPSS User's Guide* for details of the DFS interface.

2.6 HPSS Server Considerations

Servers are the internal components of HPSS. They must be configured correctly to ensure that HPSS operates properly. This sections describes key concepts and notions of the various servers and their impact on system use, operation, and performance.

2.6.1 Name Server

The HPSS Name Server (NS) maintains a data base in five Encina SFS files. An SFS relative sequenced file is used to store data associated with NS objects. (NS objects are bitfiles, directories, symbolic links, junctions and hard links.) The four other files are SFS clustered files. Two of these files store text data and ACL entries, and the remaining two files are SFS clustered files that are used to store fileset information.

The total number of objects permitted in the name space is limited by the number of SFS records allocated to the NS. Refer to Section 2.9.2.2 for details on selecting the size of the name space. With this release of HPSS, no provisions have been made for increasing the size of the name space by adding additional SFS files or additional Name Servers. However, if more name space is needed, additional space can be obtained by allocating more SFS records. Refer to Section 8.7.3 for information on handling an NS space shortage.

The NS uses DCE threads to service concurrent requests. Refer to Section 5.3.1 for more information on selecting an appropriate number of DCE threads. The NS accepts requests from any client that is authenticated through DCE; however, certain NS functions can be performed only if the request is from a trusted client. Trusted clients are those clients for whom control permission has been set in their CDS ACL entry for the NS. Higher levels of trust are granted to clients who have both control and write permission set in their CDS ACL entry. Refer to Table 5-1 for information concerning the CDS ACL for the Name Server.

For listing directory entries, the number of bytes that can be returned is bounded in two ways: a client specifies the maximum number of bytes that it can accept, and, in addition, the NS has an upper limit on the number of bytes it will return. The upper limit for the NS must be carefully selected. A size that is too small will degrade performance. Refer to Section 2.10 for information on selecting an appropriate size.

The NS can be configured to allow or disallow super-user privileges (root access). When the NS is configured to allow root access, the UID of the super-user is configurable.

In the current HPSS release, only one NS is supported.

2.6.2 Bitfile Server

The Bitfile Server (BFS) provides a view of HPSS as a collection of files. It provides access to these files and maps the logical file storage into underlying storage objects in the Storage Servers.

When a BFS is configured, it is assigned a server ID. This value should never be changed. It is embedded in the identifier that is used to name bitfiles in the BFS. Future releases of HPSS that will allow multiple BFS will use this value to link a bitfile to the BFS that will manage it.

The BFS maps bitfiles to their underlying physical storage by maintaining mapping information that ties a bitfile to the storage server storage segments that contain its data. These storage segments are referenced using an identifier that contains the server ID of the Storage Server that manages the storage segment. For this reason, once a Storage Server has been assigned a server ID, this ID must never change. For additional explanation on this point, see Sections 2.6.3 and 2.6.4.

In the current HPSS release, only one BFS will be supported.

The relationship of BFS bitfiles to SS storage segments and other structures is shown in Figure 2-1.

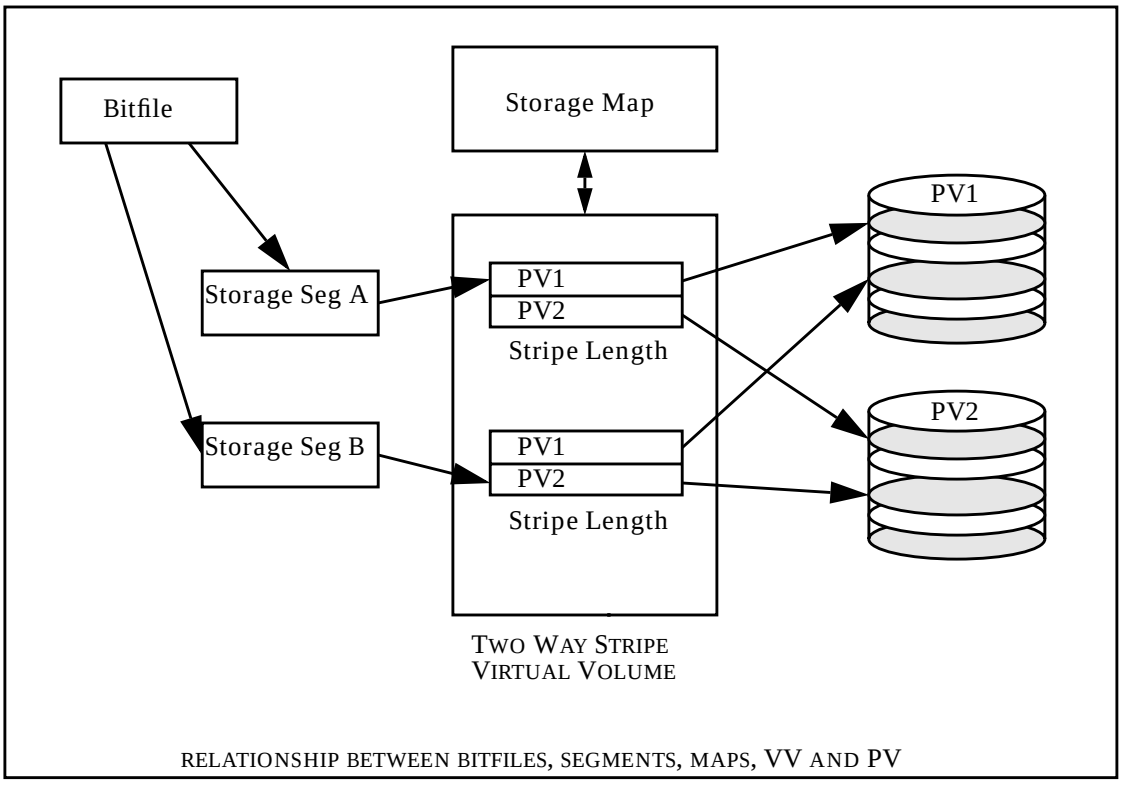


Figure 2-1 The Relationship of Various Server Data Structures

2.6.3 Disk Storage Server

Each Disk Storage Server manages random access magnetic disk storage units for HPSS. It maps each disk storage unit onto an HPSS disk Physical Volume (PV) and records configuration data for the PV. Groups of one or more PVs (disk stripe groups) are managed by the server as disk Virtual Volumes (VVs). The server also maintains a storage map for each VV that describes which portions of the VV are in use and which are free. Figure 2-1 shows the relationship of SS data structures such as VVs to other server data structures.

Each Disk Storage Server must have its own set of metadata files (storage map, storage segment, VV, and PV) in SFS. Disk Storage Servers may not share metadata files among themselves.

Once a Disk Storage Server is established in the system and a server ID is selected, the server ID must never be changed. The BFS uses the server ID, which can be found inside storage segment IDs, to identify which Disk Storage Server provides service for any given disk storage segment. If the server ID is changed, disk storage segments provided by that server will be unreachable. A Disk Storage Server ID can be changed only if all of the server's storage segments have been removed from the system.

The server can manage information for any number of disk PVs and VVs; however, because a copy of all of the PV, VV, and storage map information is kept in memory at all times while the server runs, the size of the server will be proportional to the number of disks it manages.

The Disk Storage Server is designed to scale up its ability to manage disks as the number of disks increases. As long as sufficient memory and CPU capacity exist, threads can be added to the server to increase its throughput. If available memory and CPU capacity runs short, new Disk Storage Servers will need to be added to the system. A new Disk Storage Server can be added to a system by selecting a group of disks for it to manage, emptying them, and then deleting them from their original server. A new server can then be added to the system and the disks recreated in the new server.

2.6.4 Tape Storage Server

Each Tape Storage Server manages serial access magnetic tape storage units for HPSS. The server maps each tape storage unit onto an HPSS tape PV and records configuration data for the PV. Groups of one or more PVs (tape stripe groups) are managed by the server as tape VVs. The server maintains a storage map for each VV that describes how much of each tape VV has been written and which storage segment, if any, is currently writable in the VV. Figure 2-1 shows the relationship of SS data structures such as VVs to other server data structures.

Each Tape Storage Server must have its own set of metadata files (storage map, storage segment, VV, and PV) in SFS. Tape Storage Servers may not share metadata files among themselves.

Once a Tape Storage Server is established in the system and a server ID is selected, the server ID must never be changed. The BFS uses the server ID, which can be found inside storage segment IDs, to identify which Tape Storage Server provides service for any given tape storage segment. If the server ID is changed, tape storage segments provided by that server will be unreachable. A Tape Storage Server ID can be changed if all of the server's storage segments have been removed from the system, but this is a time-consuming task because it requires migrating all the server's tape segments to another Storage Server.

The server can manage information for any number of tape PVs and VVs. The Tape Storage Server can manage an unlimited number of tape PVs, VVs, maps, and segments without impacting its size in memory.

The Tape Storage Server is designed to scale up its ability to manage tapes as the number of tapes increases. As long as sufficient memory and CPU capacity exist, threads can be added to the server to increase its throughput. If available memory and CPU capacity runs short, new Tape Storage Servers will need to be added to the system. Note that the number of tape units the server manages has much more to do with the throughput of the server than the number of tapes the server manages. If the number of tape units in the system increases, adding a new Tape Storage Server to the system may be the best way to deal with the increased load. It should be noted that moving tape volumes from one Tape Storage Server to another may not be as easy as between Disk Storage Servers because of the sequentiality of the tape and the amount of data stored on each tape volume.

2.6.5 Migration/Purge Server

The Migration / Purge Server (MPS) migrates (moves data down the storage hierarchy) and purges (removes data from a higher level in the hierarchy once it has been migrated downward) data within a storage hierarchy based on the migration and purge policies that have been defined. In addition, for the disk type storage classes, the MPS provides the capability to make duplicate copies of bitfiles if this option has been selected.

During migration runs, MPS will take checkpoints of metadata restart information. MPS views the metadata as a sequential file. During normal runs, MPS will stop a run after it has met the defined criteria of the amount of data to migrate. When the next migration run begins, it starts migrating at the metadata location in the checkpoint. This has the effect of distributing the data being migrated more evenly over the storage class. If there is a failure and MPS has to be restarted, it will restart at the metadata location in the last checkpoint. Without the checkpoint it would be necessary to start completely over, which would waste time.

The MPS manages only those storage classes that have an MPS ID specified in their configuration entry. The storage classes at the bottom of a storage hierarchy, which do not support migration or purge, should still be configured with an MPS ID for space usage reporting. However, these storage classes must not be configured with a migration or a purge policy.

MPS uses threads to do the migration / purge and to gather the storage class statistics from Storage Servers. The appropriate number of DCE threads to be specified in the general server configuration entry can be calculated as given in Section 5.3.1.

MPS provides the capability to generate migration / purge report files that document the activities of the server. The specification of the UNIX report file in the MPS specific configuration enables the server to create the report files. Once reporting is enabled, a report is generated every 24 hours. The file name of the report files contains a UNIX file name prefix from the configuration entry, plus a year-month-day suffix. These report files can be interpreted by the utility **mps_reporter**. Since the report files grow rapidly, each site should develop a **cron** job that will periodically remove the reports that are no longer needed.

Because the MPS uses a BFS and several Storage Servers to perform the data movement between hierarchy levels, the BFS and the Storage Servers must be running in order for the MPS to perform its functions. In addition, the MPS obtains the space usage statistics from the Storage Servers. The server IDs of the BFS and Storage Servers must be specified in the MPS specific configuration entry.

It is possible to have multiple MPSs in operation at the same time. To do this, it is necessary to divide the storage classes into groups that are then assigned to the several MPSs. Configuring multiple MPSs is done the same way as configuring a single MPS except that different storage classes are assigned to the various MPSs. Multiple MPSs may be needed if migration runs are taking so long that a sufficient amount of data cannot be migrated by one server.

2.6.6 Location Server

All HPSS client API applications, which includes all end user applications, will need to contact the Location Server at least once during initialization and usually later during execution in order to locate the appropriate servers to contact. If the Location Server is down for an extended length of time, these applications will eventually give up on retrying their requests and become non-operational. To avoid letting the Location Server become a single point of failure, consider replicating it, preferably on a different machine. If replicating the Location Server is not an option or desirable, consider increasing the automatic restart count for failed servers in SSM. Since the Location Server's requests are short lived, and each client contacts it through a cache, performance alone is not usually a reason to replicate the Location Server. Generally the only time a Location Server should be replicated solely for performance reasons is if it is reporting heavy load conditions to SSM.

If any server is down for an extended length of time it is important to mark the server as non-executable within SSM. As long as a server is marked executable the Location Server continues to advertise its location to clients which may try to contact it. For example, in the case of a Bitfile Server, the Location Server will continue to try to contact it periodically to gather storage statistics information.

If a Class of Service configuration is modified, the BFS must be recycled. In addition, the Location Server must be reinitialized to keep the Class of Service metadata synchronized. Otherwise, clients may have problem selecting a Class of Service during file creations.

The Location Server must also be reinitialized or recycled whenever the Location Policy or its server configuration is modified, Note that it is not necessary to recycle the Location Server if an HPSS server's configuration is added, modified, or removed since this information is periodically reread.

2.6.7 PVL

The PVL is responsible for mounting and dismounting PVs (such as tape and magnetic disk) and queuing mount requests when required drives and media are in use. The PVL usually receives requests from Storage Server clients. The PVL accomplishes any physical movement of media that might be necessary by making requests to the appropriate Physical Volume Repository (PVR). The PVL communicates directly with HPSS Movers in order to verify media labels.

The PVL is not required to be co-resident with any other HPSS servers and is not a CPU-intensive server. With its primary duties being queuing, managing requests, and association of physical volumes with PVRs, the PVL should not add appreciable load to the system.

In the current HPSS release, only one PVL will be supported.

2.6.8 PVR

The PVL manages a set of imported cartridges, mounts and dismounts them when requested by the PVL. It is possible for multiple HPSS PVRs to manage a single robot. This is done if it is necessary to partition the tape drives in the robot into pools. Each tape drive in the robot is assigned to exactly one PVR. The PVRs can be configured identically and can communicate with the robot through the same interface.

The following sections describe the considerations for the various types of PVRs supported by HPSS.

2.6.8.1 STK PVR

The STK PVR communicates to the ACSLS server via STK's SSI software.

Configuration instructions for the SSI can be found in the *STK Automated Cartridge System Library Software (ACSLS) System Administrators Guide.*

The SSI must be started before the HPSS PVR. If the SSI is started after the PVR, the PVR should be stopped and restarted.

If multiple STK robots are managed, SSIs that communicates with each of the robots should be configured on separate CPUs. A PVR can be configured on each of the CPUs that is running an SSI. If multiple STK robots are connected and are controlled by a single Library Management Unit (LMU), a single PVR can manage the collection of robots. The PVR can be configured on any CPU that is running an SSI.

2.6.8.2 3494/3495 PVR

The 3494/3495 PVR can manage any IBM tape robot—3494 or 3495, BMUX, Ethernet, or TTY attached. The PVR will create a process to receive asynchronous notifications from the robot.

At least one PVR should be created for every robot managed by HPSS. If multiple 3494/3495 robots are managed, the PVRs must be configured to communicate with the correct **/dev/lmcp** device. The PVRs can run on the same CPU or different CPUs as long as the proper **/dev/lmcp** devices are available.

2.6.8.3 AML PVR

The AML PVR can manage ADIC AML robots that use Distributed AML Server (DAS) software. The DAS AML Client Interface (ACI) operates synchronously, that is, once a request is made to the AML, the request process does not regain control until the operation has completed or terminated. Therefore, the AML PVR must create a process for each service request sent to the DAS (such as mount, dismount, eject a tape, etc.).

2.6.8.4 Operator PVR

The Operator PVR simply displays mount requests for manually mounted drives. The mount requests are displayed on the appropriate SSM screen

All of the drives in a single Operator PVR must be of the same type. Multiple operator PVRs can be configured without any additional considerations.

2.6.9 Mover

The Mover configuration will be largely dictated by the hardware configuration of the HPSS system. Each Mover can handle both disk and tape devices and must run on the node to which the storage devices are attached—with the exception of network attached peripherals (i.e., HiPPI attached disk arrays), in which case the Mover controlling the peripheral must be connected to the HiPPI network (to handle non-block aligned transfers and non IPI-3 transfers). The Mover is also capable of supporting a number of data transfer mechanisms for sending data to or receiving data from HPSS clients (e.g., TCP/IP, IPI-3 over HiPPI, and shared memory).

Asynchronous I/O must be enabled on the nodes on which the Mover will be running.

To enable asynchronous I/O on an AIX platform, use either the **chdev** command:

```
chdev -l aio0 -a autoconfig=available
```

or **smitty**:

```
smitty aio
<select “Change / Show Characteristics of Asynchronous I/O”>
<change “STATE to be configured at system restart” to “available”>
<enter>
```

All tape devices that will be used for HPSS data must be set to handle variable block sizes (to allow for the ANSI standard 80-byte volume label and file section headers).

To set the devices to use variable blocks on an AIX platform, either use the **chdev** command (substituting the appropriate device name for **rmt0** - also take into accounts differences in the interface based on the specific device driver supporting the device):

```
chdev -l rmt0 -a block_size=0
```

or **smitty**:

```
smitty tape
<select “Change / Show Characteristics of a Tape Drive”>
<select the appropriate tape device>
<change “BLOCK size (0=variable length)” to “0”>
<enter>
```

All locally attached magnetic disk devices (e.g., SCSI, SSA) should be configured using the pathname of the raw device (i.e., character special file).

The configuration of the storage devices (and subsequently the Movers that control them) can have a large impact on the performance of the system because of constraints imposed by a number of factors (e.g., device channel bandwidth, network bandwidth, processor power).

A number of conditions can influence the number of Movers configured and the specific configuration of those Movers:

- Each Mover executable is built to handle a single particular device interface (e.g., IBM SCSI-attached 3490E / 3590 drives, IBM BMUX-attached 3480 / 3490 / 3490E drives). If multiple types of device specific interfaces are to be supported, multiple Movers must be configured.

- Each Mover currently limits the number of concurrently outstanding connections. If a large number of concurrent requests are anticipated on the drives planned for a single Mover, the device work load should be split across multiple Movers (this is primarily an issue for Movers that will support disk devices).

- The planned device allocation should be examined to verify that the device allocated to a single node will not overload that node's resource to the point that the full transfer rates of the device cannot be achieved (based on the anticipated storage system usage). To off-load a single node, some number of the devices can be allocated to other nodes, and corresponding Movers defined on those same nodes.

- In general, the connectivity between the nodes on which the Movers will run and the nodes on which the clients will run should have an impact on the planned Mover configuration. For TCP / IP data transfers, the only functional requirement is that routes exist between the clients and Movers; however, the existing routes and network types will be important to the performance of client I / O operations. Movers must be configured on nodes that are HiPPI-attached for any devices that will store data that clients will access via IPI-3 over HiPPI.

- Mover to Mover data transfers (accomplished for migration, staging, and repack operations) also will impact the planned Mover configuration. For devices that support storage classes for which there will be internal HPSS data transfers, the Movers controlling those devices should be configured such that there is an efficient data path among them. If Movers involved in a data transfer are configured on the same node, the transfer will occur via a shared memory segment (involving no explicit data movement from one Mover to the other). For Movers controlling devices in the same storage hierarchy as a HiPPI attached disk array, it is desirable that those Movers be configured on HiPPI attached nodes to allow the data to be transferred directly to or from the disk array (without being forwarded to the Mover controlling that disk array).

2.6.10 Logging Service

Logging Services is comprised of the Log Daemon, Log Client, and Delog processes.

If central logging is enabled (default), log messages from all HPSS servers will be written by the Log Daemon to a common log file. There is a single Log Daemon process. It is recommended that the Log Daemon execute on the same node as the Storage System Manager, so that any Delogs executed by the Storage System Manager can access the central log file. If the central log file is accessible from other nodes (e.g., by NFS), it is not required that the Log Daemon execute on the same node as the Storage System Manager.

The Delog process is executed as an on-demand process by the Storage System Manger, or can be executed as a command line utility. If Delog is to be initiated from the Storage System Manager, the Delog process will execute on the same node as the Storage System Manager. If Delog is initiated from the command line utility, the central log file must be accessible from the node on which the

command is being executed (e.g., NFS mounted). Refer to Section 6.9.2.2 for detailed information on Delog.

If a Mover is being run in the non-DCE mode (where the processes that perform the device management and data transfers run on remote node from the processes that handle the configuration and managements interfaces), all Mover logging services will be directed to the Log Client running on the node on which the Mover DCE/Encina process runs.

2.6.11 Metadata Monitor

The primary function of the Metadata Monitor is to periodically collect statistics on the amount of space used by SFS and notify SSM whenever the percentage of space used exceeds various thresholds.

A single Metadata Monitor server monitors one and only one Encina SFS server. If multiple Encina SFS servers are used in an HPSS configuration, multiple Metadata Monitor servers should also be defined (one per Encina SFS server). A Metadata Monitor server does not necessarily have to execute on the same machine as the Encina SFS server it monitors.

2.6.12 NFS Daemons

The HPSS NFS Daemon is required for NFS access to HPSS name space objects and bitfile data. As mentioned in Section 2.5.5, NFS is recommended for enabling functionality not provided through other HPSS interfaces, such as access through standard system shells and utilities. Even if no client NFS access is required, the NFS interface may provide a useful mechanism for HPSS name space object administration.

The HPSS NFS Daemon must be run on a node that does not export UNIX file systems. That is, the HPSS and AIX NFS Daemon cannot be running at the same time. In addition, the NFS Daemon will require memory and local disk storage to maintain caches for HPSS file data and attributes. NFS memory and disk requirements are discussed in Section 2.9.3.2. Since the NFS Daemon communicates with the HPSS Name Server frequently, running the NFS Daemon on the same platform as the HPSS Name Server is recommended.

NFS access to HPSS exported directory subtrees is controlled through use of an exports file, a UNIX text file located on the NFS Daemon platform. Entries in the exports file identify what HPSS directory subtrees are accessible through NFS and what client systems can access them. Additional options are available to specify the type of access allowed and additional security related features. Export entry options are described in more detail in Sections 2.7.4 and 5.8.4.

Use of HPSS NFS also requires running an HPSS Mount Daemon component on the same platform as the HPSS NFS Daemon. As with standard UNIX NFS, the HPSS Mount Daemon provides client systems with the initial handle to HPSS exported directories.

It is possible to run several NFS Daemons in HPSS, but there are some restrictions. The NFS Daemons cannot run on the same platform and the directory trees supported by the NFS should not overlap. This is necessary because with overlapping directories, it is possible for different users to be updating the same file at essentially the same time with unpredictable results. This is typically called “the cache consistency problem.”

By default, files created with NFS will have the HPSS accounting index set to 0. Standard HPSS accounting mechanisms are supported only through the **UIDMAP** option described in section 5.8.3. If the **UIDMAP** option is specified, the user's default account index will be used for file creation. The **nfsmap** utility provides a capability for specifying an account index other than the user's default.

2.6.13 Startup Daemon

The Startup Daemon is responsible for starting, monitoring, and stopping the HPSS servers.The Daemon responds only to requests from the SSM System Manager. It shares responsibility with each HPSS server for ensuring that only one copy of the server runs at a given time. It helps the SSM determine whether servers are still running, and it allows the SSM to send signals to servers. Normally, the SSM stops servers by communicating directly with them, but in special cases, the SSM can instruct the Startup Daemon to send a **SIGKILL** signal to cause the server to shut down immediately.

If a server is configured to be restarted automatically, the Startup Daemon will restart the server when it is terminated abnormally. The server can be configured to be restarted forever, up to a number of auto-restarts or no auto-restart.

Choose a descriptive name for the Daemon that includes the name of the computer where the Daemon will be running. For example, if the Daemon will be running on a computer named tardis, use the descriptive name "Startup Daemon (**tardis**)". In addition, choose a similar convention for CDS names (for example, **/.:/hpss/hpssd_tardis**).

The Startup Daemon is started by running the script **/etc/rc.hpss**. This script should be added to the **/etc/inittab** file during the HPSS infrastructure configuration phase. However, the script should be manually invoked after the HPSS is configured and whenever the Startup Daemon dies. It is not generally desirable to kill the Daemon; if needed, it can be killed using the **hpss.clean** utility. The Startup Daemon must be run under the **root** account so that it has sufficient privileges to start the HPSS servers.

The Startup Daemon runs on every node where an HPSS server runs. For a Mover executing in the non-DCE mode, a Startup Daemon is only required on the node on which the Mover DCE/Encina process runs.

2.6.14 Storage System Management

SSM has three components: (1) the System Manager server, which communicates with all other HPSS components requiring monitoring or control, (2) the Data Server, which provides the bridge between the System Manager and the GUI, and (3) the GUI itself, which includes the Sammi runtime environment and the set of SSM windows.

The SSM Data Server need not run on the same host as the System Manager or Sammi; however, SSM performance will be better if all SSM components are running on the same host.

There can be only one SSM System Manager configured for an HPSS installation. The System Manager is able to handle multiple SSM clients (on different hosts or on the same host), and it is thus possible to have multiple SSM Data Servers connected to the same System Manager. In turn, the Data Server is able to handle multiple SSM users (or "consoles"), so it is possible to have multiple users running SSM via the same Data Server.

For the SSM user to be able to view the delogged messages from the SSM window, either the Log Daemon and the SSM session must be running on the same node or the delogged messages file must be accessible to the Sammi processes (e.g., via NFS).

2.6.15 HDM Considerations

HDM (hdm) consists of several processes that must be run on the same machine as the DFS file server. The hdm processes must also be run on the machine where the DFS aggregate resides. The main process (**hpss_hdm**) is an overseer that keeps track of the other hdm process, and restarts them, if necessary. Event dispatchers (**hpss_hdm_evt**) fetch events from DFS and assign the events to an event handler. Event handlers (**hpss_hdm_han**) process events by relaying a request to the DMAP Gateway. Migration processes (**hpss_hdm_mig**) migrate data to HPSS, and purge processes (**hdm_hdm_pur**) purge migrated data from DFS. A set of processes (**hpss_hdm_tcp**) accept requests from the DMAP Gateway, and perform the requested operation in DFS. Finally, a destroy process takes care of deleting files.

There are three types of event handlers, based on the type of activity that generates the events: administrative, name space, or data. Administrative activities include mounting and dismounting aggregates. Name space activities include creating, deleting, or renaming objects, and changing an object's attribute. Data activities include reading and writing file data. The number of processes allocated to handle events generated by these activities should be large enough to allow a reasonable mix of these activities to run in parallel.

HDM fetches events from DFS, puts them on a queue, and assigns the event to an appropriate event handler when it becomes free. The total number of entries allowed in the queue is set by a configuration parameter. If the value is not large enough, some of the event handlers may be starved. For example, if the queue fills up with data events, the name space handlers will be starved. Section 7.2.3.1 discusses the criteria for selecting the size of an event queue.

HDM logs outstanding name space events. If the HDM is interrupted, the log is replayed when the HDM restarts to ensure that the events have been processed to completion and the DFS and HPSS name spaces are synchronized. The size of the log is set by a configuration parameter, as discussed in Section 7.2.3.1.

HDM has two logs, each containing a list of files that are candidates for being destroyed. One of the logs, called the "zap log", keeps track of files on archived aggregates, while the other, called the "destroy log", keeps track of files on mirrored aggregates. Because of restrictions imposed by the DFS SMT, the HDM cannot take the time to destroy files immediately, so the logs serve as a record of files that need to be destroyed by the destroy process. The size of the zap log is bounded only by the file system where the log is kept, but the size of the destroy log is determined by a configuration parameter. If the destroy log is too small, the HDM will be forced to wait until space becomes available.

Since the HDM may be running on a machine where it cannot write error messages to the HPSS message log, it uses its own log. The HDM log consists of two file that are written in round-robin fashion. The size of these files are set by a configuration parameter.

HDM logging policy allows the system administrator to determine the type of messages written to the log file: alarm, event, debug, and / or trace messages. Typically, only alarms should be enabled, although event messages can be useful, and do not add significantly to the overhead. If a problem occurs, activating debug and trace messages will provide additional information that may help

identify the problem. However, these messages add to the overhead, and the system will perform best if messages are kept to a minimum. The type of messages that can be logged is set in the configuration file, and can be dynamically changed using the **hdm_admin** utility.

HDM migrates and purges files based on policies defined in the HDM policy configuration file. The administrator can establish different policies for each aggregate in the system. Migration policy parameters include the length of time to wait between migration cycles, and the amount of time that must elapse since a file was last accessed before it becomes eligible for migration. Purge policy parameters include the length of time to wait between purge cycles, the amount of time that must elapse since a file was last accessed, an upper bound specifying the percentage of DFS space that must be in use before purging will begin, and a lower bound specifying the target percentage of free space to reach before purging will be stopped.

2.6.16 Non-DCE Client Gateway

The Non-DCE Client Gateway provides HPSS access to applications running without DCE and/or Encina which make calls to the Non-DCE Client API. It does this by calling the appropriate Client APIs itself and returning the results to the client. Any system which wishes to make use of the Non-DCE Client APIs must have a properly configured and running NDCG.

The calls made to the NDCG by way of the NDAPI are not transactional in a global sense. In other words, even though the Client API calls made by the NDCG are themselves transactional, the encompassing calls made through the NDAPI are not. Therefore, at times it may be necessary for client programs to investigate the state of the HPSS system to determine if an API call successfully completed or not.

The NDCG does not implement security.

2.7 Storage Policy Considerations

This section describes the various policies that control the operation of the HPSS system.

2.7.1 Migration Policy

The migration policy provides the capability for HPSS to copy (migrate) data from one level in a hierarchy to a lower level. In addition, for disk storage classes, the migration policy also provides the capability to make duplicate copies of a bitfile.

Each storage class which is not at the bottom of a hierarchy can have an associated migration policy. The migration policy determines the conditions under which data in one storage class will be copied to a lower level in the storage hierarchy. For normal migration, this is the level just below the one represented by the storage class being migrated.

The migration policy also provides the capability to create duplicate copies of data if the storage class being migrated is a disk type storage class. In this case, data may be copied to one or more levels in the hierarchy below the level associated with the storage class being migrated.

The site administrator will need to monitor the usage of the storage classes that are provided and adjust both the migration and purge policies to obtain the desired results.

A migration policy must not be defined for a storage class at the lowest level in a hierarchy.

2.7.1.1 Migration Policy for Disk

A disk migration in HPSS copies (migrates) bitfiles from disk to a lower level storage class in the storage hierarchy. The removing or purging of the bitfiles from disk is controlled by the purge policy. The migration and purge policies for a given disk storage class work in conjunction to maintain sufficient storage space in the disk storage class.

The migration policy associated with a disk storage class controls the conditions under which data is copied from the disk storage class to one or more lower levels in the hierarchy. If the policy does not specify duplicate copies, the data will be copied to the level in the hierarchy just below the one associated with the storage class. Duplicate copies are specified in the migration policy by the **Copy Count**. The **Copy Count** and **Skip Factor** determine how many copies are made and to which levels the copies are made. A storage level is needed for every duplicate copy that is wanted.

When data is copied from the disk, the copied data will be marked purgeable but will not be deleted. Data is deleted by running purge on the storage class. If duplicate copies are created, the copied data is not marked purgeable until all copies have been successfully created. The migration policy and purge policy associated with a disk storage class must be set up to provide sufficient free space to deal with demand for storage. This involves setting the parameters in the migration policy to migrate a sufficient amount of files and setting the purge policy to reclaim enough of this disk space to provide the free space desired for users of the disk storage class.

Disk migration is controlled by several parameters:

- The **Last Update Interval** is used to prevent files that have been written recently from being migrated. Files that have been updated within this interval are not candidates for migration. Setting this value too high may limit how much data can be migrated and thus marked purgeable. This may prevent purge from purging enough free space to meet user demands. Setting this value too low could cause the same file to be migrated multiple times while it is being updated. The setting of this parameter should be driven by the amount of disk space in the storage class and how much new data is written to the storage class within a given time period.

- The **Free Space Target** controls the number of bytes to be copied by a migration run. The value of this parameter is important in association with the purge policy. The amount of data that is copied is potentially purgeable when the next purge on this storage class is run. This value must be set at a sufficient level so that enough purgeable space is created for the purge to meet the free space demands for users of this storage class.

- The **Runtime Interval** is used to control how often the migration process will run for this storage class. The administrator can also force a migration run to start via SSM. The value of this parameter is determined by the amount of disk space and the utilization of that disk space by users of the storage class. If the amount of disk space is relatively small and heavily used, the Runtime Interval may have to be set lower to meet the user requirements for free space in this storage class.

- The **Copy Count** and **Skip Factor** are used for making duplicate copies of bitfiles. The Copy Count specifies how many copies of the bitfile the site wants. The Skip Factor is used to tell the MPS at what hierarchy levels the copies are to be made. Each copy is made into

a separate lower level in the hierarchy. There must be sufficient levels in the hierarchy to support the selected values. The maximum number of levels in an HPSS storage hierarchy is 5. For example, if the disk storage class is at level 1 and the policy specifies a **Copy Count** of 2 and a **Skip Factor** of 2, the first copy will be made at level 2 and the second copy will be made at level 4.

- The **Request Count** is used to specify the number of parallel migration threads which are used for each destination level (i.e. copy) to be migrated.

- The **Migrate At Warning Threshold** option causes MPS to begin a migration run immediately when the storage class warning threshold is reached regardless of when the **Runtime Interval** is due to expire. This option allows MPS to begin migration automatically when it senses that a storage space crisis may be approaching.

- The **Migrate At Critical Threshold** option works the same as the **Migrate At Warning Threshold** option except that this flag applies to the critical threshold. Note that if the critical threshold is set to a higher percentage than the warning threshold (as it should be for disk storage classes), then the critical threshold being exceeded implies that the warning threshold has also been exceeded. If **Migrate At Warning Threshold** is set, then **Migrate At Critical Threshold** does not need to be set.

2.7.1.2 Migration Policy for Tape

The purpose of tape migration is to free up tape virtual volumes that have become full and have significant unused space on them. Unused space on a tape is generated when files on that tape are deleted or the disk copy of the file is overwritten. This happens because tape is not a rewritable media and thus new data is always written at the end of the tape. When data is deleted from the tape, the space associated with the data cannot be reused. The only way to reuse space on a tape is to copy all the valid data off the tape to other media and then reclaim the empty tape.

The tape migration process attempts to empty tapes by moving all the data on the tape to other storage media. When a tape becomes empty, it is a candidate for reuse. A special utility called **reclaim** resets the state of the empty tapes so that they can be reused. The **reclaim** utility can be run from SSM, but it should generally be set up to run on a periodic basis via the **cron** facility. For more information on **reclaim**, see Sections 6.4.8 and I.42. The **repack** utility can also be used to create empty tapes in a storage class. The administrator needs to determine whether a tape should be repacked based on the number of holes (due to file deletion) on tape. If a tape storage class is at the bottom of a hierarchy, **repack** and **reclaim** must be run periodically to reclaim wasted space. For more information on **repack**, see Sections 6.4.7 and I.49.

Since the purpose of the tape migration is to free up virtual volumes, all the storage segments that reside in the tape will be removed. Depending on how active a given storage segment is, it will either be copied to another tape in the same storage class or migrated to the next level in the hierarchy. A number of parameters in the migration policy are used to control the migration:

- The **Last Read Interval** and **Last Update Interval** parameters are used to determine if a bitfile is active or inactive. For inactive bitfiles, the migration / purge server will move the storage segments to the tapes in the next lower level storage class that is defined in the storage hierarchy. For active bitfiles, the MPS will move the storage segments to the tapes in the same storage level in which the storage segments are stored.

- The **Free Space Target** parameter controls the total number of free virtual volumes a site intends to maintain.

- The **Whole Files** parameter is used to avoid having the storage segments of a bitfile scattered over different tapes. When this field is set, if a bitfile on a virtual volume is selected to be migrated to the next level, any parts of this bitfile that are on different virtual volumes will also be migrated even if they would ordinarily not meet the criteria for being migrated. This tends to pack all the storage segments for a given file on the same virtual volume.

- The **Runtime Interval** parameter is used to control how often the migration process will run for this class. In addition, the administrator can force a migration run to start via SSM.

- The **Migrate At Warning Threshold** option causes MPS to begin a migration run immediately when the storage class warning threshold is reached regardless of when the **Runtime Interval** is due to expire. This option allows MPS to begin migration automatically when it senses that a storage space crisis may be approaching.

- The **Migrate At Critical Threshold** option works the same as the **Migrate At Warning Threshold** option except that this flag applies to the critical threshold. Note that if the critical threshold is set to a higher percentage than the warning threshold (as it should be for tape storage classes), then the critical threshold being exceeded implies that the warning threshold has also been exceeded. If **Migrate At Warning Threshold** is set, then **Migrate At Critical Threshold** does not need to be set.

Note that after the MPS removes all the segments from a virtual volume, the virtual volume is still not available for users to write. The **reclaim** utility should be used to reset the tapes to a useful state and make the VV available for use. Section 6.4.8 provides detailed information on using the **reclaim** virtual volume utility.

2.7.2 Purge Policy

The purge policy allows the MPS to remove the bitfiles from disk after the bitfiles have been migrated to a lower level of storage in the hierarchy. A purge policy should be defined for all disk storage classes that support migration. The specification of the purge policy in the storage class configuration enables the MPS to do the disk purging according to the purge policy for that particular storage class. Purge is run for a storage class on a demand basis. The MPS maintains current information on total space and free space in a storage class by periodically extracting this information from the HPSS Storage Servers. Based upon parameters in the purge policy, a purge run will be started when appropriate. The administrator can also force the start of a purge run via SSM.

The disk purge is controlled by several parameters:

- The **Start purge when space used reaches <nnn> percent** parameter allows sites control over how often they want the purging to take place. A purge run will be started for this storage class when the total space used in this class exceeds this value.

- The use of the **Stop purge when space used falls to <nnn> percent** parameter allows sites to control how much free space they want on their disk. The MPS will attempt to create this much free space by releasing purgeable space. Once the target is hit, the purge will terminate.

- The **Do not purge files accessed within <nnn> minutes** parameter determines the minimum amount of time a site wants to keep a file on disk. Files that have been accessed within this time interval are not candidates for purge.

- The **Purge by** list box allows sites to choose the criteria used in selecting files for purge. By default, files are selected for purge based on the creation time of their purge record. Alternately, the selection of files for purging may be based on the time the file was created or the time the file was last accessed.

- The **Purge Locks expire after <nnn> minutes** parameter allows sites to control how long a file can be purge locked before it will appear on the MPS report as an expired purge lock. The purge lock is used to prevent a file from being purged from the highest level of a hierarchy. Purge locks only apply to a hierarchy containing a disk on the highest level. HPSS will not automatically unlock purge locked files after they expire. HPSS simply reports the fact that they have expired in the MPS report.

Administrators should experiment to determine the parameter settings that will fit the needs of their site. If a site has a large amount of disk file write activity, the administrator may want to have more free space and more frequent purge runs. However, if a site has a large amount of file read activity, the administrator may want to have smaller disk free space and less frequent purge runs, and allow files to stay on disk for a longer time.

A purge policy must not be defined for a tape storage class or a storage class which does not support migration.

2.7.3 Accounting Policy

HPSS supports gathering data that can be used by customer sites to implement a local accounting policy. Several parts of the Accounting Policy must be defined when the HPSS System is initially set up to determine the correct Accounting Style: Site-Style or UNIX-Style.

In the current release, the Accounting Style field in the Accounting Policy is not being used. The accounting style of a site is determined on a user-to-user basis by the presence of the **AA=<default-acct-id>** *string in the DCE Registry Information field or the DCE Registry HPSS.gecos ERA. Therefore, care must be taken when creating the DCE account for an HPSS user.*

If an user entry in the DCE registry contains a string of the form **AA=<default-acct-id>** *in its GECOS (Miscellaneous Info) field,* **<default-acct-id>** *is the default account index to be used by HPSS when creating files for this user. the that user is using site_style accounting. The GECOS field can be set when using the hpssuser utility, rgy_edit or dcecp command. The dcecp command is used to set or change the DCE Registry ERA.*

If UNIX-style accounting is used, it is important that the **AA=<default-acct-id>** *string does not appear in the GECOS field.*

The Account Map file, which maps HPSS account indexes to site specific accounts, must be set up as well as any site specific scripts or utilities needed to process the Accounting Report File.

Accounting Style Guidelines

Each site must develop software to meet their own accounting requirements. The site will define a table called the Account Map that will correlate the HPSS Account Index number with the necessary accounting information.

HPSS accounting allows the System Administrator to:

- Store account index numbers in the bitfile metadata.
- Determine the number of file accesses, amount of data transferred, and space usage per account index per hierarchy per storage class.
- For Site-style accounting, change the user's default account index using the registry editor **rgy_edit**.
- Write accounting data to a flat file or substitute a locally written module to write accounting data directly into a local data base.

For Site-style accounting, the HPSS user is able to:

- Change his or her default account index by contacting the System Administrator.
- Maintain a current session account index.
- Change the account index of a file.

For Site-style accounting, HPSS will not validate or authenticate HPSS Account Index numbers. UNIX-style accounting will provide valid, authentic account index numbers as UIDs.

Some accounting questions need to be answered by the site accounting department or whoever performs the accounting function. The accounting requirements at a site will affect how the Account Map should be set up. Each of these areas is discussed in detail in Appendix D.

2.7.4 Security Policy

HPSS server authentication and authorization make extensive use of Distributed Computing Environment (DCE) authentication and authorization mechanisms. Each HPSS server has configuration information that determines the type and level of services available to that server. HPSS software uses these services to determine the caller identity and credentials. Server security configuration is discussed more in Section 5.3.

Once the identity and credential information of a client has been obtained, HPSS servers enforce access to their interfaces based on permissions granted by the access control list attached to a Security object in the server's Cell Directory Service (CDS) directory. Access to interfaces that change a server's metadata generally requires control permission. Because of the reliance on DCE security features, HPSS security is only as good as the security employed in the HPSS DCE cell.

HPSS client interface authentication and authorization security features for end users depend on the interface, and is discussed in the following subsections.

2.7.4.1 Client API

The Client API interface uses DCE authentication and authorization features. Applications that make direct Client API calls must obtain DCE credentials prior to making those calls. Credentials can either be obtained at the command level via the **dce_login** mechanism, or within the application via the **sec_login_set_context** interface.

2.7.4.2 Non-DCE Client API

The Non-DCE Client API does not implement security features. Security will be added in HPSS Release 4.2

2.7.4.3 FTP/PFTP

By default, FTP and Parallel FTP (PFTP) interfaces use a username / password mechanism to authenticate and authorize end users. The end user identity credentials are obtained from the principal and account records in the DCE security registry. However, FTP and PFTP users do not require maintenance of a login password in the DCE registry. The FTP / PFTP interfaces allow sites to use site-supplied algorithms for end user authentication. This mechanism is enabled by running an appropriate authentication manager such as **auth_dcegss**.

2.7.4.4 DFS

DFS uses DCE authentication and authorization.

2.7.4.5 NFS

Though the HPSS NFS client interface does not directly support an end user login authorization mechanism, standard NFS export security features are supported to allow specification of read-only, read-mostly, read-write, and root access to HPSS subtrees for identified client hosts. HPSS NFS does not support Sun MicroSystems' Network Information Services to validate client hosts. HPSS NFS does provide an option to validate the network address of hosts attempting to mount HPSS directories. The default configuration disables this check. To enable client address validation, export the variable HPSS_MOUNTD_IPCHECK in the HPSS environments file (**hpss_env**). An option to specify mediation of user access to HPSS files by a credentials mapping is also provided. Export entry options are described further in Section 5.8.4.

If the user mapping option is specified, user access requires an entry in the NFS credentials map cache and user credentials are obtained from that cache. Entries in the credentials map cache, maintained by the NFS Daemon, are generated based on site policy. For instance, entries may be established by allowing users to run a site-defined map administration utility, or they may be set up at NFS startup time by reading a file. They can also be added by running a privileged map administration utility such as the **hpss_nfsmap** utility.

2.7.4.6 Bitfile

Enforcement of access to HPSS bitfile data is accomplished through a ticketing mechanism. An HPSS security ticket, which contains subject, object, and permission information, is generated by

the HPSS Name Server. Ticket integrity is certified through a checksum that is encrypted with a key shared by the Name Server and Bitfile Server. When access to file data is requested, the ticket is presented to the HPSS Bitfile Server, which checks the ticket for authenticity and appropriate user permissions. The Name Server/Bitfile Server shared key is generated at Name Server startup, and is sent to the Bitfile Server using an encrypted DCE remote procedure call to set up a shared security context. If the DCE cell in which HPSS resides does not support packet integrity, it is recommended that the Name Server and Bitfile Server components run on the same platform.

2.7.4.7 Name Space

Enforcement of access to HPSS name space objects is the responsibility of the HPSS Name Server. The access rights granted to a specific user are determined from the information contained in the object's ACL.

2.7.4.8 Security Audit

HPSS provides capabilities to record information about authentication, file creation, deletion, access, and authorization events. The security audit policy in each HPSS server determines what audit records a server will generate. In general, all servers can create authentication events, but only the Name Server and Bitfile Server will generate file events. The security audit records are sent to the log file and are recorded as security type log messages.

2.7.5 Logging Policy

The logging policy provides the capability to control which message types are written to the HPSS log files. In addition, the logging policy is used to control which alarms, events, and status messages are sent to the Storage System Manager to be displayed. Logging policy is set on a per server basis. Refer to Section 5.4.4 for a description of the supported message types.

If a logging policy is not explicitly configured for a server, the default is to log all message types with the exception of Trace messages. All Alarm, Event, and Status messages generated by the server will also be sent to the Storage System Manager.

The administrator might consider changing a server's logging policy under one of the following circumstances.

- A particular server is generating excessive messages. Under this circumstance, the administrator could use the logging policy to limit the message types being logged and/or sent to the Storage System Manager. This will improve performance and potentially eliminate clutter from the HPSS Alarms and Events window. Message types to disable first would be Trace messages followed by Debug and Request messages.

- One or more servers is experiencing problems which require additional information to trouble shoot. If Alarm, Debug, or Request message types were previously disabled, enabling these message types will provide additional information to help diagnose the problem. HPSS support personnel might also request that Trace messages be enabled for logging.

2.7.6 Location Policy

The location policy provides the ability to control how often Location Servers at an HPSS site will contact other servers. It determines how often Bitfile Servers are queried for storage statistics as well as how often remote Location Servers are contacted to exchange server location information. An administrator tunes this by balancing the local site's desire for accuracy of Class of Service selection during file creations against the desire to avoid extra network and server load.

2.8 Storage Characteristics Considerations

This section defines key concepts of HPSS storage and the impact the concepts have on HPSS configuration and operation. These concepts, in addition to the policies described above, have a significant impact on the usability of HPSS.

Before an HPSS system can be used, the administrator has to create a description of how the system is to be viewed by the HPSS software. This process consists of learning as much about the intended and desired usage of the system as possible from the HPSS users and then using this information to determine HPSS hardware requirements and determine how to configure this hardware to provide the desired HPSS system. The process of organizing the available hardware into a desired configuration results in the creation of a number of HPSS metadata objects. The primary objects created are classes of service, storage hierarchies, and storage classes.

An HPSS storage class is used to group storage media together to provide storage with specific characteristics for HPSS data. The attributes associated with a storage class are both physical and logical. Physical media in HPSS are called physical volumes. Physical characteristics associated with physical volumes are the media type, block size, the estimated amount of space on volumes in this class, and how often to write tape marks on the volume (for tape only). Physical media are organized into logical virtual volumes. This allows striping of physical volumes. Some of the logical attributes associated with the storage class are virtual volume block size, stripe width, data transfer rate, latency associated with devices supporting the physical media in this class, and storage segment size (disk only). In addition, the storage class has attributes that associate it with a particular migration policy and purge policy to help manage the total space in the storage class.

An HPSS storage hierarchy consists of multiple levels of storage with each level representing a different class of storage media (i.e., a storage class). Files are moved up and down the storage hierarchy via stage and migrate operations, respectively, based upon storage policy, usage patterns, storage availability, and user requests. If data is duplicated for a file at multiple levels in the hierarchy, the more recent data is at the higher level (lowest level number) in the hierarchy. Each hierarchy level is associated with a single storage class.

Class of Service (COS) is an abstraction of storage system characteristics that allows HPSS users to select a particular type of service based on performance, space, and functionality requirements. Each COS describes a desired service in terms of such characteristics as minimum and maximum file size, transfer rate, access frequency, latency, and valid read or write operations. A file resides in a particular COS and the class is selected when the file is created. Underlying a COS is a storage hierarchy that describes how data for files in that class are to be stored in the HPSS system. As of Release 4.1, a COS can be associated with a fileset such that all files created in the fileset will use the same COS.

A file family is an attribute of an HPSS file that is used to group a set of files on a common set of tape virtual volumes. Release 4.1.1 supports grouping of files only on tape volumes. In addition,

families can only be specified in Release 4.1.1 by associating a family with a fileset, and creating the file in the fileset. When a file is migrated from disk to tape, it is migrated to a tape virtual volume assigned to the family associated with the file. If no family is associated with the file, the file is migrated to the next available tape not associated with a family (actually to a tape associated with family zero). If no tape virtual volume is associated with the family, a blank tape is reassigned from family zero to the file's family. The family affiliation is preserved when tapes are repacked.

The relationship between storage class, storage hierarchy, and COS is shown in Figure 2-2.

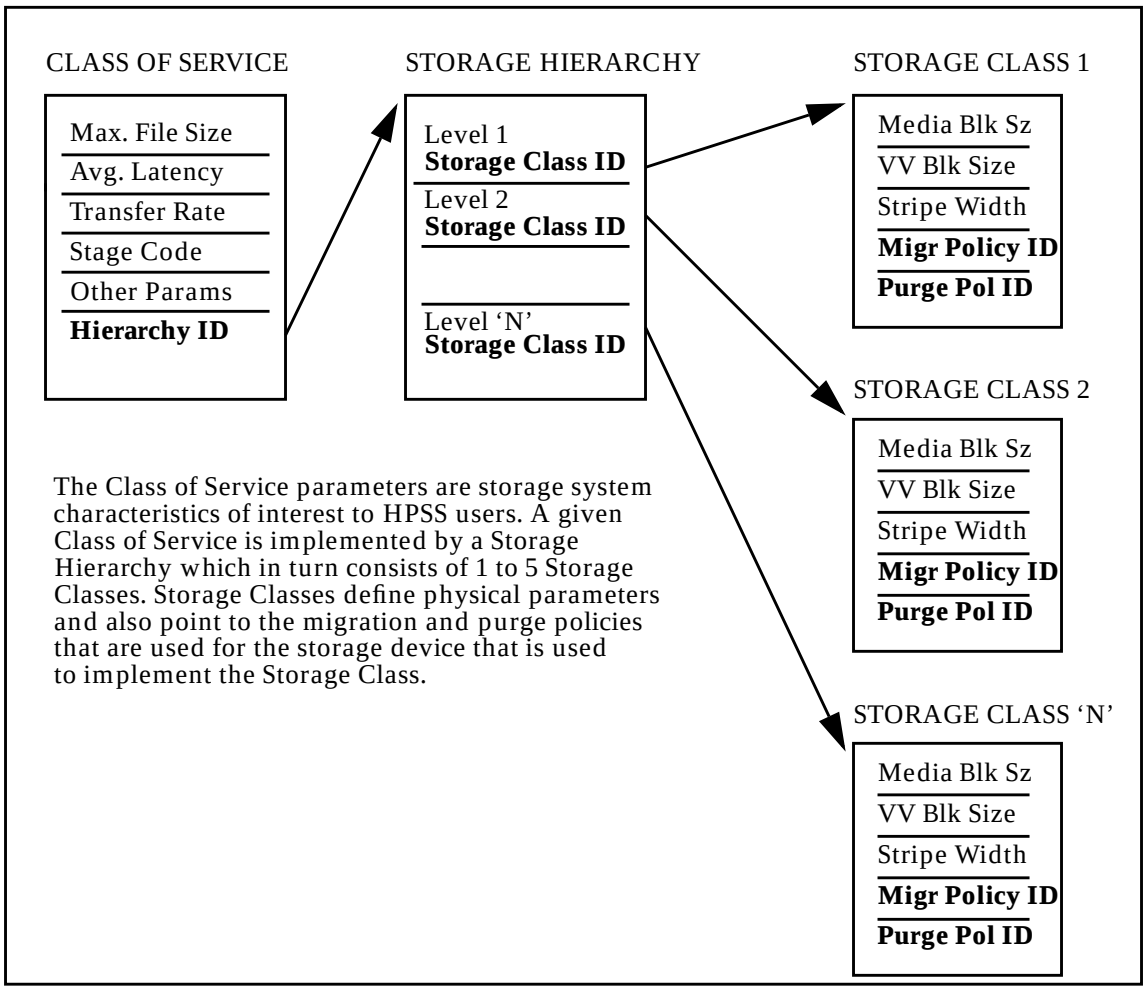


Figure 2-2 Relationship of Class of Service Storage Hierarchy, and Storage Class

2.8.1 Storage Class

Each virtual volume and its associated physical volumes belong to a particular storage class in HPSS. The SSM provides the capability to define storage classes and to add and delete virtual volumes to and from the defined storage classes. A storage class is identified by a storage class ID and its associated attributes. For detailed descriptions of each attribute associated with a storage class, see Section 5.5.1. Once a storage class has been defined, great care must be taken if the definition is to be changed or deleted. This especially applies to media and VV block size fields.

The sections that follow give guidelines and explanations for creating and managing storage classes.

2.8.1.1 Media Block Size Selection

Guideline: Select a block size that is smaller than the maximum physical block size that a device driver can handle.

Explanation: For example, if a site has ESCON attached tape drives on an RS6000, the driver can handle somewhat less than 64 KB physical blocks on the tape. A good selection here would be 32 KB. See Section 2.8.1.12 for recommended values for tape media supported by HPSS.

2.8.1.2 Virtual Volume Block Size Selection (disk)

Guideline **1:** The virtual volume (VV) block size must be a multiple of the underlying media block size.

Explanation: If it were not, media would be wasted with a proliferation of short blocks.

Guideline 2: A VV can be no larger than 16,384 * VV Block Size.

Explanation: There is a limitation in the Storage Server that has an impact upon how VV block sizes can be selected. The Storage Server maintains a map that shows the allocation status of each virtual block on the VV. The size of the map is limited to 16,384 (16 K) entries. For example, if building a 4-wide stripe across 4GB disks, the size of the virtual volume is 16GB. The smallest legal VV Block Size would be 1 MB.

2.8.1.3 Virtual Volume Block Size Selection (tape)

Guideline 1: The VV block size must be a multiple of the media block size

Explanation: If it were not, media would be wasted with a proliferation of short blocks.

Guideline 2: Pick a block size such that the size of the buffer that is being used by writers to this storage class is an integral multiple of the block size.

Explanation: For example, assume files are being written via standard FTP directly into a tape storage class. Also assume FTP is set up to use a 4 MB buffer size to write the data. This means that writes are done to the tape with a single 4 MB chunk being written on each write call. If the tape virtual volume block size is not picked as indicated by the guideline, two undesirable things will happen. A short block will be written on tape for each one of these writes, which will waste data storage space, and the Storage Server will build a separate storage segment for the data associated with each write, which will waste metadata space. See also Section 2.8.1.6 for further information about selecting block sizes.

2.8.1.4 Stripe Width Selection

Stripe width determines how many physical volumes will be accessed in parallel when doing read / writes to a storage class.

Guideline 1: The stripe width should be less than half the available drives if multiple sets of drives are required for certain operations.

Explanation: There must have enough drives (tape or disk) to support the stripe width selected. If planning to run tape repack on media in this storage class, the stripe width cannot be greater than half the number of drives available. In addition, if doing tape-to-tape migration between two storage classes that have the same media type and thus potentially share the same drives, the stripe width cannot be greater than half the number of drives available. Also, doing multiple copies from disk to two tape storage classes with the same media type will perform very poorly if the stripe width in either class is greater than half the number of drives available. The **recover** utility also requires 2 times the stripe width drives to be available to recover data from a damaged virtual volume if invoked with the repack option.

*Guideline 2***:** Select a stripe width that results in data transmission rates from the drives matching or being less than what is available through rates from the network.

Explanation: Having data transmission off the devices that is faster than the network will waste device resources, since more hardware and memory (for Mover data buffers) will be allocated to the transfer, without achieving any performance improvement over a smaller stripe width. Also, if a large number of concurrent transfers are frequently expected, it may be better (from an overall system throughput point of view) to use stripe widths that provide something less than the throughput supported by the network - as the aggregate throughput of multiple concurrent requests will saturate the network and overall throughput will be improved by requiring less device and memory resources.

Guideline **3:** For smaller files, use a small stripe width or no striping at all.

Explanation: For tape, the situation is complex. If accessing tape directly as opposed to migration operations, accessing a file will generally result in all the volumes having to be mounted and positioned before a transmission can begin. This latency will be driven by how many mounts can be done in parallel, plus the mount time for each physical volume. If the file being transmitted is too small, all of this latency could cause performance to be worse than if a smaller stripe or no striping were used at all.

As an example of how to determine stripe width based on file size and drive performance, imagine a tape drive that can transmit data at about 10 MB / second and it takes about 20 seconds on average to mount and position a tape. For a one-way stripe, the time to transmit a file would be:

File Size(MB) / 10 + 20

Now consider a 2-way stripe for this storage class which has only one robot. Also assume that this robot has no capability to do parallel mounts. In this case, the transmission time would be:

File Size(MB) / 20 + 2 * 20

An algebraic calculation indicates that the single stripe would generally perform better for files that are less than 400 MB in size.

Guideline **4:** Migration can use larger stripe widths.

Explanation: For migration operations from disk, the tape generally is mounted and positioned only once. In this case, larger stripe widths can perform much better. The number of drives available for media in this storage class also should be a multiple of the stripe width. If not, less than optimal use of the drives is likely unless the drives are shared across storage classes.

2.8.1.5 Blocks Between Tape Marks Selection

Blocks between tape marks is the number of physical media blocks written before a tape mark is generated. The tape marks are generated for two reasons: (1) To force tape controller buffers to flush so that the Mover can better determine what was actually written to tape, and (2) To quicken positioning for partial file accesses. Care must be taken, however in setting this value too low, as it can have a negative impact on performance. For recommended values for various media types, see Section 2.8.1.12.

2.8.1.6 Storage Segment Size Selection (disk only)

The Bitfile Server maps files into a series of storage segments. The size of the storage segments are controlled by the **Minimum Storage Segment Size** parameter, the **Maximum Storage Segment Size** parameter, and the **Average Number of Segments** parameter. The smallest amount of data storage that can be allocated to a file is determined by the **Minimum Storage Segment Size** parameter. This parameter should be chosen with disk space utilization in mind. For example, if writing a 4 KB file into a storage class where the storage segment size is 1,024 KB, 1,020 KB of the space will be wasted. The Bitfile Server will, when file size information is available, attempt to choose an optimal storage segment size between **Minimum Storage Segment Size** and **Maximum Storage Segment Size** with the goal of creating **Average Number of Segments** for the bitfile. The storage segment size will also be chosen as a power of 2 multiple of the **Minimum Storage Segment Siz**e parameter.

*Guideline 1***:** When a large range of file sizes are to be stored on disk, define multiple disk storage classes with appropriate storage segment sizes for the sizes of the files that are expected to be stored in each storage class.

Explanation: The Class of Service (COS) mechanism can be used to place files in the appropriate place. Note that although the Bitfile Server provides the ability to use COS selection, current HPSS interfaces only take advantage of this in two cases. First, the **pput** command in PFTP automatically takes advantage of this by selecting a COS based on the size of the file. If the FTP implementation on the client side supports the **alloc** command, a COS can also be selected based on file size. Files can also be directed to a particular COS with FTP and PFTP commands by using the **site setcos** command to select a COS before the files are stored. When setting up Classes of Service for disk hierarchies, take into account both the Storage Segment Size parameter and the Maximum Storage Segment Size parameter in determining what range of file sizes a particular COS will be configured for.

NFS is more of a challenge in this area. NFS puts all the files it creates into a single COS that is specified in the NFS Daemon configuration parameters. Since the HPSS NFS daemon is an NFS V2 implementation the maximum file size is 2GB. It is important to use the Storage Segment Size parameter, the Maximum Storage Segment Size parameter and the Average Number of Segments parameter to allow for a wide range of file sizes to be treated in a reasonably optimal way.

Guideline **2:** For optimal results, it is best to select the Storage Segment Size to be either the same as the Stripe Length or a power of 2 multiple of the Stripe Length.

Explanation: Storage Segment sizes that are a power of 2 multiple of the stripe length will lead to optimal use of the disk space. For additional explanation see Section 2.8.1.4.

2.8.1.7 Maximum Storage Size Selection (disk only)

This parameter, along with Storage Segment Size and Average Number of Storage Segments, is used by the Bitfile Server to optimally choose a storage segment size for bitfiles on disk. The largest storage segment size that can be selected for a file in a storage class is limited by this parameter.

Guideline 1: In order to avoid creating excessive fragmentation of the space on disks in this storage class, it is recommended that this parameter be set no higher that 5% of the size of the smallest disk allocated to this storage class.

2.8.1.8 Maximum VVs to Write (tape only)

This parameter restricts the number of tape VVs, per storage class, that can be concurrently written by the Tape Storage Server. The purpose of the parameter is to limit the number of tape VVs being written to prevent files from being scattered over a number of tapes and to minimize tape mounts. The number of tape drives used to write files in the storage class will be limited to approximately the value of this field times the stripe width defined for the storage class. Note that this field only affects tape write operations. Read operations are not limited by the value defined by this parameter.

2.8.1.9 Average Number of Storage Segments (disk only)

This parameter, along with Storage Segment Size and Maximum Storage Segment Size, is used by the Bitfile Server to optimally choose a storage segment size for bitfiles on disk. The Bitfile Server attempts to choose a storage segment size between Storage Segment Size and Maximum Storage Segment Size that would result in creating the number of segments indicated by this parameter.

Guideline 1: For best results, it is recommended that small values (< 10) be used. This results in minimizing metadata created and optimizing migration performance. The default of 4 will be appropriate in most situations.

2.8.1.10 PV Estimated Size Selection

Guideline 1: For tape, select a value that represents how much space can be expected to be written to a physical volume in this storage class with compression factored in.

Explanation: The Storage Server will fill the tape regardless of the value indicated. Setting this value differently between tapes can result in one tape being favored for allocation over another.

Guideline 2: For disk, this value should be the exact number of bytes on the disk. Also for disk, it is best if this value is a multiple of the size of the virtual volume block size. For disk storage classes, it is also good if this value can be a multiple of the Storage Segment Size; otherwise, media space may be wasted.

Explanation: The size of the virtual volume is the size of the physical volume * Stripe Width.

Rule: For disk, the **PV Estimated Size** value must be less than or equal to the **Bytes on Device** value described in Section 5.7.

2.8.1.11 Optimum Access Size Selection

Guideline: Generally, a good value for Optimum Access Size is the Stripe Length.

Explanation: This field is advisory in nature in the current HPSS release. In the future, it may be used to determine buffer sizes. Generally, a good value for this field is the Stripe Length; however, in certain cases, it may be better to use a buffer that is an integral multiple of the Stripe Length. The simplest thing at the present time is to set this field to the Stripe Length. It can be changed in the future without complication.

2.8.1.12 Some Recommended Parameter Values for Supported Storage Media

Tables 2-1 and 2-2 contain suggested values for storage resource attributes based on the type of storage media. The specified values are not the *only* acceptable values, but represent reasonable settings for the various media types. See Section 2.7.6 for more information about setting the storage characteristics.

Disk Media Parameters

Table 2-2 contains attributes settings for the supported disk storage media types.

Table 2-2 Suggested Block Sizes for Disk

Disk Type	Media Block Size	Minimum Access Size	Minimum Virtual Volume Block Size	Notes
SCSI Attached	4 KB	0	1 MB	1
SSA Attached	4 KB	0	1 MB	1
Fibre Channel Attached	4 KB	0	1 MB	1
Maximum Strategies Inc. and IBM 9570	32 KB or 64 KB	4 MB	4 MB	2

In Table 2-2:

- Disk Type is the specific type of media to which the values in the row apply.
- Media Block Size is the block size to use in the storage class definition. For disk, this value should also be used when configuring the Mover devices that correspond to this media type. Note that this value will not limit the amount of data that can be read from or written to a disk in one operation—it is used primarily to perform block boundary checking to ensure that all device input/output requests are block aligned. This value should correspond to the physical block size of the disk device.

- Minimum Access Size is the smallest size data access requests that should regularly be satisfied by the corresponding media type. Any accesses for less than the listed amount of data will suffer severe performance degradation. A value of zero indicates that the media is in general suitable for supporting the small end of the system's data access pattern.

- Minimum Virtual Volume Block Size is the smallest block size value that should be used for the corresponding media type when physical volumes are combined to form a striped virtual volume. A value smaller than that specified may result in severely degraded performance when compared to the anticipated performance of the striped virtual volume.

- **Notes**:

    1. When SCSI, SSA or Fibre Channel attached disks are combined to form striped virtual volumes, the minimum access size should become—at a minimum—the stripe width of the virtual volume multiplied by the virtual volume block size. If not, data access will only use a subset of the striped disks and therefore not take full advantage of the performance potential of the virtual volume.

    2. The block size used for the Maximum Strategies Inc., and IBM 9570 disk arrays should be based on the values used when the array was formatted.

Tape Media Parameters

Table 2-3 contains attributes settings for the supported tape storage media types.

Table 2-3 Suggested Block Sizes for Tape

Tape Type	Media Block Size	Blocks Between Tape Marks	Estimated Physical Volume Size
3480	32 KB	512	200 MB
3490	32 KB	1024	400 MB
3490E - BMUX	32 KB	1024	800 MB
3490E - SCSI	64 KB	512	800 MB
3590	256 KB	512	10, 20GB
3590E	256 KB	512	20, 40GB
StorageTek 4220	32 KB	512	200 MB
StorageTek Redwood	256 KB	512	50 GB
StorageTek Timberline	64 KB	1024	800 MB
StorageTek 9840	256 KB	1024	20 GB
Ampex DST-312	1 MB	1024	50, 150, 330 GB
Ampex DST-314	1 MB	1024	100, 300, 660 GB

In Table 2-3:

- Tape Type is the specific type of media to which the values in the row apply.
- Media Block Size is the block size to use in the storage class definition. For tape, this will be the size of the data blocks that will be written to tape. Note that for tape devices, the Mover configuration does not contain the media block size. The selected block size is set on a per physical volume basis at the request of the Tape Storage Server. This value may have a significant impact on data transfer performance, as for most tape devices each input/output request must be for the media block size. If however, a large block size is used for relatively small write requests, space may be wasted on the tape if the drive can not compress the unused portion of the data blocks.
- Block Between Tape Marks is the number of data blocks to be written between tape marks, to be set in the storage class definition. A relatively small value has the benefits of lower access time for position to the middle of a file (the start of a file will be just after a tape mark) and less data will need to be re-written to another volume when end-of-media is reached. Small values have the penalties of increased media space that cannot be used for user data and some of level of performance penalty to handle the tape marks while reading or writing data.
- Estimated Physical Volume Size is the estimated size of the physical volumes to be set in the storage class definition. These values are based on the expected media to be used with the specified type. In some cases, different length tape media may be used, which may have an effect on the estimated size for a given physical volume (e.g., regular or extended length 3480/3490 format cartridges). Note that the values listed do not take into account any data compression that may be performed by the tape drive.

2.8.2 Storage Hierarchy

Each HPSS file is stored in a single storage hierarchy consisting of an ordered list of storage classes. A storage hierarchy can have up to 5 levels starting with level 1. The highest level is always level 1 and the lowest possible level is level 5. For example, a level 1 storage class could be fast disk while a level 5 storage class could be a slow, large capacity tape system. The SSM provides operational capabilities to define storage hierarchies. A storage hierarchy is identified by a storage hierarchy ID and its associated attributes. For detailed descriptions of each attribute associated with a storage hierarchy, see Section 5.5.2. The following is a list of rules and guidelines for creating and managing storage hierarchies.

Rule 1: All writes initiated by clients are directed to the highest level (level 1) in the hierarchy.

Rule 2: The data of a file at a storage class level in a hierarchy is associated with a single Storage Server.

Rule 3: Parts or all of a file may appear at multiple levels in a storage hierarchy. If data for a file does appear at multiple levels of the hierarchy, the data at the higher level is always the more recent data.

Rule 4: Migration of data does not skip levels in the hierarchy, except in the special case of creating duplicate copies when doing disk migration.

Rule 5: The client stage command can only stage data to the top level (level 1) in the hierarchy.

Rule 6: A given storage class can only occur once in the same hierarchy.

Guideline: Care must be taken when selecting the storage segment size for a disk storage class. If data is to be migrated from this disk to a tape storage class, the storage segment size as specified by the **Minimum Storage Segment Size** parameter in the storage class definition should meet one of the following conditions. These rules are associated with the internal migration process. If not adhere to, can result in excessive storage segment creations.

- Storage segment size on disk is an integral multiple of the stripe length on tape. If this option is selected, normally these values would be set equal.
- Stripe length on tape is an integral multiple of the storage segment size on disk. In this case, the multiple must be less than or equal to 16.

2.8.3 Class of Service

Each HPSS file belongs to a single Class of Service (COS) that is selected when the file is created. It is selected via Class of Service Hints information passed to the Bitfile Server when the bitfile is created. If using the Client API, the application program has full access to this hints information. If using NFS, the COS is the same for each file and is determined by the COS ID that is specified in the NFS configuration information. For FTP, there is a quote command to set the desired COS ID. A **pput** request in PFTP automatically selects a COS based on file size by using the COS Hints unless the user explicitly selects the COS. The SSM provides operational capabilities to define classes of service. A COS is identified by a COS ID and its associated attributes. For detailed descriptions of each attribute associated with a class of service, see Section 5.5.3.

The Force Selection flag can be set in the COS definition to eliminate automatic selection. If this flag is set, the designated COS can only be selected by asking for the COS by ID or Name.

The paragraphs that follow give guidelines and explanations for creating and managing classes of service.

2.8.3.1 Selecting Minimum File Size

Guideline: This field can be used to indicate the smallest file that should be stored in this COS.

Explanation: This limit is not enforced and is advisory in nature. If the COS Hints mechanism is used, minimum file size can be used as a criteria for selecting a COS. Currently, PFTP and FTP clients that support the **alloc** command will use the size hints when creating a new file. Ensure that the Minimum File Size and the Maximum File Size do not overlap; otherwise the data placement may be indeterminate.

2.8.3.2 Maximum File Size

Guideline: This field can be used to indicate the largest file that is to be stored in this COS.

Explanation: If the **Enforce Max File Size** option is selected, an attempt to perform an operation on a file that would cause this value to be exceeded will be rejected. The underlying storage hierarchy

should be set up so that the defined storage classes support files of this size in a reasonable fashion. For details, see Sections 2.8.1 and 2.8.2 on storage class and storage hierarchies. This field can be used via the COS Hints mechanism to affect COS selection. PFTP and FTP clients that support the **alloc** command will use the size hints when creating a new file. Ensure that the Minimum File Size and the Maximum File Size do not overlap; otherwise the data placement may be indeterminate.

2.8.3.3 Selecting Stage Code

This field determines whether a file is to be staged to the highest level in the storage hierarchy when the file is opened. This field can be used via the COS Hints mechanism to affect COS selection. The valid options are as follows:

Guideline **1:** Select the **No Stage** option if file to be staged on open is not desired.

Explanation: Data read from the file may come from lower levels in the storage hierarchy if the data does not already exist at the top level. This option is normally selected if the top level in the hierarchy is not disk or if the users of files in this COS wish to control staging directly via user stage requests.

Guideline 2: Select the **Stage on Open** option if it is desirable have the entire file staged to the top level in the hierarchy and to wait until this completes before the open call returns.

Explanation: This would be a commonly selected option when the top level is disk and the files in this class are small to moderate in size. Also, use this option if want to be guaranteed that the file is completely and successfully staged. If the stage fails, the open will return with an error.

Guideline 3: Select the **Stage on Open Async** option if wishing to stage the entire file to the top level in the hierarchy and do not want the open to block.

Explanation: When this option is selected, the file is staged up in increments and the read and write calls that are accessing this file are blocked only until that portion of the file they wish to operate on has been completely staged. Normally, this option would be selected when the top level is disk and the files in this class are fairly large in size. This option is only available when the top level in the hierarchy is disk. If the top level is tape and this option is specified, the **Stage on Open** option will be used instead.

Guideline 4: Select the **Stage on Open Background** option if you want the stage to be queued internally in the Bitfile Server and processed by a background BFS thread on a scheduled basis. The open request will return with success if the file is already staged. If the file needs to be staged an internal staged request is placed in queue which will be selected and processed by the Bitfile Server in the background. A busy error is returned to the caller. This option allows a large number of stages (up to 2000) to be queued in the bitfile server and processed as thread resources are available. The other stage options will result in an busy error if thread resources are not immediately available to process the request.

2.8.3.4 Selecting Optimum Access Size

This field is only advisory in nature; however, for later releases it may be used by interfaces in dynamically selecting good buffer sizes.

Guideline 1: Generally, if the file is being staged on open, **Optimum Access Size** should be set to the same value as **Optimum Access Size** is set to in the storage class that is at the top of the hierarchy.

Guideline 2: If data is not being staged to the top level before it is read (either automatically or by user command), select a value that is an integral multiple of the largest **Optimum Access Size** field found among the storage classes that make up this hierarchy.

Explanation: Attempting to set this field correctly will potentially ease conversion to later HPSS releases.

2.8.3.5 Selecting Average Latency

This field can be used via the COS Hints mechanism to affect COS selection.

Guideline **1**: This field should generally be set to the value of the **Average Latency** field in the storage class that is at the top level in the hierarchy if data is being staged on open. If files are not being accessed multiple times after they are staged, the average latency should be set to the latency of the level they are being staged from.

Guideline 2: If most of the requests for files in this COS are read requests, then it may be best to set the value of this field equal to the **Average Latency** field in the storage class in the hierarchy where most of the data accesses will come from.

Guideline 3: If the predominant activity is write, use the **Average Latency** field from the storage class at the top level in the hierarchy.

2.8.3.6 Selecting Transfer Rate

This field can be used via the COS Hints mechanism to affect COS selection.

Guideline 1: This field should generally be set to the value of the **Transfer Rate** field in the storage class that is at the top level in the hierarchy. This should always be the case if the data is being staged on open.

Guideline **2**: If a large percentage of the reads are being done from a lower level in the hierarchy, consider setting the transfer rate based on the **Transfer Rate** associated with the storage class at this lower level.

2.8.3.7 StripeLength and StripeWidth Hints

These fields can be used via the COS Hints mechanism to affect COS selection.

Guideline 1: StripeLength and StripeWidth hints are available in the hints mechanism for the current release. When specified in the hints, StripeLength and StripeWidth from the storage class at the top level of each hierarchy are used in the COS selection algorithm.

2.8.4 File Families

Each file in HPSS is assigned a family designation. A family can be assigned to a fileset, then any file created in that fileset will belong to that family. The default family is family zero, which is interpreted by the system as meaning that the file is not associated with a family. The family designation has no effect on the placement of files on disk, but does control the tape to which the file migrates. HPSS will select a tape that has been assigned to the file's family when the file is migrated to tape. If no tapes have been assigned to the family, or all tapes assigned to the family are busy, and if other tape assignment criteria are met (e.g. max active tapes per storage class), the tape storage server will assign a blank tape to the family and the file will migrate to it.

HPSS places no restriction on the values assigned to the File Family IDs, other than zero is reserved by the system to indicate that a file has no family association. A name must be assigned to each file family.

Defining multiple file families may have an impact on system migration performance. MPS may have to mount a significantly larger number of tapes to complete a migration from a disk storage class if the files are spread across a large number of file families, compared to the number of mounts that would be required if all the files were in the same family.

2.9 HPSS Sizing Considerations

This section will help define the amount of storage that is needed for various aspects of HPSS, including storage for user files, user directories, and the additional metadata storage needed for servers to control various operations on files.

2.9.1 HPSS Storage Space

HPSS files are stored on the media that is defined to HPSS via the import and create storage server resources mechanisms provided by the Storage System Manager. You must provide enough physical storage to meet the demands of your user environment. HPSS assists you in determining the amount of space needed by providing SSM screens with information on total space and used space in all of the storage classes that you have defined. In addition, alarms can be generated automatically based on configurable threshold values to indicate when space used in a given storage class has reached some threshold level. In a hierarchy where data is being migrated from one storage class level to a lower one, management of space in the storage class provided is done via the migration and purge policies that you provide. The basic factors involved here are the total amount of media space available in the storage class being migrated and the rate at which this space is used. This will drive how the migration and purge policies are set up for the storage class. For more details on this, see Sections 2.7.1 and 2.7.2. Failure to have enough storage space to satisfy a user request results in the user receiving a NO SPACE error. One important factor in storage space growth relates to how HPSS handles write requests. The Bitfile Server always writes data to the top level in the hierarchy. When the top level is out of space, the Bitfile Server will not attempt to allocate space in a lower level and write the remaining data to the lower level.

2.9.2 HPSS Metadata Space

During the HPSS planning phase, it is important to properly assess how much disk space will be required to support the HPSS production environment. The first step in this process is to understand the various metadata files managed by each HPSS server. The sections that follow

explain the metadata files used by each HPSS server and provide hints on how to best estimate how many records will be in each file. The second step in the process is to use the metadata sizing spreadsheet, explained in Section 2.9.2.18, which helps to calculate disk space requirements based on various HPSS sizing assumptions.

2.9.2.1 Server Configuration Metadata

HPSS uses the following SFS server configuration metadata files (the default SFS file names are shown in parentheses):

- General Server Configurations (*serverconfig*)
- BFS Configurations (*bfs*)
- NS Configurations (*cns*)
- Log Client Configurations (*logclient*)
- Log Daemon Configurations (*logdaemon*)
- Metadata Monitor Configurations (*mmonitor*)
- Migration / Purge Server Configurations (*mps*)
- Mount Daemon Configurations (*mountd*)
- Mover Configurations (*mover*)
- NFS V2 Daemon Configurations (*nfs2*)
- PVL Configurations (*pvl*)
- PVR Configurations (*pvr*)
- Storage Server Configurations (*ss*)
- DMAP Gateway Configurations (*dmg*)
- Non-DCE Client Gateway Configurations (*ndcg*)

General Server Configurations. Every HPSS server (except for the SSM Data Server) must have a general configuration record describing its DCE principal, CDS server name, executable pathname, etc. SSM and the Startup Daemons use the information in this metadata file to properly start HPSS servers. HPSS servers also use the entry to determine where more specific configuration information is stored (information unique to that type of server). All other configuration metadata files described below fall into this category.

BFS Configurations. Each BFS must have an entry in the BFS configuration metadata file describing various startup / control arguments. For HPSS Release 4.1.1, only one BFS server can be defined.

NS Configurations. Each NS must have an entry in the NS configuration metadata file describing various startup / control arguments. For HPSS Release 4.1.1, only one NS can be defined.

Log Client Configurations. Each Log Client must have an entry in this configuration metadata file describing various startup / control arguments. Since a Log Client should be defined for each node that will run one or more HPSS servers (including Movers), the number of entries in this metadata file should be equal to the number of nodes used by HPSS.

Log Daemon Configurations. This metadata file describes startup / control information used by the Log Daemon. Only one Log Daemon is used in HPSS.

Metadata Monitor Configurations. Each Metadata Monitor (MMON) must have an entry in this configuration metadata file describing various startup / control arguments. The number of MMON servers will be equal to the number of Encina SFS servers planned for the HPSS environment.

Migration/Purge Server Configurations. This metadata file describes startup / control information used by the MPS Server. One or more MPS servers may be defined.

Mount Daemon Configurations. Each NFS Mount Daemon must have an entry in this configuration metadata file describing various startup / control arguments. There will be one Mount Daemon entry for each NFS server defined.

Mover Configurations. Each Mover (MVR) must have an entry in this configuration metadata file describing various startup / control arguments. The number of MVRs configured is determined by the number of nodes that have attached storage devices, plus the number of nodes used to manage network-attached storage devices.

NFS V2 Daemon Configurations. Each NFS Daemon must have an entry in this configuration metadata file describing various startup / control arguments.

PVL Configurations. Each PVL server must have an entry in this configuration metadata file describing various startup / control arguments. Currently, only one PVL server is used in a single HPSS system.

PVR Configurations. Each PVR server must have an entry in this configuration metadata file describing various startup / control arguments. Generally, one PVR will be defined for each robotic tape library used by HPSS.

Storage Server Configurations. Each Storage Server (SS) must have an entry in this configuration metadata file describing various startup / control arguments. Generally, one disk SS and one tape SS will be defined.

DMAP Gateway Configurations: This metadata file describes the startup / control information used by the DMAP Gateway. Multiple DMAP Gateway Servers may be defined.

Non-DCE Client Gateway Configurations: Each Non-DCE Client Gateway must have an entry in this configuration metadata file describing various startup / control arguments. It is possible for a single HPSS system to configure and run multiple concurrent NDCGs.

2.9.2.2 Name Server Metadata

The NS is the primary user of the following SFS metadata files (the default SFS filenames are shown in parentheses):

- Name Space Objects (*nsobjects*)
- Access Control List Extensions (*nsacls*)
- Text Extensions (*nstext*)
- Fileset Attributes (*nsfilesetattrs*)
- Global Filesets (*nsglobalfilesets*)

Name Space Objects. Each name space object (file, fileset, directory, hard link, junction or soft link) uses one record in the object file. The total number of objects permitted in the name space is limited by the number of SFS records available to the NS. The number of SFS records available to the NS should therefore be large enough to handle the projected capacity of the name space. For example, in a name space expected to consist of 50,000 directories that have 1,000,000 files distributed among them, should have at least 1,050,000 SFS records allocated to the NS. However, that would leave no space for any symbolic links, hard links, or growth. To cover these needs, the total number of SFS records might be rounded up to 1,500,000.

With this release of the NS, no provisions have been made for increasing the size of the name space by adding additional SFS files or additional Name Servers. However, if more name space is needed, additional space can be obtained by allocating more SFS records. Refer to Section 8.7.33. for information on handling an NS space shortage.

Access Control List Extensions. This metadata file is used for overflow ACL entries. An object with more than four ACL entries (in addition to the standard user, group, and other entries) will have one or more ACL records in the ACL overflow file. Each of these ACL extension records can contain up to 16 additional ACL entries. Sites that anticipate using a large number of ACLs should ensure that sufficient SFS space is available for the ACL overflow file.

Text Extensions. A name space object whose name is greater than 23 characters or that has a comment will create records in the text overflow (or extensions) file. In addition, the symbolic link data of all symbolic link objects is placed in the text overflow file. These text records are variable length with a maximum size of 1023 bytes. Sites that anticipate using comments, symbolic links and/or long names should ensure that sufficient SFS space is available to accommodate these text overflow entries.

Fileset Attributes. This metadata file is used to hold fileset information about a particular fileset. When fileset objects are created, there is insufficient room in the object file record to hold all of the needed information. This additional information is put into a record in this file. So, there is a record in this file for each fileset managed by this Name Server.

Global Filesets. This file is shared by all Name Servers running in a particular DCE cell. If more than one HPSS is running in a DCE cell, each Name Server in each HPSS system should share this file. The Global Filesets file is used to guarantee the uniqueness of fileset names and fileset identification numbers, and to identify which Name Servers and which Gateways manage which fileset. The importance of keeping these fileset names and identification numbers unique will become more apparent in future releases of HPSS.

2.9.2.3 Bitfile Server Metadata

The Bitfile Server is the primary user of the following SFS metadata files (the default SFS filenames are shown in parentheses):

- Storage Classes (*storageclass*)
- Hierarchies (*hierarchy*)
- Classes of Service (*cos*)
- Bitfiles (*bitfile*)
- Bitfile Disk Segments (*bfdisksegment*)
- Bitfile Disk Allocation Maps (*bfdiskallocrec*)
- Bitfile Tape Segments (*bftapesegment*)
- BFS Storage Segment Checkpoint (*bfsssegchkpt*)
- BFS Storage Segment Unlinks (*bfssunlink*)
- Bitfile COS Changes (*bfcoschange*)
- Bitfile Migration Records (*bfmigrrec*)
- Bitfile Purge Records (*bfpurgerec*)
- Accounting Summary Records (*acctsum*)
- Accounting Logging Records (*acctlog*)

Storage Classes. One record is created in this metadata file for each storage class that is defined. Space for 50 storage classes should generally be sufficient for metadata planning purposes.

Hierarchies. This metadata file defines the various storage hierarchies in HPSS. This number generally will equal the COS record count. Planning for 25 to 50 hierarchies should be sufficient.

Classes of Service. The class of service metadata file records the definition of each COS. Space for 25 to 50 classes of service should be sufficient for planning purposes.

Bitfiles. For each bitfile in the system, a record is created in this metadata file. The amount of space allocated for this SFS file will normally limit how many bitfiles can be created. Regardless of how many *copies* of a bitfile exist and whether the bitfile is spread across disk and/or tape, only one bitfile record is created for the file definition.

Bitfile Disk Segments. For a bitfile that is stored on disk, you will have one or more records in the bitfile disk segment file. Because bitfile disk segments simply keep track of contiguous pieces of a bitfile, there normally will be only one disk segment record needed to map the file.

To arrive at the best estimate of the number of records you need in the bitfile disk segment, bitfile tape segment, and bitfile disk map files, you need to take into account the way you have set up your classes of service and hierarchies. For example, suppose you set up a three-level hierarchy that has disk at the top level and two levels of tape. The migration policy calls for creating duplicate copies on tape. Assume that you have a 2 GB disk in the disk storage class and that the storage segment size is 512K. The maximum number of disk storage segments that could be created in this case would be 4,096. Thus you would need a maximum of 4,096 disk map records. Assume that you also

plan to store 100,000 files in this hierarchy. You will have two tape copies for each file. Assuming an average of 2 bitfile tape segment records per file, you would end up creating 400,000 bitfile tape segment records. You may not have this level of detail, but the more you know, the more accurate your estimates can be.

Bitfile Disk Allocation Maps. Also for disk files, one or more records will be created in the disk map file. The number of these records is determined by the storage segment size in the storage class in which the files are to be stored and the average file sizes to be stored in that storage class. It is generally recommended that the average number of storage segments per file (which is the average file size divided by the storage segment size) be 10 or less.

A single disk allocation map can hold up to eight storage segments. For planning purposes, you can assume two segments per disk file, on average, such that the total number of disk allocation map records is the total number of disk storage segments divided by 2. Another way to put an upper bound on the number of disk map records is as follows:

- For each disk storage class defined, determine the total amount of disk space in bytes available in the storage class.
- Divide this by the storage segment size for the storage class. This will give the maximum number of storage segments that could be created for this storage class.
- Sum the number of storage segments needed over all the storage classes. This will give you an upper bound on the number of storage segments. This value is also an upper bound on the number of disk maps, making the worst case assumption that each file uses only one storage segment. If you assume two segments per file, divide the total number of segments by 2.

Bitfile Tape Segments. For a bitfile that is a stored on tape, one or more records will be created in the bitfile tape segment file. Under normal conditions, you would expect one bitfile tape segment record for each tape on which the file is stored. A safe assumption is that for each bitfile stored on tape, you need two bitfile tape segment records.

BFS Storage Segment Checkpoint. The BFS storage segment checkpoint file is used to maintain consistency of allocated storage segments. Basically, this consistency is needed because it is necessary to allocate space in a separate Encina transaction to allow maximal storage access concurrence. This file should normally be large enough to store a few hundred records.

BFS Storage Segment Unlinks. The BFS storage segment unlink file is used to keep track of Storage Server storage segments that need to be deleted. Deletion of storage segments is done in a Bitfile Server background thread for enhanced performance. This file normally should be large enough to store a few thousand records.

Bitfile COS Changes. The COS change file keeps track of bitfiles that have each pending COS change. When the COS of a bitfile is changed, a record is created in this file. It is deleted by a background thread in the BFS when the COS change has been successfully accomplished. This usually involves copying the file data to new media. Under normal operations, this file would be large enough for a few hundred records. However, if a utility is used to change the COS on a large set of files, this file may need to be enlarged; otherwise, the utility will have to do its work in a piece-meal fashion over some period of time.

Bitfile Migration Records. The Bitfile Server creates a migration record for a file that needs to be migrated and deletes this record when the migration is completed. In the worst case scenario, this

file would contain a record for every bitfile that is stored on disk. If a site is able to estimate how many files are in an active and unmigrated state, then this file can be limited to this number.

Bitfile Purge Records. The Bitfile Server creates a purge record for each file that has data in a purgeable state. This includes files that have been migrated and files that have been staged but not overwritten. In the worst case scenario, this file would contain a record for every bitfile that is stored on disk. If a site is able to estimate how may files would be in a purgeable state, then this file can be limited to that number. Generally, a file will not have both a migration record and a purge record. Based on this, the number of records in the migration file plus the number of records in the purge file would be equal to the number of files stored on disk.

Accounting Summary Records:. This file contains accounting information for an account index, COSid and storage classes combinations. There are essentially two kinds of records: 1) if the storage class is 0, then the record is a storage summary record and contains the total number of bitfiles and bytes stored by the account index in the given COS, 2) if the storage class is not 0, this is a statistics record and contains the total number of bitfile accesses and bytes transferred associated with the account index, COSid, and referenced storage class. The number of records in this file should be the number of users in the HPSS system multiplied by the average number of levels in a hierarchy plus 1. For most configurations, the average number of levels will be 2. For sites with two copies configured, the average number of levels will be 3.

Accounting Logging Records. This file is used by the BFS to log updates to the (acctsum) records. It consists of values and flags plus the same information as defined in the (acctsum) file. The size of the file will depend upon load, but the maximum number of records for planning purposes should be 3000.

2.9.2.4 Disk Storage Server Metadata

The Disk Storage Server is the primary user of the following SFS metadata files (the default SFS filenames are shown in parentheses):

- SS Disk Storage Maps (*storagemapdisk*)
- SS Disk Storage Segments (*storagesegdisk*)
- SS Disk Physical Volumes (*sspvdisk*)
- SS Disk Virtual Volumes (*vvdisk*)

Each Disk Storage Server requires its own set of four metadata storage files: one file for disk storage maps, one for disk storage segment metadata records, one for virtual volume metadata records, and one for physical volume metadata records. If a system has more than one Disk Storage Server, each server must have its own set of files. Files must not be shared between servers.

SS Disk Storage Maps. There will be as many disk storage map records and virtual volume records as there are disk virtual volumes, and as many disk physical volume records as there are disk units, but since the number of volumes is likely to be relatively small (compared to the likely number of tape volumes), these files will probably be small. Also, since increasing or decreasing the number of disk virtual volumes is likely to be a rare event, these files will usually be a constant size. Even for relatively large disk systems, the consumption of SFS storage resources by disk storage server map, virtual volume, and physical volume metadata will be small.

SS Disk Storage Segments. The number of disk storage segment metadata records is likely to be larger than any of the other disk storage server metadata types, but it too is bounded. In the maximum case, each disk virtual volume (VV) can support 16,384 disk storage segments, each with its own metadata record. The upper bound for a disk storage segment metadata file is 16,384 times the number of VVs.

If the default storage segment size for the storage class is a multiple of the VV block size (and it probably will be), the maximum number of disk storage segments is reduced by that factor. For example, if the length of disk storage segments in a particular storage class is 4 VV blocks, the maximum number of storage segments that can exist on such a VV is 16,384 divided by 4 or 4,096. The actual maximum number in this case is 4097, because the first VV block on each VV is reserved for volume label information.

Expect both the disk storage segment metadata file and the disk storage map metadata file to be quite volatile. As files are added to HPSS, disk storage segments will be created, and as files are migrated to tape and purged from disk, they will be deleted. If SFS storage is available on a selection of devices, the disk storage segment metadata file and the storage disk map file would be good candidates for placement on the fastest device of suitable size.

SS Disk Physical Volumes. The disk PV metadata file describes each disk partition or logical volume imported into HPSS. There will be at least one partition for each disk device managed by HPSS, and there is one record in this SFS file for each partition.

SS Disk Virtual Volumes. The disk VV metadata file describes all disk virtual volumes created by this Storage Server. Each VV is described by a separate record in this file.

2.9.2.5 Tape Storage Server Metadata

The Tape Storage Server is the primary user of the following SFS metadata files (the default SFS filenames are shown in parentheses):

- SS Tape Storage Maps (*storagemaptape*)
- SS Tape Storage Segments (*storagesegtape*)
- SS Tape Physical Volumes (*sspvtape*)
- SS Tape Virtual Volumes (*vvtape*)

Each Tape Storage Server requires its own set of four metadata storage files: one file for tape storage maps, one for tape storage segment metadata records, one for virtual volume metadata records and one for physical volume metadata records. If a system has more than one Tape Storage Server, each server must have its own set of files. Files must not be shared between servers.

SS Tape Storage Maps. This metadata file is used to track the allocation of storage on tape virtual volumes. The number of tape storage map records should be equal to the number of tape virtual volumes plus the number of tape storage classes because a special map record is used to track statistics for each tape storage class.

SS Tape Storage Segments. The tape storage segment metadata file may become large. Storage segments are created by the server to describe contiguous *chunks* of data written on tapes. Some segments may be long but others may be short. The number of storage segments found on a given tape is not limited by the server. This number is a function of the length of the files written on the tape, the VV block size, the size of the data buffer used to write the tape, and other factors.

While the size of the tape metadata files may become large as the tape system grows, the volatility of the tape metadata files is low. PV and VV metadata records are created and deleted only when tapes are added to and removed from the system. These records are modified only when the associated tapes are read or written, so most records will remain unchanged for long periods of time. Most storage segment and storage map metadata records are created and/or modified only as the associated tapes are written. The volatility of these files is a function of the rate at which files are written to tape, but compared to a Disk Storage Server, a Tape Storage Server's metadata files are relatively static. Therefore, larger, possibly slower, disk systems would be good candidates to assign to SFS for the storage of Tape Storage Server metadata files.

SS Tape Physical Volumes. The tape PV metadata file describes each tape physical volume imported into HPSS. The number of records in this file will therefore equal the total number of tape cartridges that will be managed by this Storage Server.

SS Tape Virtual Volumes. The tape VV metadata file describes all tape virtual volumes created by this Storage Server. Each VV is described by a separate record in this file. The number of VV records should be approximately equal to the number of tape physical volumes divided by the average stripe width of the managed tape PVs.

2.9.2.6 PVL Metadata

The PVL is the primary user of the following SFS metadata files (the default SFS filenames are shown in parentheses):

- PVL Drives (*pvldrive*)
- PVL Physical Volumes (*pvlpv*)
- PVL Jobs (*pvljob*)
- PVL Activities (*pvlactivity*)

PVL Drives. Each record in the PVL drive metadata file describes a single drive managed by the PVL. The number of PVL drives will be equal to the total number of Mover disk and tape devices.

PVL Physical Volumes. Every physical volume, whether disk or tape, must be imported into HPSS through the PVL. Each import operation creates a new record in this metadata file describing the PV. The number of records in the file will equal the total number of disk physical volumes that will be defined as well as the total number of tape cartridges that will be imported.

PVL Jobs. A single PVL job involves one or more PVL activities. For example, when reading from a 4-way tape volume, a single PVL job would be generated consisting of four PVL activities (one for each cartridge mount request). Depending on how many concurrent I/O requests are allowed to flow through the Tape Storage Server and PVL at any given time, this metadata file should not grow beyond a few hundred records. However, a PVL job is also created for each disk physical volume, so the total number of disk physical volumes should be added to this number.

PVL Activities. This metadata file stores individual PVL activity requests such as individual tape mounts. Depending on how many concurrent I/O requests are allowed to flow through the Storage Server and PVL at any given time, this metadata file should not grow beyond a few hundred records.

2.9.2.7 PVR Metadata

The PVR is the primary user of one SFS metadata file (the default SFS filenames are shown in parentheses):

- Cartridges (*cartridge_3494* for 3494 PVR, *cartridge_3495* for 3495 PVR, *cartridge_stk* for STK PVR, *cartridge_aml* for AML PVR, *cartridge_operator* for Operator PVR)

Cartridges. The cartridge metadata file contains a single record for each cartridge managed by the PVR. Therefore, the number of records will equal the number of tape cartridges injected into this PVR. If multiple PVRs are used, each PVR should use a separate SFS file to store cartridge metadata.

2.9.2.8 Mover Metadata

The Mover is the primary user of one SFS metadata file (the default SFS filename is shown in parentheses):

- Mover Devices (*moverdevice*)

Mover Devices. This metadata file contains a single record for each Mover device that is defined, including both disk and tape devices. All Mover devices should be defined in a single SFS file.

2.9.2.9 Migration/Purge Server Metadata

The MPS Server is the primary user of the following SFS metadata files (the default SFS filenames are shown in parentheses):

- Migration Policies (*migpolicy*)
- Purge Policies (*purgepolicy*)
- Migration / Purge Checkpoints (*mpchkpt*)

Migration Policies. Each migration policy creates a single record in this metadata file. The maximum number of migration policies will be equal to the total number of storage classes defined.

Purge Policies. Each purge policy creates a single record in this metadata file. The maximum number of purge policies will be equal to the total number of storage classes defined.

Migration/Purge Checkpoints. For each storage class supported by an MPS, the server requires two records in the MPS SFS checkpoint file to keep track of where to start the next migration or purge run.

The MPS Server shares the following SFS metadata files with the Bitfile Server (the default SFS filenames are shown in parentheses). These files are configured via the BFS configuration screens:

- Migration Record File (*bfmigrrec*)
- Purge Record File (*bfpurgerec*)

Migration Record File. When the Use Migration Record File option is enabled, BFS writes a migration record into this metadata file each time a bitfile on disk becomes a migration candidate. MPS can then identify disk migration candidates from these records instead of sequentially reading the bitfile metadata for each bitfile in the storage class.

Purge Record File. When the Use Purge Record File option is enabled, BFS writes a purge record into this metadata file each time a bitfile on disk becomes a purge candidate. MPS can then identify disk purge candidates from these records instead of sequentially reading the bitfile metadata for each bitfile in the storage class.

2.9.2.10 Logging Services Metadata

The Log Clients and Daemon manage one SFS metadata file (the default SFS filename is shown in parentheses):

- Log Policies (*logpolicy*)

Log Policies. This metadata file contains logging policy information for each HPSS server. The maximum number of records will equal the maximum number of HPSS servers that will be defined.

2.9.2.11 Metadata Monitor Metadata

Metadata Monitor servers use no other metadata files beyond the standard server configuration files.

2.9.2.12 Storage System Management Metadata

The SSM System Manager is the primary user of the following SFS metadata files (the default SFS filenames are shown in parentheses):

- File Family (*filefamily*)

In addition, the SSM System Manager requires an entry in the generic server configuration file. The SSM Data Server does not require an entry in any SFS file.

File Families. The file families metadata file defines the available groups to which files can be assigned so that they are stored on tape volumes with other files in the same group. Space for 25 to 50 families should be sufficient for planning purposes.

2.9.2.13 NFS and Mount Daemons Metadata

The NFS and Mount Daemons use no other metadata files beyond the standard server configuration files.

2.9.2.14 DMAP Gateway Metadata

The DMAP Gateway is the primary user of one SFS metadata file (the default SFS filename is shown in parentheses):

- DMAP Gateway Filesets (*dmgfileset*)

DMAP Gateway Filesets. This metadata file contains an entry for each DFS fileset a DMAP Gateway is managing. For planning purpose, most sites should consider no more than 1000 records per (**dmgfileset**) file. Each DMAP Gateway Server must define its own unique DMAP Fileset Filename.

2.9.2.15 Location Server Metadata

The Location Server is the primary user of the following SFS metadata file:

- Location Policy (lspolicy)

Location Policy. This metadata file describes the startup / control information used by the Location Server. Only one Location Policy can be defined.

2.9.2.16 Non-DCE Client Gateway

The Non-DCE Client Gateway uses no other metadata files beyond the standard server configuration files.

2.9.2.17 Metadata Constraints

The generic configurations for all HPSS servers must be contained in a single SFS file. SSM makes this happen transparently. Also, all server-specific configuration entries for a given server type can be in the same SFS file. For example, all Storage Servers should be defined in the single SFS file named *ss*. The following metadata files should not be shared between servers; instead, each server should use a distinct metadata SFS filename:

- MPS Checkpoints
- PVR Cartridges
- SS Disk Storage Maps
- SS Tape Storage Maps
- SS Disk Storage Segments
- SS Tape Storage Segments
- SS Disk Physical Volumes
- SS Tape Physical Volumes
- SS Disk Virtual Volumes
- SS Tape Virtual Volumes

2.9.2.18 Metadata Sizing Spreadsheet

To help with projecting disk space requirements for HPSS metadata, an Excel 5.0 spreadsheet is available in **/usr/lpp/hpss/tools/sfs/mdsizing.xls**. The spreadsheet is divided into two worksheets: the first defines the various HPSS and SFS sizing assumptions; the second calculates the sizing estimates.

Metadata Sizing Assumptions

The metadata sizing assumptions are organized by HPSS dynamic variables, HPSS static configuration variables, and SFS sizing assumptions. Note that all assumptions are with respect to sizing the *maximum* amount of metadata required to support the various assumptions.

HPSS Dynamic Variables. Table 2-4 defines the HPSS dynamic variables, detailing how large the HPSS production environment will be. These variables therefore significantly impact the metadata sizing estimates.

Table 2-4 HPSS Dynamic Variables

Variable	Description
Max Total Bitfiles	The maximum number of bitfiles that will exist in HPSS. The spreadsheet also considers this value to also be the total number of bitfiles on tape, since it is assumed that every HPSS bitfile will eventually migrate to tape. If it is expected for 2 million files to be placed in HPSS, enter 2,000,000. This value significantly impacts the overall metadata sizing estimate.
Max Bitfiles on Disk	Used to determine how many bitfiles might reside on disk at any point in time. This number must be less than or equal to *Max Total Bitfiles*. For example, if disk space is sufficient to hold / cache 20,000 of the 2 million total files, enter 20,000. This value significantly impacts the overall metadata sizing estimate.
Avg. Copies Per Bitfile	Defines the average number of copies per bitfile (since HPSS supports duplicate bitfile copies). For example, if there is approximately 1 extra bitfile copy for every 100 bitfiles created, enter a value of 1.01. This value significantly impacts the overall metadata sizing estimate. If not n using duplicate bitfile copies, use a value of 1.0.
Percent of Extra Copies Stored on Tape	When duplicate bitfile copies are created, the extra copy(s) are written to a lower level in the storage hierarchy. Usually, lower levels are defined as tape storage classes. Enter the average percentage of these extra bitfile copies that will be stored on tape (versus another disk storage class). This value significantly impacts the overall metadata sizing estimate if the duplicate copy feature is used.
Total Number of Users	Defines the total number of users in the HPSS system. This value is used in conjunction with “Avg. Number of Levels Per Hierarchy” to define the size of accounting records.
Avg. Number of Levels Per Hierarchy	This value is the average number of levels defined in the hierarchies. For most hierarchies which are defined as disk and tape, this value would be 2.
Max Accounting Log Records	This value is the maximum number of log records that is expected at any given time when the BFS is updating the (acctsum) file. The number of records will depend upon system load, but should typically not exceed 3000.
Avg. Name Space Objects Per Bitfile	One name-space object is created for each HPSS bitfile; however, other name space objects, such as directories and links, are also created. This value defines, on average, how many name space entries there are per HPSS bitfile. For example, if there is one additional name-space object created for each 20 bitfiles (directory or link) on average, a value of 1.05 would be entered. This value significantly impacts the overall metadata sizing estimate.

Table 2-4 HPSS Dynamic Variables (Continued)

Variable	Description
Avg. ACL Overflows Per Name Space Object	A name-space object record can store up to 4 access control list entries (in additional to the standard user, group, other, and foreign permissions). If the object has more than four ACL entries, an ACL overflow record must be created. Each ACL overflow record can hold up to 16 additional ACL entries. This value defines, on average, how many ACL overflow records will be created per name-space object. For example, if 1 in 100 name-space objects will have between 5 and 20 ACL entries, a value of .01 would be entered.
Avg. Text Overflows Per Name Space Object	A name-space object record can store a filename that is 23 characters long (the base name, not the full pathname). If a filename is longer than 23 characters, a text overflow record must be generated. Also, if a comment is attached to a name-space object, a text overflow record must also be created. This value defines, on average, how many text overflow records will be created per name-space object. For example, if 5 in 100 name-space objects will either have a filename longer than 23 characters or a comment attached, a value of .05 would be entered.
Filesets per Name Server	The number of filesets a Name Server will be managing.
Global Number of Filesets	The total number of filesets that will managed by the Name Servers in a given DCE Cell. (This will have more meaning when multiple Name Servers are supported).
Total Number of DMAP Gateways	This defines the total number of DMAP Gateway Servers that will be defined in the system.
Max Filesets Per DMAP Gateway	This value is the maximum number of filesets that a DMAP Gateway can be expected to manage. This value can have an impact on the overall storage sizing estimate.
Avg. Bitfile Segments Per Tape Bitfile	The average number of bitfile segments created for each tape bitfile. When files are migrated to tape from disk, there normally will be only one tape bitfile segment for the file. However, if the file is partially modified, additional bitfile segments will be migrated to tape. This value significantly impacts the overall metadata sizing estimate.
Avg. Storage Segments Per Disk VV	The average number of storage segments that will be created per disk virtual volume. Since there are a maximum of 16,384 blocks per disk VV, and each storage segment uses one or more VV blocks, the number must be less than or equal to 16,384. This value can be determined by dividing the average disk VV size by the average storage segment size.
Avg. Storage Segments Per Disk Alloc Rec	Each disk allocation record stores information for up to 8 disk storage segments. If, on average, 1 extra disk allocation record is required per 5 disk bitfiles, a value of 1.2 would be entered.
Avg. Storage Segments Per Tape Bitfile	The average number of storage segments per tape bitfile. If files are only written to tape through migration, the value can be 1, assuming the storage class characteristics have been set up properly. If users are allowed to write directly to tape, the average should be something higher than 1.
Max BFS Storage Segment Checkpoints	The maximum number of storage segment checkpoint records created by BFS. A value of 500 should be sufficient for planning purposes.
Max Queued Bitfiles Changing COS	The maximum number of bitfiles that have a pending class of service change. A value of 1,000 should be sufficient for planning purposes.
Max Queued BFS Storage Segment Unlinks	The maximum number of storage segments that could be queued to be unlinked. A value of 1,000 should be sufficient for planning purposes.

Table 2-4 HPSS Dynamic Variables (Continued)

Variable	Description
Max Disk Bitfiles Queued for Migration	The maximum number of unmigrated bitfiles that may exist on disk at one time. The value should generally equal *Max Bitfiles on Disk*.
Max Disk Bitfiles Queued for Purging	The maximum number of migrated bitfiles that may exist on disk at one time. The value should generally equal *Max Bitfiles on Disk*.
Max Queued PVL Jobs	The maximum number of jobs that might be queued in the PVL at any one time.
Avg. Activities Per PVL Job	Each PVL job has one or more activities associated with it (mount tape #1, mount tape #2, etc. for a 2-way tape stripe). This variable defines the average number of PVL activities per PVL job, which should represent the average stripe width for tape virtual volumes.
Total Disk Physical Volumes	The maximum total number of disk physical volumes.
Total Disk Virtual Volumes	The total number of disk virtual volumes that will be created, which is equal to the number of disk physical volumes divided by the average disk stripe width.
Total PVR #1 Tape Cartridges	The spreadsheet allows up to three PVR's to be defined, each with a different number of cartridges. This field represents the maximum number of cartridges that will be managed by PVR #1.
Total PVR #2 Tape Cartridges	If a second PVR will be used, enter the maximum number of cartridges managed by the second PVR; otherwise, use zero.
Total PVR #3 Tape Cartridges	If a third PVR will be used, enter the maximum number of cartridges managed by the third PVR; otherwise, use zero.
Total Tape Virtual Volumes	The total number of tape virtual volumes that will be created, which is equal to the total number of all PVR cartridges divided by the average tape stripe width.

HPSS Static Configuration Variables. Table 2-5 defines the HPSS static configuration variables, which have more of a static nature in that they are not particularly relevant to how large the HPSS system will be. They relate more to how many HPSS servers, storage classes, etc. will be defined and therefore are not significant in the overall metadata size projection.

Table 2-5 HPSS Static Configuration Values

Variable	Description
Total Storage Classes	The maximum number of storage classes that will be defined.
Total Storage Hierarchies	The maximum number of storage hierarchies that will be defined.
Total Storage Families	The maximum number of file families that will be defined.
Total Classes of Service	The maximum number of classes of service that will be defined, which typically will equal the total number of storage hierarchies created.

Table 2-5 HPSS Static Configuration Values (Continued)

Variable	Description
Total Tape Drives	The total number of tape drives used by HPSS.
Total HPSS Servers	The overall number of HPSS servers that will be defined. This includes the Name Server, BFS, Storage Servers, Log Daemon, Log Clients, Startup Daemons, Movers, etc.
Total BFS Servers	The total number of BFS servers that will be configured. This value should be 1.
Total Name Servers	The total number of Name Servers that will be created. This value should be 1.
Total Log Clients	The total number of Log Clients that will be used, which will be equal to the total number of nodes running any type of HPSS server.
Total Log Daemons	The total number of Log Daemons. This value should be 1.
Total Metadata Monitor Servers	The total number of Metadata Monitor servers, which should equal the total number of Encina SFS servers used by the HPSS system.
Total Movers	The total number of Movers that will be created.
Total NFS Mount Daemons	The total number of NFS Mount Daemons that will be configured.
Total NFS Servers	The total number of NFS servers that will be configured.
Total PVR Servers	The total number of PVR servers that will be used.
Total Storage Servers	The total number of Storage Servers. Normally, sites will have at least one disk Storage Server and at least one tape Storage Server.
Total Migration / Purge Servers	The total number of Migration / Purge Servers. Normally, sites will have at least one MPS server.

SFS Sizing Assumptions. The only assumption specific to SFS that should be reviewed is the *Leaf Page Load Factor* field. This field defines, on average, how "full" each SFS leaf page will be. A value of 50% is overly conservative since SFS will likely utilize each leaf page more than this. A value over 80% would be too optimistic and would likely cause insufficient disk space to be allocated to metadata. A value between 60 to 75% is probably a good compromise.

Sizing Computations

The second worksheet in the metadata sizing spreadsheet calculates the projected total number of records for each metadata file and the corresponding amount of required disk space, based on the assumptions previously entered. The spreadsheet allows the record count for any given metadata file to be manually overridden, if desired, and allows the required disk space to be allocated to one of 10 SFS data volumes, allowing individual SFS data volumes to be properly sized.

The various columns in this worksheet are described below.

Subsystem/Metadata File—This column lists all HPSS metadata files, preceded by the acronym for the subsystem that manages the file or is the primary user of the file. Due to performance gains provided by SFS and DCE when the SFS server and client are on the same machine, it is generally advisable to allocate all metadata files primarily used by a given HPSS server with the SFS server running on that same machine.

Total Records—This column shows the projected number of metadata records for each metadata file, given the assumptions from the assumption worksheet. The formulas for computing the number of records are shown below. Note that the variable names on the left match the metadata file names listed in the *Subsystem/Metadata File* column while variables on the right of the equals sign ("=") represent values from the assumptions worksheet (unless otherwise noted).

ACCT / Accounting Policies (accounting) = 1

ACCT/Log Records (acctlog) = Max Accounting Log Records

*ACCT/Summary Records (acctsum) = Total Number of Users * (Avg Number of*

Levels Per Hierarchy + 1)

*ACCT/Snapshot Records (acctsnap) = Total Number of Users * (Avg Number*

of Levels Per Hierarchy + 1)

BFS/Bitfiles (bitfile) = Max Total Bitfiles

BFS/Change COS Records (bfcoschange) = Max Queue Bitfiles Changes COS

BFS/Classes of Service (cos) = Total Classes of Service

BFS/Disk Alloc Records (bfdiskallocrec) = [Computed] Disk Bitfile

Segments/Avg. Storage Segments Per Disk Alloc Rec

BFS/Disk Bitfile Segments (bfdisksegment) = Avg. Storage Segments Per

*Disk VV * Total Disk Virtual Volumes*

BFS/Server Configs (bfs) = Total BFS Servers

BFS/Storage Classes (storageclass) = Total Storage Classes

BFS/Storage File Families (filefamily) = Total Storage Families

BFS/Storage Hierarchies (hierarchy) = Total Storage Hierarchies

BFS/Storage Segment Checkpoint (bfssegchkpt) = Max BFS Storage Segment Checkpoints

BFS/Tape Bitfile Segments (bftapesegment) = Avg Bitfile Segments Per

*Tape Bitfile * (Max Total Bitfiles + (Max Total Bitfiles **

*(Avg Copies Per Bitfile - 1) * Percent of Extra Copies Stored on Tape))*

BFS/Unlink Records (bfssunlink) = Max Queued BFS Storage Segment Unlinks

*NS/ACL Extensions (nsacls) = [Computed] NS Objects * Avg. ACL Overflows*

Per Name Space Object

NS/Fileset Attributes (nsfilesetattrs) = Filesets Per Name Server

NS/Global Filesets (nsglobalfilesets) = Global Number of Filesets

*NS/Objects (nsobjects) = Max Total Bitfiles * Avg Name Space Objects Per Bitfile*

NS/Server Configs (ns) = Total Name Servers

*NS/Text Extensions (nstext) = [Computed] NS Objects * Avg Text Overflows Per Name Space Object*

DMAP Gateway Configurations (dmg) = Total Number of DMAP Gateways

*DMAP Gateway Filesets (dmgfileset) = Total Number of DMAP Gateways * Max Filesets Per DMAP Gateway*

LOC/Location Server Policies (lspolicy) = 1

LOG/Log Client Server Configs (logclient) = Total Log Clients

LOG/Log Daemon Server Configs (logdaemon) = Total Log Daemons

LOG/Log Policies (logpolicy) = Total HPSS Servers

MMON/Servers (mmonitor) = Total Metadata Monitor Servers

MOVR/Devices (moverdevice) = Total Disk Physical Volumes + Total Tape Devices

MOVR/Server Configs (mover) = Total Movers

MPS/Bitfile Migration Records (bfmigrrec) = Max Disk Bitfiles Queued for Migration

MPS/Bitfile Purge Records (bfpurgerec) = Max Disk Bitfile Queued for Purging

*MPS/Checkpoints (mpchkpt) = 2 * Total Storage Classes*

MPS/Migration Policies (migpolicy) = Total Storage Classes

MPS/Purge Policies (purgepolicy) = Total Storage Classes

MPS/Server Configs (mps) = Total Migration/Purge Servers

NFS/Mount Daemons (mountd) = Total NFS Mount Daemons

NFS/Server Configs (nfs2) = Total NFS Servers

*PVL/Activities (pvlactivity) = Max Queued PVL Jobs * Avg Activities Per PVL Job*

PVL/Drives (pvldrive) = Total Disk Physical Volumes + Total Tape Drives

PVL/Jobs (pvljob) = Max Queued PVL Jobs

PVL/Physical Volumes (pvlpv) = Total Disk Physical Volumes + Total PVR

#1 Tape Cartridges + Total PVR #2 Tape Cartridges +

Total PVR #3 Tape Cartridges

PVL/Server Configs (pvl) = 1

PVR/Cartridges #1 (cartridge) = Total PVR #1 Tape Cartridges

PVR/Cartridges #2 (cartridge) = Total PVR #2 Tape Cartridges

PVR/Cartridges #3 (cartridge) = Total PVR #3 Tape Cartridges

PVR/Server Configs (pvr) = Total PVR Servers

Remote Site Configs (site) = Total Remote Sites

SS/Disk Maps (storagemapdisk) = Total Disk Virtual Volumes

SS/Disk Physical Volumes (sspvdisk) = Total Disk Physical Volumes

SS/Disk Storage Segments (storagesegdisk) = Avg Storage Segments Per

*Disk VV * Total Disk Virtual Volumes*

SS/Disk Virtual Volumes (vvdisk) = Total Disk Virtual Volumes

SS/Server Configs (ss) = Total Storage Servers

SS/Tape Maps (storagemaptape) = SS Tape Virtual Volumes + Total Storage Classes

SS/Tape Physical Volumes (sspvtape) = Total PVR

#1 Tape Cartridges + Total PVR #2 Tape Cartridges +

Total PVR #3 Tape Cartridges

SS/Tape Storage Segments (storagesegtape) = Avg Storage Segments Per

*Tape Bitfile * (Max Total Bitfiles + (Max Total Bitfiles **

*(Avg Copies Per Bitfile - 1) * Percent of Extra Copies Stored on Tape))*

SS/Tape Virtual Volumes (vvtape) = Total Tape Virtual Volumes

SSM/Server Configs (serverconfig) = Total HPSS Servers

Record Count Override—This column allows the computed *Total Records* value to be overridden, if desired. The value entered in this column must be greater than zero (0) in order to be used. Normally this column should remain blank.

Primary Data/Index—This section is composed of two columns. The first column, labeled *Size (MBs)*, shows the computed disk space requirement for the number of *Total Records* computed (or the *Record Count Override*, if set). This takes into account the actual size of the data in each record as well as the primary index. The second column, labeled *SFS Vol #*, specifies which logical SFS data volume to use in allocating this disk space. An integer value from 1 to 10 can be entered. For example, if four AIX logical volumes is allocated to SFS, use values from 1 to 4 to assign each metadata file to the appropriate volume. The *Space Allocation Per Encina Volume* table, located on the same worksheet, will use this allocation to calculate total disk space requirements per SFS volume. By varying the SFS volume number, the administrator can experiment with how to best allocate the metadata files based on the logical volume sizes.

It is important to note that calculating the size of an SFS balanced-tree file is not easy and is sometimes inaccurate. The file sizes do not grow linearly according to the number of records because of the changing height of the B-tree and the fact that leaf pages may not be full. In this spreadsheet, the maximum height of the B-tree is calculated based on the order of the B-tree and the number of records. In other words, the spreadsheet assumes a worst case, which is appropriate when planning disk space requirements.

For more information on how the size of SFS B-trees can be calculated, refer to *Encina Overview, an IBM ITSO Redbook* (GG24-2512-00).

Secondary Index 1—This section shows the disk space requirement for the first secondary index associated with each metadata file, if such an index is used. The *SFS Vol #* column allows this disk space to be allocated to a particular logical SFS volume. However, the **managesfs** utility used to create HPSS metadata files currently creates the file and all associated indices on the same SFS volume. So normally, the SFS volume columns in the spreadsheet will contain the same logical SFS volume number for a given metadata file row.

If, for performance or disk-space limitation reasons, a secondary index is to be created on a different SFS volume from the primary index, use **managesfs** to first create the metadata file specifying the SFS volume desired for the primary data/index, manually delete the secondary index using the **sfsadmin** command, and attempt to recreate the metadata file using **managesfs** while specifying the SFS volume desired for the secondary index. The second attempt to create the metadata file will yield an error indicating that the metadata file already exists; however, it will go ahead and recreate the secondary index on the different SFS volume.

Secondary Index 2—This section shows the disk space requirement for the second secondary index associated with each metadata file, if such an index is used.

Secondary Index 3—This section shows the disk space requirement for the third secondary index associated with each metadata file, if such an index is used.

Total Size (MBs)—This column calculates, in megabytes, the sum of all disk space used for the primary data and index, as well as all secondary indices.

Space Allocation Per Encina Volume—Continuing to the right on this worksheet, is another table that summarizes the disk space allocation per logical SFS volume per metadata file. These calculations are based on the *SFS Vol* numbers entered in the previous table. At the bottom of this table are the overall totals for each SFS volume, in megabytes.

2.9.2.19 Encina SFS Disk Space

This section explains the disk space requirements of Encina, part of which comes from completing the metadata sizing spreadsheet. The disk space used by Encina SFS falls into the following categories:

- Encina Transaction log
- Encina Media Recovery Archive Files
- Encina SFS Data
- Encina SFS Backup Files

Encina Transaction Log. The Encina transaction log is a wrap-around log used to store transaction information. It records all changes in SFS data on a per-transaction basis. In the event of a system or server crash, upon restart the SFS server reads the transaction log to determine what, if any, transactions were in progress but not committed and properly aborts all data changes associated with that transaction. Likewise, it properly updates all SFS data for transactions that had been committed but not written to disk in the SFS data volumes.

The optimal type of disk used for the transaction log is a low-latency disk since the I/O is performed in many small chunks. To ensure full recoverability in the event of a media failure on the transaction log disk, it is important to mirror the transaction log on a separate physical disk.

The size of the transaction log is dependent on the volume of HPSS transactions. Although the Encina administrative documentation gives some guidelines for sizing the transaction log, HPSS testing has typically been done using either 128 MB or 256 MB.

Since I/O delays associated with the transaction log can affect HPSS performance, it is recommended that few, if any, other logical volumes be placed on the same disks used for the transaction log. For example, it would be unwise to use the **rootvg** volume group for the transaction log or put the transaction log on the same disk where paging space has been created.

Note that there is at least one transaction log per Encina SFS server. Although SFS allows multiple log volumes (transaction logs) to be defined for a single SFS server, the HPSS installation scripts assume that only one log volume is created per SFS server.

Encina Media Recovery Archive Files. Because media failures may occur between SFS backups, it is important to be able to replay all SFS transactions from the last SFS backup up to the point of a media failure. Encina "media archiving" is the feature that enables this level of recovery. When media archiving is enabled, a running backup of the transaction log is written to disk. The transaction log wraps around, but media archive files do not. The administrator can then move these media archive files to offline storage for safe keeping. If media archiving has not been enabled prior to a media failure, the SFS data cannot be restored.

Because media failures can occur where these backup files are written, mirroring can be considered for improved reliability. It is recommended that at least 2 GB of disk space be available for the media recovery archive files.

Media archive files are written to the directory **/opt/encinalocal/encina/sfs/hpss/archives**. Before Encina is configured as described in Section 4.7, either appropriate disk space should be added to **/opt** or a separate file system should be created (e.g., **/archive**) and an appropriate link created from the directory mentioned above. For example, if a separate file system was created called **`/archive`**

to hold SFS backup data and the log archive files will be placed in **/archive/encina**, then create the following link:

```
% mkdir /archive/encina
% ln -s /archive/encina /opt/encinalocal/encina/sfs/hpss/archives
```

The HPSS utility **backman** can be used to move log archive files to tape and delete them from disk.

Encina SFS Data. Actual HPSS metadata is stored in SFS data volumes. SFS data volumes store the actual SFS record data as well as associated index and B-tree overhead information. SFS must have sufficient disk space allocated for its data volumes in order to store the projected amount of HPSS metadata.

Calculating the amount of disk space required to support an HPSS system of a given size is quite complicated. This is due in part to how HPSS is implemented internally and also by the way in which SFS balanced-trees are implemented. To assist in projecting metadata disk space requirements, a Microsoft Excel spreadsheet has been developed and can be found in **/usr/lpp/hpss/tools/sfs/mdsizing.xls**. Section 2.9.2.18 explains how to use this spreadsheet.

Encina SFS Backup Files. SFS has the ability to generate on-line backup files while continuing to service HPSS metadata requests. However, the backups can only be generated by issuing the **tkadmin backup lvol** command. It is important to generate regular backups of SFS in case of future disk failures.

Details on performing SFS backups can be found in the *Encina Overview, an ITSO Redbook* (GG24-2512-00) and the *Encina Structured File Server Administrator's Guide and Reference for AIX*.

An HPSS menu-based utility called **backman** can be used to assist in creating SFS backups and moving them to offline tape. One option in **backman** is to generate a specified number of incremental SFS data-volume backup files. This should be done on a routine basis (i.e., daily). Refer to Appendix I for additional details on the **backman** utility.

SFS backups must be performed separately on each SFS server used by the HPSS environment.

2.9.3 System Memory and Disk Space

2.9.3.1 Disk Space Requirements for HPSS Installation

The estimated disk space requirement for the HPSS installation node is as follows:

- HPSS Source: 50 MB
- HPSS Binary: 170MB
- HPSS Libraries: 20MB

In addition, temporary disk space in the **/usr/lpp/hpss/tarfile** directory is required on the installation node to perform the HPSS subsystem remote installation. The estimated temporary disk space requirement for the installation node is approximately 85MB.

For each HPSS remote node, only the binary of the selected subsystems will be installed. The disk space requirement for each node can be calculated by adding the disk space required for the

subsystems to be remotely installed. Table 2-6 lists the disk space and the temporary disk space requirements for each HPSS subsystem. Note that the Infrastructure component includes the HPSS entities that should be available on each node. The Infrastructure component includes the Accounting server, Log Client, Metadata Monitor, Startup Daemon, HPSS Utilities, and HPSS Message Catalog. The Infrastructure component will be installed automatically to a remote node during the HPSS remote installation process.

Table 2-6 HPSS Remote Installation Disk Space Requirements

Component	/usr/lpp/hpss (MB)	/usr/lpp/hpss/tarfile (MB)
BFS	4.0	2.0
DMG	13.0	5.5
HDM	15.0	5.5
LOGD	5.5	2.5
LS	2.5	1.0
MPS	2.5	1.0
MVR	8.5	4.0
NS	2.5	1.0
PVL	2.0	1.0
PVR	7.0	3.0
FTP	4.5	2.0
NFS	4.0	1.5
INST	0.5	0.5
Infrastructure	66.0	30.0
SS	5.0	2.0
SSM	10.0	6.0
NDCG	5.0	2.5
NDC_AIX	16.0	6.0
NDC_SGI	20.0	8.0

2.9.3.2 Disk Space Requirements for Running HPSS Servers

Some of the Unix configuration files required by the HPSS servers and other files generated during their executions are usually placed in the **/var/hpss** directory by default. There should be sufficient disk space available to accommodate these files. The following subsections describe the space requirements needed for running the individual HPSS servers:

Disk Space Requirements for HPSS Files

The **/var/hpss/etc** is the default directory where some of the additional Unix configuration files are placed. These files are typically very small.

The **/var/hpss/tmp** is the default directory where the Startup Daemon creates a lock file for each of the HPSS servers it brought up in the node. These lock files are typically very small.

Disk Space Requirements for Running Log Daemon

The Log Daemon will create two central log files in the directory specified in its configuration, usually **/var/hpss**. The size of these log files are also specified in its configuration, usually 5 megabytes each.

Disk Space Requirements for Running Log Client

Each Log Client will create one local log file in the directory specified in its configuration, usually **/var/hpss**. The size of the local log file is also specified in its configuration, usually 5 megabytes.

Disk Space Requirements for Running MPS

If the migration / purge report is configured, the MPS will generate a migration / purge report every 24 hours. The size of these reports varies depending on the number of files being migrated / purge during the report cycle, These reports should be configured to be written into the **/var/hpss/mps** directory and should be removed when no longer needed.

Disk Space Requirements for Running Accounting Report

If Accounting report is requested. a report and a checkpoint files are created in the directory specified in the Accounting Policy, usually **/var/hpss/acct**. The sizes of these files are depending the number of files stored in HPSS.

Disk Space Requirements for Running NFS Daemon

The HPSS NFS server memory and disk space requirements are largely determined by the configuration of the NFS request processing, attribute cache, and data cache. Data cache memory requirements can be estimated by multiplying the data cache buffer size by the number of memory data cache buffers.

Attribute cache memory requirements can be estimated by combining requirements for directory name with file attribute memory requirements. The total number of name space objects kept in the NFS attribute cache is determined by the maximum number of entries in the attribute cache least recently used (LRU) list. A portion of these entries will be directories, and the rest will be bitfiles, symbolic links, or hard links.

Directory name memory requirements are determined by multiplying the estimated number of cached directories by the directory size configuration. Estimating the number of cached directories can be done by multiplying the number of entries in the attribute cache LRU list by the percentage of directories to files at the site. For example, if the LRU list maximum is set to 100 and the

percentage of directories at the site is 1 directory to 10 files (10%), the estimated number of cached directories is 10.

File attribute memory requirements are determined by subtracting the number of cached directories from the maximum number of entries on the LRU list and multiplying the result by 100.

Request processing memory requirements can be estimated by multiplying the number of ONC remote procedure call threads by 65 KB.

The HPSS NFS Server requires disk storage for five UNIX files:

- Exports file—a text file that specifies how NFS access is offered.
- Remote mount (**rmtab**) file—this file, in the same directory as the exports file, is a text file that identifies what clients have mounted HPSS directories.
- Credentials map file—a text file that is used to checkpoint the NFS credentials map. The credential map file size will be based on the number of entries in the credentials map cache.
- Checkpoint file—required by the data cache. The data cache checkpoint file size is related to the number of cache entries, but is much smaller than the cache entry file.
- Entry file—required by the data cache. Disk storage for the data cache entry file is determined by multiplying the number of cache entries by the data cache buffer size. The data cache entry file should be placed in its own disk partition to avoid disk contention.

Disk Space Requirements for Running SSM

If SSM is configured to buffer alarm and event messages in a disk file, each SSM Data Server process will require an alarm file approximately 5MB in size. This file may be placed wherever it is convenient. There is normally only one Data Server process.

2.9.3.3 System Memory and Paging Space Requirements

Specific memory and disk space requirements for the nodes on which the HPSS servers will execute will be influenced by the configuration of the servers—both as to which nodes the servers will run on and the amount of concurrent access they are set up to handle.

For nodes that will run one or more of the HPSS servers (and most likely an SFS server), excluding the Movers, it is recommended that at least 512 MB of memory be configured. However, more memory will certainly be required for systems that run most of the servers on one node and / or are supporting many concurrent users. The availability of memory to the HPSS servers and SFS servers will be critical to providing acceptable response time to many end user operations. Disk space requirements are primarily covered by Section 2.9.2, for SFS space, and the preceding subsections under Section 2.9.3 for the individual HPSS servers. In addition, sufficient disk space should be available for paging space based on the recommendations made in the system documentation for the amount of memory configured.

For nodes that will only run Movers and no SFS server, the amount of memory is dependent on the number and types of devices configured on those nodes, the expected usage of those devices, and

the configuration of the Movers on those nodes. In general, Movers supporting a disk device will require more memory than Movers supporting tape devices. This is because many outstanding requests are likely for a disk device, but there is usually only one outstanding request for a tape device. The size of the buffers used internally by the Mover will have an impact on the Mover memory requirements because each Mover request process will use two buffers to handle I/O requests. It is recommended that at least 256 MB of memory be configured for these nodes. However, more memory will certainly be required for nodes that are supporting many devices, especially disk devices that will support a large number of concurrent end-user requests.

2.10 HPSS Performance Considerations

This section provides some additional hints for enhancing HPSS performance:

2.10.1 DCE

Due to several problems observed by HPSS, it is highly recommended that all DCE client implementations use the **RPC_SUPPORTED_PROTSEQS=ncadg_ip_udp** environment variable. Frequent timeouts may be observed if this is not done.

Each HPSS system should be periodically checked for invalid/obsolete endpoints. Failure to comply may cause miscellaneous failures to occur as well as significantly decreasing HPSS performance.

The AIX DCE implementations provides the ability to initiate the Global Directory Access Daemon with a "**–r filename**" flag allowing the specification of a specific "**resolv.conf**" file to be used by **gdad**. **gdad** is necessary to implement "Cross Cell" features. This option may not be supported on all platforms.

2.10.2 Encina

Because all HPSS transactions update metadata information, it essential that the SFS sub-system be optimized to access the database in an efficient manner. There are a number of items that can greatly affect performance of the SFS server and its access to the metadata. The following is a list of these that should be analyzed:

- Number of SFS Threads
- Buffer Pool Size
- SMP Node/Machine
- HPSS/SFS Server Proximity
- Storage Media

2.10.2.1 SFS Server Parameters

Number of SFS Threads. The SFS server allows the administrator to change the number of threads the server uses to carry out user requests. For those using a script like "/etc/rc.encina" to start the SFS manually, the command line should include an argument to override the default value of 10 processing threads. A more reasonable setting for SFS in an HPSS environment is 200 to 350 processing threads. The argument on the SFS command line would be "-P 200:2". For those sites using Enconsole, the GUI provides an option to change the "Thread Pool Size". The server should be recycled to start using the new setting. The increase in the number processing threads will allow HPSS to carry out more simultaneous operations. The default of 10 threads will cause HPSS to serialize many of its transaction operations which will decrease performance.

Buffer Pool Size. As with the number of SFS threads, the SFS allows the administrator to define how much memory the server is allowed to use to carry out its operations. The default value is 1000 1k pages of memory which is too small for most HPSS installations. A value of 16000 or 32000 is more reasonable and can be changed on the SFS command line by adding a "-b 16000" to the arguments or using Enconsole to change the value. Be sure to update the "/etc/security/ limits" file (and reboot) to allow the system to handle the added processing size that both this and the increase in SFS threads will create. In the default section, the values for "data" should be increase to "data = 524288" and for "rss" should be increased to "rss = 262144". These are recommended values from Transarc.

2.10.3 Workstation Configurations

SMP Node/Machine. SFS can easily become CPU bound in highly utilized HPSS systems. HPSS transaction speeds can be limited by the processing capacity of the CPU/Node where the SFS server resides. It is recommended that the placement of the SFS server be on a machine with multiple processors (H50 or Silver Node SP) rather than on a high-end single processor machine (R24).

HPSS/SFS Server Proximity. The placement of HPSS Servers with their corresponding SFS Server can affect the performance of the HPSS system. Some HPSS Servers which are SFS intensive (Name Server) are best served with SFS being local. The following HPSS servers should be co-located with the SFS server on the same machine/node:

- Name Server
- BFS Server
- Disk & Tape Storage Server

Other HPSS servers are not so metadata intensive and can be distributed in the configuration without as much concern.

Storage Media. Allocating storage for the log and data volume(s) should be carefully planned. The log and data volumes need to be on separate sets of disk not only for data integrity in case of a media failure but also for performance. In addition, the log archive files being written to the JFS filesystems should be backed by physical media separate of both the log and data volumes physical storage. For those sites with a high number of anticipated transactions, the distribution of the metadata files across multiple data volumes should correspond with backing them with separate physical media and I/O adapters to prevent any interference. Ideally, the configuration of the SFS

storage should try to distribute the disk activity across as many I/O adapters and hard drives as possible.

When determining what configuration is best, keep in mind that some devices may behave better in a mirrored configuration rather than setup as RAID. Performance tests to monitor data and transactions rates of a given set of disks and I/O adapters should be made before permanently configuring the resources for SFS.

2.10.4 Bypassing Potential Bottlenecks

HPSS performance is influenced by many factors, such as device and network speeds, configuration and power of HPSS server machines, Encina SFS server configuration, storage resource configuration, and client data access behavior.

HPSS provides mechanisms to bypass potential bottlenecks in the performance of data transfers, given that the system configuration provides the additional resources necessary. For example, if the performance of a single disk device is the limiting factor in a transfer between HPSS and a client application, a number of the disks can be grouped together in a striped Storage Class to allow each disk to transfer data in parallel to achieve improved data transfer rates. If after forming the stripe group, the I/O or processor bandwidth of a single machine becomes the limiting factor, the devices can be distributed among a number of machines, alleviating the limitation of a single machine.

If the client machine or single network between the client and HPSS becomes the limiting factor, HPSS supports transferring data to or from multiple client machines, potentially using multiple physical networks, in parallel, to bypass those potential bottlenecks.

During system planning, consideration should be given to the number and data rates of the device, machine I/O bandwidth, network bandwidth, and client machine bandwidth to attempt to determine a configuration that will maximize HPSS performance given an anticipated client work load.

2.10.5 Configuration

The configuration of the HPSS storage resources (see Section 2.7.6) is also an important factor in overall HPSS performance, as well as how well the configuration of those resources matches the client data access patterns.

For example, if a site provides access to standard FTP clients and allows those clients to write data directly to tape, the buffer size used by the FTP server and the virtual volume block size defined for the Storage Class being written to will have a significant impact. If the buffer size used by the FTP server is not a multiple of the virtual volume block size, each buffer written will result in a distinct storage segment on the tape. This will cause additional metadata to be stored in the system and extra synchronization processing of the tape. (If the buffer size is a multiple of the virtual volume block size, each write will continue to append to the same storage segment as the previous write. This will continue until the final write for the file, which will usually end the segment, thus reducing the amount of metadata generated and media processing.)

2.10.6 FTP/PFTP

Data transfers performed using the standard FTP interface are primarily affected by the buffer size used by the FTP Daemon. The buffer size can be configured as described in Section 5.8.3. It should be a multiple of the storage segment size, if possible. If not, it should be, at least, a multiple of the virtual volume block size. If the buffer size is too small, the FTP Daemon will need to issue a large number of individual read or write requests; however, if the buffer size is too large, the FTP Daemon will require a large amount of memory, which may cause additional paging activity on the system.

The size of the FTP Daemon buffer is extremely important if the FTP clients write files directly to a tape storage class, as described in Section 2.10.5.

Parallel FTP (PFTP) can be used to move data using either TCP/IP or IPI-3 over HiPPI. The default option, to use TCP/IP, is supported on all platforms. If both the Mover nodes for the Movers managing the storage location of the data within HPSS and the client nodes support IPI-3 over HiPPI, that option can also be selected. The IPI-3 over HiPPI transfer option generally provides a higher performance transfer with less CPU utilization on the Mover and client nodes.

Note that using the PFTP data transfer commands (e.g., **pput** and **pget**) are not influenced by the FTP Daemon buffer size because the data flows directly between the client and Movers.

Note that PFTP clients that use the standard FTP data transfer commands (e.g., **put** and **get**) have the same performance considerations as standard FTP clients as described in Section 2.10.6.

Parallel transfers move the data between the Mover and the end-client processes bypassing the HPSS FTPD. Customers should be educated to use the parallel functions rather than the non-parallel functions. NOTE: ASCII transfers are not supported by the parallel functions and the non-parallel functions will need to be specified for ASCII transfers. ASCII transfers are NOT typically required; but the end-customer should familiarize themselves with the particulars.

Parallel transfers should be optimized so that the Class of Service (COS), Media Stripe Widths, Network Stripe Widths, and Parallel Block Sizes are consistent with each other. E.g., using a Network Stripe Width of 4 with a Media Width of 2 may result in poorer performance than if both specifications are equal! Specifying a Network Stripe Width of 4 where there is only one network interface may not provide any improvement over a lower Network Stripe Width (2) if the bandwidth of the network is (over-)filled by a single stripe.

Non-parallel transfers occur via a “stage and forward” approach (Device <==> Mover <==> HPSS FTP Daemon <==> FTP Client.) It is recommended that the “–h” option be specified on the hpss_pftpd and that the “hpss_option HOST hostname” be specified in the ftpaccess file if the FTP Daemon system has multiple interfaces. The hostname should refer to the highest speed interface available for transmission of data between the FTP Daemon and HPSS Movers. NOTE: this interface MUST be available to all client systems that contact the FTP Daemon.

Where reasonable, the standard FTP ports should be modified to something other than 20/21 on the system acting as the HPSS FTP Daemon. The HPSS FTP Daemon should be set to use the 20/21 ports by default. This reduces the problem of requiring the end-customer from needing to know which port to use for transfers to HPSS. In conjunction with this, it is highly recommended that the {ftpbanner} file be used with an appropriate message to provide information to the end-customer that they are accessing HPSS as opposed to a standard system.

2.10.7 Client API

The Client API provides the capability to perform data transfer of any size (the size being parameters supplied by the client to the read and write interfaces). The size of the data transfers can have a significant impact on the performance of HPSS. In general, larger transfers will generate less overhead than a series of smaller transfers for the same total amount of data.

If the Client API handles the actual data transfer for a request (as opposed to just handling the control of the request while the client handles the data transfer), an environment variable can be used to select the data transfer mechanism to be used. The current choices are TCP / IP, which is the default and supported on all platforms, or IPI-3 over HiPPI, which requires that the Mover nodes and client nodes support IPI-3 over HiPPI. Using IPI-3 over HiPPI, especially for large requests, generally provides a higher performance transfer with less CPU utilization on the Mover and client nodes.

The size of the transfers is extremely important because the clients may write files directly to a tape storage class, as described in Section 2.10.5.

2.10.8 Name Server

For listing directory operations, the number of bytes that can be returned is bounded in two ways: a client specifies the maximum number of bytes that it can accept, and, in addition, the Name Server has an upper limit on the number of bytes it will return. The upper limit for the Name Server must be carefully selected. A size that is too small can degrade performance. During system testing, a value of 64 KB has been used.

Other items that can cause minor performance degradations in the Name Server are processing path names with a large number of components, listing directories containing entries with long (greater than 23 characters) names, and servicing requests for objects which have a large (greater than 7) number of ACL entries.

2.10.9 Location Server

The Location Policy defined for a site generally determines how the Location Server will perform and how it will impact the rest of the HPSS system. View the help for the fields on this screen to determine if the values need to be changed. The default policy values are adequate for the majority of sites. Usually, the only time the policy values need to be altered is when there is unusual HPSS setup.

The Location Server itself will give warning when a problem is occurring by posting alarms to SSM. Obtain the information for the Location Server alarms listed in the *HPSS Error Manual*. To get a better view of an alarm in its context, view the Location Server's statistics screen.

If the Location Server consistently reports a heavy load condition, increase the number of request threads and recycle the Location Server. Remember to increase the number of threads on the Location Server's basic server configuration screen as well. If this doesn't help, consider replicating the Location Server on a different machine. Note that a heavy load on the Location Server should be a very rare occurrence.

If timeout errors occurred while attempting to contact a Bitfile Server , verify if the server is down. If the timeouts occurred while BFS is running, it may help by increasing the **COS/BFS Timeout** value.

2.10.10 Logging

Excessive logging by the HPSS servers can degrade the overall performance of HPSS. If this is the case, it may be desirable to limit the message types that are being logged by particular servers. The Logging Policy can be updated to control which message types are logged. By default, Trace messages are not logged. Message types to consider disabling are Debug and Request messages. Other message types can also be disabled. Once the Logging Policy is updated for one or more HPSS servers, the Log Clients associated with those servers must be reinitialized.

2.10.11 MPI-IO API

MPI-IO client applications must be aware of HPSS Client API performance on certain kinds of data transfers (see Section 2.10.7), which is basically that HPSS is optimized for transferring large, contiguous blocks of data.

The MPI-IO interface allows for many kinds of transfers that will not perform well over the HPSS file system. In particular, MPI-IO's use of filetypes enables specification of scatter-gather operations on files, but when performed as noncollective reads and writes, these discontiguous accesses will result in suboptimal performance in the best case, and in failure to complete the access, if excessive file fragmentation results, in the worst case. Collective I/O operations may be able to minimize or eliminate performance problems that would result from equivalent noncollective I/O operations by coalescing discontiguous accesses into a contiguous one.

An MPI-IO application should make use of HPSS environment variables (see Section 5.8.1) and file hints at open time to secure the best match of HPSS resources to a given task. MPI-IO can make use of either TCP/IP or IPI-3 over HiPPI connections, as supported by HPSS on a given platform. If available, the use of IPI-3 transfers will give the best performance for large transfers.

2.10.12 Cross Cell

"Cross Cell" Trust should be established with the minimal reasonable set of cooperating partners (N-squared problem.) Excessive numbers of "Cross Cell" connections may diminish Security and may cause performance problems due to Wide Area Network delays. The communications path between cooperating cells should be reliable.

2.10.13 DFS

DFS performance for HPSS is dependent on a number of factors: fileset type (mirrored or archived), CPU performance, memory throughput rates, DFS client caching, etc. Mirrored filesets will perform at HPSS rates for name space changes. Name space accesses will perform at normal DFS rates. Archived filesets will perform close to normal DFS rates for both name space changes and accesses. Data access and changes will perform at DFS rates if the data is resident, but will be delayed if HPSS must stage the data onto the Episode disks.

When setting up aggregates it is suggested that the fragment size be set to 1024 and the blocksize be set to 8192. These are the defaults and have been tested much more thoroughly than other settings. An important thing to remember is that any file smaller than the blocksize can not currently be purged from the Episode disk, and setting the blocksize larger than 8192 may caused

space and resource problems on the disk. (This is a limitation of the Episode DMAPI on which the HPSS DFS code is implemented).

During testing the biggest gains were realized by altering the DFS client caching. A large memory cache instead of a disk cache may improve performance dramatically if the client machine can afford the loss of memory for client caching buffers. For more information on DFS configuration for AIX please refer to the DCE 2.2 document "*Distributed File Service Administration Guide and Reference*".

Since HPSS must read DFS anodes to determine which files to migrate or purge, it is suggested that aggregates be sized so that no more than 250,000 files and directories are present. This will allow the migration and purge algorithms to determine what to do in a reasonable amount of time. (This number is based on tests run on several machines and may be raised or lowered based on the throughput rates of the disks and the performance of the CPU). With current Episode limitations on the amount of space allowed for anodes per aggregate and the design of the HPSS DFS code, the maximum number of anodes per aggregate at this time on an aggregate managed by HPSS will be around 1,000,000 files (2GB / 2K per migrated file). When planning the system, assume that it may not always be possible to grow an aggregate to get more files. It may be necessary to add a new aggregate. In fact, since migration and purging is based per aggregate, the system may perform better with more aggregates than just a few.

2.11 HPSS Metadata Backup Considerations

The following is a set of "rules" associated with backing up HPSS metadata. Some of these rules are covered in the following sections. Tools like **backman**/**sfsbackup** as well as site specific generated scripts and procedures can be used to backup and protect the SFS metadata. It is important that each site review this list of "rules" and check to insure that their site's backup is consistent with these policies.

2.11.1 Rules for Backing Up SFS Log Volume and MRA Files

- Media archiving must always be enabled while HPSS is running in production mode. If it is ever temporarily disabled (e.g., during a one-time UniTree migration), then before HPSS is placed back into production mode, media archiving must be re-enabled and a complete backup of all SFS data volumes must be made using TRB files.
- The SFS log volume must be mirrored on at least two separate physical disks or be stored on a redundant RAID device.
- The file system used to store MRA files must be mirrored on at least two separate physical disks or be stored on a redundant RAID device.
- Separate disks must be used to store the SFS log volume versus the file system used to store MRA files.
- The latest (i.e., most current) MRA file must never be manipulated, moved, copied, or deleted since SFS may still be actively writing to it.
- MRA files (except the latest) must be copied to tape at least once per day.

- MRA files (except the latest) must be copied to at least two different tapes.

2.11.2 Rules for Backing Up SFS Data Volumes and TRB Files

- SFS data volumes must be mirrored on at least two separate physical disks or be stored on a redundant RAID device.
- The file system used to store TRB (i.e., data volume backup) files must be mirrored on at least two separate physical disks or be stored on a redundant RAID device.
- Separate disks must be used to store the SFS data volumes versus the file system used to store TRB files.
- Separate disks must be used to store the SFS log volume versus the SFS data volumes.
- A sufficient number of TRB files must be generated such that a complete backup of each SFS data volume is made at least every 5 days (recommend using a TRB file size such that it takes no more than 20 files to cover the entire data volume).
- TRB files must be copied to tape as soon as they are generated.
- TRB files must be copied to at least two different tapes.
- A complete set of TRB files must be generated (i.e., a complete SFS data volume backup) prior to running HPSS in production mode.
- A data dump (dd) of all SFS log and data volumes should be made at least once per month while HPSS and SFS are not running and should be copied to at least two different tapes.

2.11.3 Miscellaneous Rules for Backing Up HPSS Metadata

- Once a month, a site must tell SFS to truncate old backups and retain backup information for at least the latest 10 complete backups.
- At least one image of the tape backups must be moved offsite at a frequency that meets the site's disaster recovery objectives. For example, if a site always wants to be able to recover to the previous day in case of a disaster, then tapes must be moved offsite each day.
- The Encina SFS configuration information must be backed up to at least two separate tapes after SFS is initially configured and immediately after the SFS configuration changes (i.e., a data volume is added, deleted, or expanded). It must also be backed up at least once per week.
- The DCE servers information (e.g., the security registry and CDS name space) must be backed up to at least two separate tapes immediately after HPSS is configured and then at least once per week.
- If configured, the HPSS PFTP configuration files must be backed up at least once per week.
- If configured, the HPSS NFS configuration files must be backed up at least once per week.

- HPSS must not be used to read / write tapes used to store backup files (Encina, DCE, etc.).

Chapter 3

HPSS Installation

3.1 Overview

This chapter provides instructions and supporting information for installing the HPSS software from the HPSS distribution media.

To install this system, we recommend that the administrator be familiar with UNIX commands and configuration, be familiar with a UNIX text editor, and have some experience with the C language and shell scripts.

Note: For information on upgrading from a previous version of HPSS, please see Appendix M.

3.1.1 Installation Roadmap

The steps required to install the HPSS system are listed below. Each step is discussed in more detail in the section referenced.

1. Create owner account for HPSS files (Section 3.2).
2. Install HPSS on the installation node (Section 3.3).
3. Verify HPSS installed files (Section 3.4.1).

3.2 Create Owner Account for HPSS Files

The HPSS software must be installed by a **root** user. In addition, an AIX User ID of **hpss** and Group ID of **hpss** is required for the HPSS installation process to assign the appropriate ownership for the HPSS files. If the **hpss** User ID does not exist, the installation process will fail. The **hpss** User ID will be later used as the default User ID under which the majority of the HPSS servers will run.

If the **hpss** User ID and **hpss** Group ID do not exist, create them using the normal AIX procedures.

It is very important that the HPSS files' permissions are set up properly. If they are not, the HPSS may not work properly after it is configured. We recommend that the HPSS files' ownerships and permissions set by the installation process be preserved. If they must be changed, care must be taken to ensure they are changed correctly. Refer to 3.4.1 for more information on the HPSS file ownerships and permissions.

3.3 Install HPSS on the Installation Node

For the current release, the HPSS distribution media is available on one of the following media:

- 8mm tape
- **installp** image file (network)

Regardless of the format of the distribution media, HPSS can be installed using the **installp** command or the AIX **SMIT** utility.

Install HPSS Using the installp Command

Issue the **installp** command as follows to install HPSS:

installp -aQqX -d <input device/directory> all

Install HPSS Using the AIX 4.2/4.3 SMIT Utility

The HPSS software is installed in the same manner as other AIX optional or licensed program products. A detailed description can be found in the *AIX Version 4.x for RISC System/6000 Installation Guide* (SC23-2550-03), "Installing Optional Software and Service Updates" document.

Perform the following steps to install the HPSS distribution media:

1. Complete the HPSS Installation Worksheet (Section C.2) from Appendix C.

2. Log on to the installation node as **root** user. Enter **smitty install** and select the following options in the order shown:

   Install and Update Software
   Install and Update from LATEST Available Software

3. Enter the INPUT device / directory for software information using the media information from the worksheet (See Step 1).

4. Set the following parameter values on the screen:

INPUT device / directory for software	**<data entered from step 3>**
SOFTWARE to install	**[hpss]**
PREVIEW only? (install operation will NOT occur)	**no**
COMMIT software updates?	**no**
SAVE replaced files?	**yes**
AUTOMATICALLY install requisite software?	**no**
EXTEND file systems if space needed?	**yes**
OVERWRITE same or newer versions?	**no**
VERIFY install and check file sizes?	**no**
Include corresponding LANGUAGE filesets?	**yes**
DETAILED output?	**no**
Process multiple volumes?	**yes**

5. Press **Enter** on the dialog box to begin installing the software. (Press **F3** to return to Step 3.)

The content of the HPSS distribution media is installed in the **/usr/lpp/hpss** directory on the installation node. If one or more HPSS subsystems need to be installed onto other nodes, the HPSS Subsystem Remote Install capability described in Section 4.6.5 can be used during the HPSS infrastructure configuration phase.

3.4 Post Installation Procedures

3.4.1 Verify HPSS Installed Files

1. When the HPSS installation has completed successfully, verify that the following HPSS file structures are created:

   /usr/lpp/hpss/bin/ <HPSS binaries>

   /usr/lpp/hpss/lib/ <HPSS libraries>

   /usr/lpp/hpss/include/<HPSS include, idl, and tidl files>

   /usr/lpp/hpss/msg/ <HPSS message catalog>

   /usr/lpp/hpss/tools/ <HPSS tools>

   /usr/lpp/hpss/man/ <HPSS man pages>

   /usr/lpp/hpss/config/<HPSS configuration scripts>

   /usr/lpp/hpss/stk/ <STK files>

   /usr/lpp/hpss/sammi/ hpss_ssm/<HPSS Sammi files>

   /usr/lpp/hpss/src/<HPSS source files - if source code is installed>

2. In addition, verify that the HPSS file ownerships and file permissions set by the installation process are preserved as follows:

   Binary files: rwxr-xr-x bin bin

   Source files: r--r----- bin bin

   Sammi files: r--r----- bin bin

After the HPSS Infrastructure Configuration is performed (as described in Chapter 4), the permissions of the files in the following directory are changed by the **mkhpss** script as follows:

/usr/lpp/hpss/sammi/ hpss_ssm: rw-rw----hpss hpss

The HPSS file permissions can be further restricted according to the site's policy. However, this should be done carefully so that the new permissions will not cause problems for the HPSS. Special care should be exercised when changing the permission for the Sammi files. A number of Sammi data files require read / write access by the SSM user in order for Sammi to run correctly, even though it may not be obvious that a user needs write access to a particular file. The Sammi files as installed by the HPSS installation procedure should allow read / write access to all configured SSM

users. When changing the permissions set by the installation process, ensure that the new permissions will not disable part or all of the configured SSM sessions.

3.4.2 Remake HPSS

The HPSS distribution package includes the binaries required by HPSS and no additional "make" is needed. However, due to special circumstances, it may be necessary for a site to rebuild the binaries. In this case, the following steps must be performed to rebuild the HPSS binaries:

- Login as **root**.
- cd to **/usr/lpp/hpss** directory**.**
- Review the **/usr/lpp/hpss/Makefile.macros** file. This file defines the "**make**" environments and options for the HPSS software compilation. Ensure that the environments and options are specified properly before running the "**make**" command.
- Issue "**make clean**" command.
- Issue "**make undepend**" command.
- Issue "**make idl-tidl**" command.
- Issue "**make depend**" command.
- Issue "**make**" command.

3.4.3 Set Up Sammi License Key

If Sammi is to be executed from an HPSS node, the Sammi software must be installed. The Sammi license key provided by Kinesix must be set up before Sammi can be invoked. The Sammi license key can be set up as follows:

1. Change directory to **<Sammi installation directory>/bin**:

   cd <Sammi installation directory>/bin

2. Execute the sammi_license utility:

   sammi_license -w

The user will be prompted for the host name where the SSM session will run and the Sammi license key. Verify that the **<Sammi installation directory>/bin/.s2_license.<hostname>** file is created with the read / write permissions for all SSM users.

Chapter 4

HPSS Infrastructure Configuration

4.1 Overview

This chapter provides instructions and supporting information for the infrastructure configuration of HPSS.

Before configuring the HPSS infrastructure, we recommend that the administrator be familiar with UNIX commands and configuration, be familiar with a UNIX text editor, and have some experience with the C language, shell scripts, DCE, and Encina.

The procedures described in this chapter assume that DCE, Encina, Sammi and HPSS have been successfully installed. In addition, a DCE / DFS cell has been fully configured.

4.1.1 Infrastructure Road Map

The steps required to configure the HPSS infrastructure are listed below. Each step is discussed in more detail in the section referenced.

1. Verify the prerequisite software (Section 4.2).
2. Define the HPSS environment variables (Section 4.3).
3. Configure the HPSS infrastructure on the installation node (Section 4.4).
4. Configure the HPSS infrastructure on each remote node (Section 4.5).

4.2 Verify the Prerequisite Software

Verify that the following prerequisites have been performed in preparation for the HPSS infrastructure configuration:

- Install DCE / DFS 2.1 and 2.2 (if DFS is to be used)

- Install Encina TxSeries 4.2 software
- Install Sammi 4.1.2 software
- Install HPSS 4.1.1 software
- Configure DCE / DFS server / client machine(s)

Refer to the *DCE Version 2.1 for the AIX Administration Guide* and the *Encina Server Administration: System Administrator's Guide and Reference for AIX* for more information on installing and configuring DCE and Encina.

4.3 Define the HPSS Environment Variables

The HPSS environment variables are defined in two files: The **/usr/lpp/hpss/config/hpss_env.default** file defines environment variables that may need to be changed by the administrator. The **/usr/lpp/hpss/include/hpss_env_defs.h** file defines environment variables that most likely do not need to be modified. Before running the **mkhpss** script on the installation node (and subsequently on each remote node), copy the **hpss_env.default** file to **hpss_env.** Review and edit it to ensure that the variables reflect the local environment. To override an environment variable defined in the **hpss_env_defs.h** file, redefine the environment variable in the **hpss_env** file. Table 4-1 lists and describes all the environment variables defined in these two files. The variables defined in the **hpss_env.default** file are indicated in bold print.

The utility **/usr/lpp/hpss/config/hpss_set_env** is provided to help managing the HPSS environment variables.

Usage: hpss_set_env [-all] [-def] [-set] | ENVNAME

-all - Show current HPSS environment values

-def - Show all HPSS default environment values

-set - Set HPSS default environment values

ENVNAME - Display current value for ENVNAME

The hpss_env file on the installation node is copied to the remote node during the HPSS subsystem remote installation process.

Table 4-1 HPSS Environment Variables

Variable	Description	Default Value
HPSS_PATH	Pathname of HPSS top level	${HPSS_ROOT:-/usr/lpp/hpss}
HPSS_PATH_SAMMI_INSTALL	Pathname where Sammi is installed	/usr/lpp/sammi
HPSS_SYSTEM	System platform name	$(uname)

Table 4-1 HPSS Environment Variables (Continued)

Variable	Description	Default Value
HPSS_SYSTEM_VERSION	System platform version	$(oslevel)
HPSS_HOST	Host name	$(hostname)
HPSS_HOST_FULL_NAME	Fully qualified host domain name	"host $HPSS_HOST |cut -f1 -d"
HPSS_CELL_ADMIN	Principal name for administrating HPSS in a cell	cell_admin
HPSS_CDS_PREFIX	CDS prefix for HPSS servers	/.:/hpss
HPSS_CDS_HOST	CDS host name	$HPSS_HOST
HPSS_SFS_ADMIN	Principal name for administrating HPSS in Encina SFS	encina_admin
HPSS_SFS_SERVER	Encina SFS server name with CDS prefix	/.:/encina/sfs/hpss
HPSS_SFS_VOL	Encina SFS internal data volume name	sfsVol$HPSS_SFS
HPSSLOG	HPSS logging output destination [stdout | NULL]	NULL
HPSS_ROOT	Root pathname for HPSS Unix top level	/usr/lpp/hpss
HPSS_HOST	Machine host name	%H
HPSS_KEYTAB_FILE_SERVER	Fully qualified DCE HPSS server keytab file	/krb5/hpss.keytabs
HPSS_KEYTAB_FILE_CLIENT	Fully qualified DCE HPSS client keytab file	/krb5/hpssclient.keytab
HPSS_PATH	Pathname for HPSS Unix top level	${HPSS_ROOT}
HPSS_PATH_BIN	Pathname for /bin	/bin
HPSS_PATH_SLASH_BIN	Pathname for /bin	/bin
HPSS_PATH_SLASH_ETC	Pathname for /etc	/etc
HPSS_PATH_USR_BIN	Pathname for /usr/bin	/usr/bin
HPSS_PATH_USR_SBIN	Pathname for /usr/sbin	/usr/sbin
HPSS_PATH_VAR	Pathname where HPSS HDM logs and configuration files are placed	/var /hpss/hdm
HPSS_USER	HPSS user name	hpss
HPSS_USERROOT	Root user ID	root

Table 4-1 HPSS Environment Variables (Continued)

Variable	Description	Default Value
HPSS_PATH_SAMMI_BIN	Pathname for Sammi bin	$HPSS_PATH_SAMMI_INSTALL/bin
HPSS_PATH_SAMMI_DATA	Pathname for Sammi data	$HPSS_PATH_SAMMI_INSTALL/data
HPSS_GRP_NAME	HPSS group name	hpss
HPSS_GRP_NAME_SERVER	Server group name	hpss_server
HPSS_GRP_NAME_CLIENT	Client group name	hpss_client
HPSS_PRINCIPAL	DCE Principal name for HSEC Server	NULL
HPSS_PRINCIPAL_BFS	DCE Principal name for Bitfile Server	hpss_bfs
HPSS_PRINCIPAL_SS	DCE Principal name for Storage Server	hpss_ss
HPSS_PRINCIPAL_MVR	DCE Principal name for Mover	hpss_mvr
HPSS_PRINCIPAL_PVR	DCE Principal name for PVR	hpss_pvr
HPSS_PRINCIPAL_PVL	DCE Principal name for PVL	hpss_pvl
HPSS_PRINCIPAL_NDCG	DCE Principal name for Non-DCE Gateway	hpss_ndcg
HPSS_PRINCIPAL_NS	DCE Principal name for Name Server	hpss_cns
HPSS_PRINCIPAL_SSM	DCE Principal name for SSM	hpss_ssm
HPSS_PRINCIPAL_LOG	DCE Principal name for Log Client and Daemon	hpss_log
HPSS_PRINCIPAL_MM	DCE Principal name for Metadata Monitor	hpss_mm
HPSS_PRINCIPAL_FTPD	DCE Principal name for FTP Daemon	hpss_ftp
HPSS_PRINCIPAL_PFSD	DCE Principal name for PFS Daemon	hpss_piofsie
HPSS_PRINCIPAL_HPSSD	DCE Principal name for Startup Daemon	hpss_hpssd
HPSS_PRINCIPAL_MOUNTD	DCE Principal name for Mount Daemon	hpss_mountd
HPSS_PRINCIPAL_NFSD	DCE Principal name for NFS Daemon	hpss_nfs2
HPSS_PRINCIPAL_MPS	DCE Principal name for Migration/Purge Server	hpss_mps

Table 4-1 HPSS Environment Variables (Continued)

Variable	Description	Default Value
HPSS_PRINCIPAL_DMG	DCE Principal name for DMAP Gateway	hpss_dmg
HPSS_PRINCIPAL_LS	DCE Principal name for Location Server	hpss_ls
HPSS_PRINCIPAL_CLIENT_API	DCE Principal name for Client API	hpss_client_api
HPSS_PRINCIPAL_BFS_UID	Principal ID for BFS	“”
HPSS_PRINCIPAL_SS_UID	Principal ID for SS	“”
HPSS_PRINCIPAL_MVR_UID	Principal ID for MVR	“”
HPSS_PRINCIPAL_PVR_UID	Principal ID for PVR	“”
HPSS_PRINCIPAL_PVL_UID	Principal ID for PVL	“”
HPSS_PRINCIPAL_NDCG_UID	Principal ID for Non-DCE Gateway	“”
HPSS_PRINCIPAL_NS_UID	Principal ID for NS	“”
HPSS_PRINCIPAL_SSM_UID	Principal ID for SSM	“”
HPSS_PRINCIPAL_LOG_UID	Principal ID for Logging	“”
HPSS_PRINCIPAL_MM_UID	Principal ID for Metadata Monitor	“”
HPSS_PRINCIPAL_FTPD_UID	Principal ID for FTP Daemon	“”
HPSS_PRINCIPAL_PFSD_UID	Principal ID for PIOFS Import/ Export Daemon	“”
HPSS_PRINCIPAL_HPSSD_UID	Principal ID for Startup Daemon	“”
HPSS_PRINCIPAL_MOUNTD_UID	Principal ID for Mount Daemon	“”
HPSS_PRINCIPAL_NFSD_UID	Principal ID for HPSS NFS Daemon	“”
HPSS_PRINCIPAL_MPS_UID	Principal ID for MPS	“”
HPSS_PRINCIPAL_DMG_UID	Principal ID for MPS	“”
HPSS_PRINCIPAL_LS_UID	Principal ID for MPS	“”
HPSS_PRINCIPAL_CLIENT_API_UID	Principal ID for Client API	“”
HPSS_EXEC_ACCT	executable name for Accounting	${HPSS_PATH_BIN}/hpss_acct
HPSS_EXEC_BFS	executable name for Bitfile Server	${HPSS_PATH_BIN}/hpss_bfs
HPSS_EXEC_DMG	executable name for DMAP Gateway	${HPSS_PATH_BIN}/ hpss_dmg
HPSS_EXEC_FTPD	executable name for FTPD	${HPSS_PATH_BIN}/hpss_ftp
HPSS_EXEC_HPSSD	executable name for Start Daemon	${HPSS_PATH_BIN}/hpssd

Table 4-1 HPSS Environment Variables (Continued)

Variable	Description	Default Value
HPSS_EXEC_LOGC	executable name for Log Client	${HPSS_PATH_BIN}/hpss_logc
HPSS_EXEC_LOGD	executable name for Log Daemon	${HPSS_PATH_BIN}/hpss_logd
HPSS_EXEC_LS	executable name for Location Server	${HPSS_PATH_BIN}/hpss_ls
HPSS_EXEC_MM	executable name for Metadata Monitor	${HPSS_PATH_BIN}/hpss_mmon
HPSS_EXEC_MOUNTD	executable name for Mount Daemon	${HPSS_PATH_BIN}/hpss_mnt1
HPSS_EXEC_MPS	executable name for Migration/Purge Server	${HPSS_PATH_BIN}/hpss_mps
HPSS_EXEC_MVR	executable name for Mover	${HPSS_PATH_BIN}/hpss_mvr
HPSS_EXEC_MVR_TCP	executable name for Mover TCP	${HPSS_PATH_BIN}/hpss_mvr_tcp
HPSS_EXEC_NDCG	executable name for Non-DCE Gateway	${HPSS_PATH_BIN}/hpss_ndcg
HPSS_EXEC_NFSD	executable name for NFS Daemon	${HPSS_PATH_BIN}/hpss_nfs2
HPSS_EXEC_NS	executable name for Name Server	${HPSS_PATH_BIN}/hpss_cns
HPSS_EXEC_PFSD	executable name for PFS Daemon	${HPSS_PATH_BIN}/hpss_piofsie
HPSS_EXEC_PVL	executable name for PVL	${HPSS_PATH_BIN}/hpss_pvl
HPSS_EXEC_PVR_AMPEX	executable name for PVR Ampex	${HPSS_PATH_BIN}/hpss_pvr_ampex
HPSS_EXEC_PVR_OPER	executable name for PVR Operator	${HPSS_PATH_BIN}/hpss_operator
HPSS_EXEC_PVR_STK	executable name for PVR STK	${HPSS_PATH_BIN}/hpss_stk
HPSS_EXEC_PVR_3494	executable name for PVR 3494	${HPSS_PATH_BIN}/hpss_3494
HPSS_EXEC_PVR_3495	executable name for PVR 3495	${HPSS_PATH_BIN}/hpss_3495
HPSS_EXEC_PVR_AML	executable name for PVR AML	${HPSS_PATH_BIN}/hpss_pvr_aml
HPSS_EXEC_SSDISK	executable name for Storage Server - Disk	${HPSS_PATH_BIN}/hpss_disk
HPSS_EXEC_SSTAPE	executable name for Storage Server - Tape	${HPSS_PATH_BIN}/hpss_tape

Table 4-1 HPSS Environment Variables (Continued)

Variable	Description	Default Value
HPSS_EXEC_SSMDS	executable name for SSM Data Server	${HPSS_PATH_BIN}/hpss_ssmds
HPSS_EXEC_SSMSM	executable name for SSM Storage Manager	${HPSS_PATH_BIN}/hpss_ssm
HPSS_EXEC_ACL_EDIT	Pathname for the acl_edit utility	${HPSS_PATH_SLASH_BIN}/acl_edit
HPSS_EXEC_CDSCP	Pathname for the cdscp utility	${HPSS_PATH_SLASH_BIN}/cdscp
HPSS_EXEC_DELOG	Pathname for the delog utility	${HPSS_PATH_BIN}/hpss_delog
HPSS_EXEC_RECLAIM	Pathname for the for the reclaim utility	${HPSS_PATH_BIN}/reclaim.ksh
HPSS_EXEC_REPACK	Pathname for the repack utility	${HPSS_PATH_BIN}/repack
HPSS_PATH_LOG	unix path name for logging files	${HPSS_PATH_VAR}/log
HPSS_UNIX_LOCAL_LOG	local log file	${HPSS_PATH_LOG}/local.log
HPSS_PATH_ACCT	unix path name for accounting files	${HPSS_PATH_VAR}/acct
HPSS_UNIX_ACCT_CHECKPOINT	accounting checkpoint file	${HPSS_PATH_ACCT}/acct_checkpoint
HPSS_UNIX_ACCT_REPORT	accounting report file	${HPSS_PATH_ACCT}/acct_report
HPSS_UNIX_ACCT_COMMENTARY	accounting commentary file	${HPSS_PATH_ACCT}/acct_commentary
HPSS_PATH_NFS	unix path name for NFS files	${HPSS_PATH_VAR}/nfs
HPSS_UNIX_NFS_EXPORTS	NFS exports file	${HPSS_PATH_NFS}/exports
HPSS_UNIX_NFS_CREDMAP	NFS credmap file	${HPSS_PATH_NFS}/credmap.nfs2
HPSS_UNIX_NFS_CACHEFILE	NFS cache file	${HPSS_PATH_NFS}/cachefile.nfs2
HPSS_UNIX_NFS_CHECKPOINT	NFS checkpoint	${HPSS_PATH_NFS}/checkpoint.nfs2
HPSS_PATH_MPS	unix path name for MPS files	${HPSS_PATH_VAR}/mps
HPSS_UNIX_MPS_REPORT	MPS report file	“”
HPSS_SFS	Encina SFS server name without CDS prefix	hpss
HPSS_SFS_SERVER	Encina SFS server name with CDS prefix	/.:/encina/server/${HPSS_SFS}

Table 4-1 HPSS Environment Variables (Continued)

Variable	Description	Default Value
HPSS_SFS_SUFFIX	Suffix to add to end of SFS file names	""
HPSS_CONFIG_ACCOUNTING	Common SFS Files - Accounting Policy	${HPSS_SFS_SERVER}/accounting${HPSS_SFS_SUFFIX}
HPSS_CONFIG_ACCT	Common SFS Files - Old name for Accounting Policy	${HPSS_CONFIG_ACCOUNTING}
HPSS_CONFIG_ACCTSUM	Common SFS Files - Accounting summary	${HPSS_SFS_SERVER}/acctsum${HPSS_SFS_SUFFIX}
HPSS_CONFIG_COS	Common SFS Files - Class of service	${HPSS_SFS_SERVER}/cos${HPSS_SFS_SUFFIX}
HPSS_CONFIG_FILE_FAMILY	Common SFS Files - File Family	${HPSS_SFS_SERVER}/filefamily${HPSS_SFS_SUFFIX}
HPSS_CONFIG_SCLASS	Common SFS Files - Old name for Storage class	${HPSS_SFS_SERVER}/storageclass${HPSS_SFS_SUFFIX}
HPSS_CONFIG_STORAGECLASS	Common SFS Files - Storage class	${HPSS_CONFIG_SCLASS}
HPSS_CONFIG_HIERARCHY	Common SFS Files - Hierarchy	${HPSS_SFS_SERVER}/hierarchy${HPSS_SFS_SUFFIX}
HPSS_CONFIG_SERVER	Common SFS Files - Generic server configurations	${HPSS_SFS_SERVER}/serverconfig${HPSS_SFS_SUFFIX}
HPSS_CONFIG_ACCTSNAP	Accounting SFS File - snap shot	${HPSS_SFS_SERVER}/acctsnap${HPSS_SFS_SUFFIX}
HPSS_CONFIG_ACCTLOG	BFS SFS File - Accounting log	${HPSS_SFS_SERVER}/acctlog${HPSS_SFS_SUFFIX}
HPSS_CONFIG_BFCOSCHANGE	BFS SFS File - COS changes	${HPSS_SFS_SERVER}/bfcoschange${HPSS_SFS_SUFFIX}
HPSS_CONFIG_BFDISKALLOCREC	BFS SFS File - disk allocation records	${HPSS_SFS_SERVER}/bfdiskallocrec${HPSS_SFS_SUFFIX}
HPSS_CONFIG_BFDISKSEGMENT	BFS SFS File - bitfile segments - disk	${HPSS_SFS_SERVER}/bfdisksegment${HPSS_SFS_SUFFIX}
HPSS_CONFIG_BFMIGRREC	BFS SFS File - migration records	${HPSS_SFS_SERVER}/bfmigrrec${HPSS_SFS_SUFFIX}
HPSS_CONFIG_BFPURGEREC	BFS SFS File - purge records	${HPSS_SFS_SERVER}/bfpurgerec${HPSS_SFS_SUFFIX}

Table 4-1 HPSS Environment Variables (Continued)

Variable	Description	Default Value
HPSS_CONFIG_BFS	BFS SFS File -type specific	${HPSS_SFS_SERVER}/ bfs${HPSS_SFS_SUFFIX}
HPSS_CONFIG_BFSSSEGCHKPT	BFS SFS File - segment checkpoints	${HPSS_SFS_SERVER}/ bfsssegchkpt${HPSS_SFS_SUFFIX}
HPSS_CONFIG_BFSSUNLINK	BFS SFS File - unlink records	${HPSS_SFS_SERVER}/ bfssunlink${HPSS_SFS_SUFFIX}
HPSS_CONFIG_BFTAPESEGMENT	BFS SFS File - bitfile segments - tape	${HPSS_SFS_SERVER}/ bftapesegment${HPSS_SFS_SUFFIX}
HPSS_CONFIG_BITFILE	BFS SFS File - bitfile descriptors	${HPSS_SFS_SERVER}/ bitfile${HPSS_SFS_SUFFIX}
HPSS_CONFIG_NS	NS SFS File - type specific	${HPSS_SFS_SERVER}/ nsconfig${HPSS_SFS_SUFFIX}
HPSS_CONFIG_NSACLS	NS SFS File - ACLs	${HPSS_SFS_SERVER}/ nsacls${HPSS_SFS_SUFFIX}
HPSS_CONFIG_NSFILESETATTRS	NS SFS File - Fileset attributes	${HPSS_SFS_SERVER}/ nsfilesetattrs${HPSS_SFS_SUFFIX}
HPSS_CONFIG_NSGLOBALFILESETS	NS SFS File - Global Filesets	${HPSS_SFS_SERVER}/ nsglobalfilesets${HPSS_SFS_SUFFIX}
HPSS_CONFIG_NSOBJECTS	NS SFS File - objects	${HPSS_SFS_SERVER}/ nsobjects${HPSS_SFS_SUFFIX}
HPSS_CONFIG_NSTEXT	NS SFS File - text	${HPSS_SFS_SERVER}/ nstext${HPSS_SFS_SUFFIX}
HPSS_CONFIG_DMG	DMG SFS File - type specific	${HPSS_SFS_SERVER}/ dmg${HPSS_SFS_SUFFIX}
HPSS_CONFIG_DMGFILESET	DMG SFS File -filesets	${HPSS_SFS_SERVER}/ dmgfileset${HPSS_SFS_SUFFIX}
HPSS_CONFIG_LOGC	LOG SFS File -Log client type specific	${HPSS_SFS_SERVER}/ logclient${HPSS_SFS_SUFFIX}
HPSS_CONFIG_LOGD	LOG SFS File -Log daemon type specific	${HPSS_SFS_SERVER}/ logdaemon${HPSS_SFS_SUFFIX}
HPSS_CONFIG_LOGP	LOG SFS File -Old name for Log policy	${HPSS_SFS_SERVER}/ logpolicy${HPSS_SFS_SUFFIX}
HPSS_CONFIG_LOGPOLICY	LOG SFS File -Log policy	${HPSS_CONFIG_LOGP}

Table 4-1 HPSS Environment Variables (Continued)

Variable	Description	Default Value
HPSS_CONFIG_LS	LS SFS File - Old name for Location Server policy	${HPSS_SFS_SERVER}/ lspolicy${HPSS_SFS_SUFFIX}
HPSS_CONFIG_LSPOLICY	LS SFS File - Location Server policy	${HPSS_CONFIG_LS}
HPSS_CONFIG_SITE	LS SFS File - site file	${HPSS_SFS_SERVER}/ site${HPSS_SFS_SUFFIX}
HPSS_CONFIG_MM	MM SFS File - type specific	${HPSS_SFS_SERVER}/ mmonitor${HPSS_SFS_SUFFIX }
HPSS_CONFIG_MOUNTD	MOUNTD SFS File - Mount Daemon type specific	${HPSS_SFS_SERVER}/ mountd${HPSS_SFS_SUFFIX}
HPSS_CONFIG_MIGRP	MPS SFS File - Old name for MPS migration policy	${HPSS_SFS_SERVER}/ migpolicy${HPSS_SFS_SUFFIX }
HPSS_CONFIG_MIGPOLICY	MPS SFS File - migration policy	${HPSS_CONFIG_MIGRP}
HPSS_CONFIG_MPCHKPT	MPS SFS File - checkpoint	${HPSS_SFS_SERVER}/ mpchkpt${HPSS_SFS_SUFFIX}
HPSS_CONFIG_MPS	MPS SFS File - type specific	${HPSS_SFS_SERVER}/ mps${HPSS_SFS_SUFFIX}
HPSS_CONFIG_PURGP	MPS SFS File - Old name for MPS purge policy	${HPSS_SFS_SERVER}/ purgepolicy${HPSS_SFS_SUFFIX}
HPSS_CONFIG_PURGEPOLICY	MPS SFS File - purge policy	${HPSS_CONFIG_PURGP}
HPSS_CONFIG_MOVERDEVICE	MVR SFS File - device	${HPSS_SFS_SERVER}/ moverdevice${HPSS_SFS_SUFFIX}
HPSS_CONFIG_MVR	MVR SFS File - type specific	${HPSS_SFS_SERVER}/ mover${HPSS_SFS_SUFFIX}
HPSS_CONFIG_NDCG	Gateway type specific	${HPSS_SFS_SERVER}/ ndcg${HPSS_SFS_SUFFIX}
HPSS_CONFIG_NFSD	NFSD SFS File - type specific	${HPSS_SFS_SERVER}/ pvl${HPSS_SFS_SUFFIX}
HPSS_CONFIG_PVL	PVL SFS File - type specific	${HPSS_SFS_SERVER}/ pvlactivity${HPSS_SFS_SUFFIX
HPSS_CONFIG_PVLACTIVITY	PVL SFS File - activity	${HPSS_SFS_SERVER}/ pvlactivity${HPSS_SFS_SUFFIX}
HPSS_CONFIG_PVLDRIVE	PVL SFS File - drive	${HPSS_SFS_SERVER}/ pvldrive${HPSS_SFS_SUFFIX}
HPSS_CONFIG_PVLJOB	PVL SFS File - job	${HPSS_SFS_SERVER}/ pvljob${HPSS_SFS_SUFFIX}

Table 4-1 HPSS Environment Variables (Continued)

Variable	Description	Default Value
HPSS_CONFIG_PVLPV	PVL SFS File - physical volume	${HPSS_SFS_SERVER}/pvlpv${HPSS_SFS_SUFFIX}
HPSS_CONFIG_PVR	PVR SFS File - type specific	${HPSS_SFS_SERVER}/pvr${HPSS_SFS_SUFFIX}
HPSS_CONFIG_CART_AMPEX	PVR ampex cartridges	${HPSS_SFS_SERVER}/cartridge_ampex${HPSS_SFS_SUFFIX}
HPSS_CONFIG_CART_OPER	PVR SFS File - operator cartridges	${HPSS_SFS_SERVER}/cartridge_operator${HPSS_SFS_SUFFIX}
HPSS_CONFIG_CART_STK	PVR SFS File - STK cartridges	${HPSS_SFS_SERVER}/cartridge_stk${HPSS_SFS_SUFFIX}
HPSS_CONFIG_CART_3494	PVR SFS File - 3494 cartridges	${HPSS_SFS_SERVER}/cartridge_3494${HPSS_SFS_SUFFIX}
HPSS_CONFIG_CART_3495	PVR SFS File - 3495 cartridges	${HPSS_SFS_SERVER}/cartridge_3495${HPSS_SFS_SUFFIX}
HPSS_CONFIG_CART_AML	PVR AML cartridges	${HPSS_SFS_SERVER}/cartridge_aml${HPSS_SFS_SUFFIX}
HPSS_CONFIG_SS	SS SFS File - type specific	${HPSS_SFS_SERVER}/ss${HPSS_SFS_SUFFIX}
HPSS_CONFIG_SSPVDISK	SS SFS File - physical volume - disk	${HPSS_SFS_SERVER}/sspvdisk${HPSS_SFS_SUFFIX}
HPSS_CONFIG_SSPVTAPE	SS SFS File - physical volume - tape	${HPSS_SFS_SERVER}/sspvtape${HPSS_SFS_SUFFIX}
HPSS_CONFIG_STORAGEMAPDISK	SS SFS File - storage map - disk	${HPSS_SFS_SERVER}/storagemapdisk${HPSS_SFS_SUFFIX}
HPSS_CONFIG_STORAGEMAPTAPE	SS SFS File - storage map - tape	${HPSS_SFS_SERVER}/storagemaptape${HPSS_SFS_SUFFIX}
HPSS_CONFIG_STORAGESEGDISK	SS SFS File - storage segment - disk	${HPSS_SFS_SERVER}/storagesegdisk${HPSS_SFS_SUFFIX}
HPSS_CONFIG_STORAGESEGTAPE	SS SFS File - storage segment - tape	${HPSS_SFS_SERVER}/storagesegtape${HPSS_SFS_SUFFIX}
HPSS_CONFIG_VVDISK	SS SFS File - virtual volume - disk	${HPSS_SFS_SERVER}/vvdisk${HPSS_SFS_SUFFIX}

Table 4-1 HPSS Environment Variables (Continued)

Variable	Description	Default Value
HPSS_CONFIG_VVTAPE	SS SFS File - virtual volume - tape	${HPSS_SFS_SERVER}/vvtape${HPSS_SFS_SUFFIX}
HPSS_CDS_PREFIX	CDS prefix for HPSS servers	/.:/hpss
HPSS_CDS_BFS	CDS name - Bitfile Server	${HPSS_CDS_PREFIX}/bfs
HPSS_CDS_DMG	CDS name - DMAP Gateway	${HPSS_CDS_PREFIX}/dmg
HPSS_CDS_HPSSD	CDS name - Startup Daemon	${HPSS_CDS_PREFIX}/hpssd
HPSS_CDS_LOGC	CDS name - Log Client	${HPSS_CDS_PREFIX}/logc
HPSS_CDS_LOGD	CDS name - Log Daemon	${HPSS_CDS_PREFIX}/logc
HPSS_CDS_LS	CDS name - Location Server	${HPSS_CDS_PREFIX}/ls
HPSS_CDS_MM	CDS name - Metadata Monitor	${HPSS_CDS_PREFIX}/monitor
HPSS_CDS_MOUNTD	CDS name - Mount Daemon	${HPSS_CDS_PREFIX}/mountd
HPSS_CDS_MPS	CDS name - Migration/Purge Server	${HPSS_CDS_PREFIX}/mps
HPSS_CDS_MVR	CDS name - Mover	${HPSS_CDS_PREFIX}/mvr
HPSS_CDS_NFSD	CDS name - NFS Daemon	${HPSS_CDS_PREFIX}/nfs2
HPSS_CDS_NS	CDS name - Name Server	${HPSS_CDS_PREFIX}/cns
HPSS_CDS_PFSD	CDS name - PFSD	${HPSS_CDS_PREFIX}/piofsie
HPSS_CDS_PVL	CDS name - PVL	${HPSS_CDS_PREFIX}/pvl
HPSS_CDS_PVR_AMPEX	CDS name - PVR - Ampex	${HPSS_CDS_PREFIX}/pvr_ampex
HPSS_CDS_PVR_OPER	CDS name - PVR - Operator	${HPSS_CDS_PREFIX}/pvr_operator
HPSS_CDS_PVR_STK	CDS name - PVR - STK	${HPSS_CDS_PREFIX}/pvr_stk
HPSS_CDS_PVR_3494	CDS name - PVR - 3494	${HPSS_CDS_PREFIX}/pvr_3494
HPSS_CDS_PVR_3495	CDS name - PVR - 3495	${HPSS_CDS_PREFIX}/pvr_3495
HPSS_CDS_PVR_AML	CDS name - PVR - AML	${HPSS_CDS_PREFIX}/pvr_aml
HPSS_CDS_SSDISK	CDS name - Storage Server - Disk	${HPSS_CDS_PREFIX}/ssdisk
HPSS_CDS_SSTAPE	CDS name - Storage Server - Tape	${HPSS_CDS_PREFIX}/sstape
HPSS_CDS_SSMDS	CDS name - SSM - Data Server	${HPSS_CDS_PREFIX}/ssmds

Table 4-1 HPSS Environment Variables (Continued)

Variable	Description	Default Value
HPSS_CDS_SSMSM	CDS name - SSM - System Manager	${HPSS_CDS_PREFIX}/ssmsm
HPSS_DESC_BFS	Descriptive name - Bitfile Server	“BFS”
HPSS_DESC_DMG	Descriptive name - DMAP Gateway	“DMAP Gateway”
HPSS_DESC_FTPD	Descriptive name - FTP Daemon	“FTP Daemon”
HPSS_DESC_HPSSD	Descriptive name - Startup Daemon	“Startup Daemon (${HPSS_HOST})”
HPSS_DESC_LOGC	Descriptive name - Log Client	“Log Client (${HPSS_HOST})”
HPSS_DESC_LOGD	Descriptive name - Log Daemon	“Log Daemon”
HPSS_DESC_LS	Descriptive name - Location Server	“Location Server”
HPSS_DESC_MM	Descriptive name - Metadata Monitor	“Metadata Monitor”
HPSS_DESC_MOUNTD	Descriptive name - Mount Daemon	“Mount Daemon”
HPSS_DESC_MPS	Descriptive name - MPS	“Migration/Purge Server”
HPSS_DESC_NDCG	Descriptive name - Non-DCE Gateway	"Non-DCE Gateway"
HPSS_DESC_MVR	Descriptive name - Mover	“Mover (${HPSS_HOST})”
HPSS_DESC_NFSD	Descriptive name - NFS Daemon	“NFS V2 Daemon”
HPSS_DESC_NS	Descriptive name - Name Server	“Name Server”
HPSS_DESC_PFSD	Descriptive name - PFSD	“PIOFS Import/Export”
HPSS_DESC_PVL	Descriptive name - PVL	“PVL”
HPSS_DESC_PVR_AMPEX	Descriptive name - PVR - Ampex	"Ampex PVR"
HPSS_DESC_PVR_OPER	Descriptive name - PVR - Operator	“Operator PVR”
HPSS_DESC_PVR_STK	Descriptive name - PVR - STK	“STK PVR”
HPSS_DESC_PVR_3494	Descriptive name - PVR - 3494	“3494 PVR”
HPSS_DESC_PVR_3495	Descriptive name - PVR - 3495	“3495 PVR”
HPSS_DESC_PVR_AML	Descriptive name - PVR - AML	"AML PVR"
HPSS_DESC_SSDISK	Descriptive name - SS - Disk	“Disk Storage Server”
HPSS_DESC_SSTAPE	Descriptive name - SS - Tape	“Tape Storage Server”
HPSS_DESC_SSMSM	Descriptive name - SSM System Manager	“SSM System Manager”

Table 4-1 HPSS Environment Variables (Continued)

Variable	Description	Default Value
HPSS_SM_SRV_CONNECT_FAIL_COUNT	Number of connection failures to a server before the HPSS_SM_SRV_CONNECT_INTERVAL_MAX interval takes affect	3
HPSS_SM_SRV_CONNECT_INTERVAL_MIN	Interval between attempting server connections when HPSS_SM_SERVER_CONNECT_FAIL_COUNT has not yet been reached (seconds)	20
HPSS_SM_SRV_CONNECT_INTERVAL_MAX	Interval between server connections when HPSS_SM_SERVER_CONNECT_FAIL_COUNT has been reached without a successful connection (seconds)	60
HPSS_SM_SRV_LIST_UPDATE_INTERVAL	Frequency at which the SM sends updated server lists to the Data Server so that we don't flood the Data Server with updates (seconds)	5
HPSS_SM_SRV_MONITOR_THREADS	Number of threads created to monitor server connections	10
ENCINA_TPOOL_SIZE	Thread Pool Size used by the System Manager	100
HPSS_NOTIFY_Q_DATA_THREADS	Number of threads to create per client to process the queue of notifications of managed object attribute changes	1
HPSS_NOTIFY_Q_LIST_THREADS	Number of threads to create per client to process the queue of notifications of changes to the server list, drive list, class of service list, storage class list, hierarchy list, migration policy list, or purge policy list	1
HPSS_NOTIFY_Q_LOG_THREADS	Number of threads to create per client to process the queue of notifications of alarms, events, and status messages	1
HPSS_NOTIFY_Q_TAPE_THREADS	Number of threads to create per client to process the queue of notifications of tape mounts and unmounts	1
HPSS_NOTIFY_Q_TAPE_CHECKIN_THREADS	Number of threads to create per client to process the queue of tape check-in requests	1

Table 4-1 HPSS Environment Variables (Continued)

Variable	Description	Default Value
HPSS_NOTIFY_Q_DATA_SIZE _MAX	Maximum queue size for notifications of changes in managed object attributes	500
HPSS_NOTIFY_Q_LIST_SIZE _MAX	Maximum queue size for notifications of changes to the server list, drive list, class of service list, storage class list, hierarchy list, migration policy list, or purge policy list	500
HPSS_NOTIFY_Q_LOG_SIZE _MAX	Maximum queue size for notifications of alarms, events, or status messages	500
HPSS_NOTIFY_Q_TAPE_SIZE _MAX	Maximum queue size for notifications of tape mounts and unmounts	500
HPSS_NOTIFY_Q_TAPE_CHECKIN_SIZ E_MAX	Maximum queue size for tape check-in requests	500
HPSS_ALARMS_SSMDS	File to store Sammi Alarms/Events NULL -> DS will store internally	NULL
HPSS_PORT_SSMDS	Sun RPC address for the Data Server	0x20000069
HPSS_HDM_SERVER_ID	The HDM's Server ID (default 1)	1
HPSS_HDM_SHMEM_KEY	The HDM's shared memory key (default 3789)	3789
HPSS_PATH_HDM	Pathname where HDM configuration files are placed	${HPSS_PATH_VAR}/hdm
HPSSLOG_SHMKEY	The HPSS logging shared memory key (default 2801)	2801
HPSSLOGGER	HPSS logger output destination [stdout | syslog]	""
HPSS_PATH_FTP	Pathname where FTP Daemon control files are placed	${HPSS_PATH_VAR}/ftp
HPSS_FTPD_CONTROL _PORT	FTP Daemon control default port ID	4021
HPSS_FTP_RESERVED	FTP reserved port IDs	1025
HPSS_FTP_BLOCK_SIZE	FTP block size	4
HPSS_LS_NAME	Location Server rpc group	${HPSS_CDS_LS}/group
RPC_SUPPORTED_PROTSEQS	The RPC transport protocol used	ncadg_ip_udp

Table 4-1 HPSS Environment Variables (Continued)

Variable	Description	Default Value
HPSS_PATH_RHOST_TMP	HPSS temporary pathname on the remote host	/usr/lpp/hpss_tar
HPSS_PATH_TAR_FILE	Pathname where HPSS remote install tar files are placed	${HPSS_PATH}/tarfile
HPSS_PATH_TMP	Pathname where HPSS temporary files are placed	${HPSS_PATH_VAR}/tmp
HPSS_PATH_ETC	Pathname where HPSS runtime config files are placed	${HPSS_PATH_VAR}/etc
HPSS_PATH_ADM	Pathname where HPSS administrative files are placed. Currently this directory will include a directory to include core files and a failed server log.	${HPSS_PATH_VAR)/adm
HPSS_PATH_CORE	Pathname where directories created to contain core files generated by HPSS servers are placed.	${HPSS_PATH_ADM}/core
HPSS_AUTH_LEVEL	The security authorization level	2

4.4 Configure the HPSS Infrastructure on the Installation Node

The HPSS Infrastructure Configuration must be performed on the installation node after the HPSS system is successfully installed. The following steps should be performed for the installation node:

1. Configure HPSS with DCE.
2. Configure Encina SFS (only if Encina SFS will run on this node).
3. Set up FTP Daemon (only if FTP Daemon will run on this node).
4. Set up Startup Daemon.
5. Install HPSS subsystem on remote node (if needed).
6. Add SSM User (only if SSM will run on this node).
7. Start SSM Servers / Session (only if SSM will run on this node).

The HPSS Infrastructure Configuration utility, **mkhpss**, is used to configure the above steps, except for the Encina SFS configuration. Refer to Section 4.6 for more information on the options supported by the utility. Refer to Section 4.7 for more information on how to configure Encina SFS.

To start the HPSS infrastructure configuration on the installation node:

1. Log in as **root** user to the node.
2. Change directory to **/usr/lpp/hpss/config**:

   % cd /usr/lpp/hpss/config

3. Execute the shell script **mkhpss**:

   % ./mkhpss

If the HPSS infrastructure is already configured on the installation node, the mkhpss script will issue a warning message. In this case, mkhpss will only allow the administrator to perform "Add-on" (add additional configuration), "Re-do" (delete existing configuration and reconfigure), "Un-do" (delete the existing configuration) or "Exit" (terminate mkhpss's execution). Refer to Section 4.6 for more information on these options.

4. Select the proper menu option for the desired procedure and, where appropriate, enter the planning information recorded in the HPSS Infrastructure Configuration Worksheets (Section C.3). The worksheets in Appendix C should have been used during the configuration planning phase described in Section 2.1.2.

4.5 Configure the HPSS Infrastructure on Each Remote Node

The HPSS infrastructure configuration must be performed on each remote node after the HPSS subsystem (except **hdm**, **ndc_aix**, or **ndc_sgi**) has been installed from the installation node. The following steps should be performed for each remote node:

1. Set up FTP Daemon (only if FTP Daemon is installed and will run on this node).
2. Configure Encina SFS (only if Encina SFS will run on this node).
3. Set up Startup Daemon.
4. Add SSM User (only if SSM is installed and will run on this node).
5. Start SSM Servers / Session (only if SSM is installed and will run on this node).

The HPSS Infrastructure Configuration utility, **mkhpss**, is used to configure the above steps, except for the Encina SFS configuration. Refer to Section 4.6 for more information on the options supported by the utility. Refer to Section 4.7 for more information on how to configure Encina SFS.

To start the HPSS infrastructure configuration on the remote node:

1. Log in as **root** user to the node.
2. Change directory to **/usr/lpp/hpss/config**:

   % cd /usr/lpp/hpss/config

3. Execute the shell script **mkhpss**:

   % ./mkhpss

If the HPSS infrastructure is already configured on the remote node, the mkhpss script will issue a warning message. In this case, mkhpss will only allow the administrator to perform "Add-on" (add additional configuration), "Re-do" (delete existing configuration and reconfigure), "Un-do" (delete the existing configuration) or "Exit" (terminate mkhpss's execution). Refer to Section 4.6 for more information on these options.

4. Select the proper menu option for the desired procedure and, where appropriate, enter the planning information recorded in the HPSS Infrastructure Configuration Worksheets (Section C.3). These worksheets in Appendix C should have been used during the configuration planning phase described in Section 2.1.2.

4.6 *Using the* mkhpss *Utility*

The **mkhpss** utility is an interactive, menu-driven HPSS process. It provides menu options that allow the user to perform the needed infrastructure configurations (as described in Section 4.6.1 through 4.6.9) as well as an Exit option to quit the utility. Section 4.6.10 describes additional options not on the Menu.

[1] Configure HPSS with DCE

[2] Manage SFS Files

[3] Setup FTP Daemon

[4] Setup Startup Daemon

[5] Install HPSS Subsystem on Remote Node

[6] Add SSM Administrative User

[7] Start SSM Servers / User Session

[E] Re-run hpss_env()

[R] Re-configure HPSS

[X] Exit

Messages will be provided to indicate the status of the HPSS infrastructure configuration process at each stage. The HPSS infrastructure configuration results can be obtained from the messages displayed during the HPSS infrastructure configuration process or can be reviewed by reading the **/usr/lpp/hpss/config/mkhpss.data** file.

Although it is recommended that the mkhpss script be used only once to configure the HPSS infrastructure, it can be run again to correct or modify the infrastructure configuration as needed. However, after the HPSS servers are configured, mkhpss should not be used to modify DCE, Encina, or any other data that may affect the configuration of the HPSS system.

The following steps are automatically performed at the beginning of each **mkhpss** execution:

- Load in the environment variables defined in the **hpss_env** file and **hpss_env_defs.h** file.
- Verify that the user has root authority.
- If the HPSS infrastructure is already configured, prompt the user if additional configuration, unconfiguration, or reconfiguration is desired.
- Edit the **mkhpss.data** file to record status messages and processed states.

4.6.1 Configure HPSS with DCE

This option can only be performed from the installation node. The following steps are automatically performed:

1. Remove the HPSS keytab files, if any exist.
2. Create DCE groups for the HPSS servers and clients.
3. Create DCE principals and accounts for HPSS servers.
4. Create keytab files for HPSS servers and clients.
5. Randomize the keys in the created server and client keytabs.
6. Set up the DCE CDS directory for HPSS servers.

4.6.2 Manage SFS Files

The **mkhpss** script invokes the **managesfs** utility, a front-end to the **sfsadmin** commands provided by Encina, to allow the user to manage the SFS files created for HPSS. The Manage SFS Files option should not be selected until the SFS has been configured as discussed in Section 4.7.

The script will present a menu to allow the user to create, delete, empty, query, and list SFS files. Refer to Appendix I for more information on the **managesfs** utility.

4.6.3 Setup FTP Daemon

The following steps are performed when the Setup FTP Daemon option is selected:

1. Prompt the user for the port number to be used by the HPSS FTP.
2. Add hpssftp and hpssftp-data entries to the **/etc/services** file.
3. Add the hpssftp entry to the **/etc/inetd.conf** file.
4. Set up HPSS FTP Daemon directories and template files.

4.6.4 Setup Startup Daemon

The following steps are performed if the Setup Startup Daemon option is selected:

1. Add the hpssd entry to the **/etc/inittab** file.

2. Copy the rc.hpss file to the **/etc** directory.

4.6.5 Install HPSS Subsystem on Remote Nodes

When one or more HPSS subsystems are to be executed from a node other than the installation node, this procedure will install the specified subsystems and the supporting infrastructure components to the remote node. This procedure can only be performed from the installation node.

A file, **rmkhpss.data**, is created to record the node names that are requested for remote installation along with the installed HPSS components associated with each of these nodes.

Tar files containing HPSS components are used to install the selected HPSS subsystems to the remote nodes. If the tar files exist, the user will be prompted to determine whether the existing tar files are to be used for remote installation. If the tar files do not exist or the user selects to replace the existing tar files, the tar files will be generated from the **/usr/lpp/hpss** directory.

The following steps are automatically performed to remotely install the specified HPSS subsystems to a remote node:

1. Prompt the user for the target node name.

2. Prompt the user for the root password for the target node.

3. Prompt the user for the HPSS components to be installed.

4. Copy selected HPSS component tar files to the target node.

5. Extract HPSS components from the tar files.

6. Copy the HPSS servers and clients keytab files generated at the installation node to the target node.

The Infrastructure component will be automatically installed on the remote node if any HPSS component is selected in Step 3. The Infrastructure component includes the Accounting server, Log Client, Metadata Monitor, Startup Daemon, HPSS Utilities, and HPSS Message Catalog.

4.6.6 Add SSM Administrative User

The HPSS system is ready to be configured using SSM once the HPSS software is installed on the installation node, the appropriate HPSS subsystems are remotely installed to other nodes as needed, and the HPSS infrastructure components are configured. The **mkhpss** script will optionally allow the user to configure a SSM user with authority level of “administrator” to use SSM to configure the HPSS system.

The **mkhpss** script will invoke a subset of the HPSS User Management Utility (**hpssuser**) to create the SSM User ID and password, the required Sammi configuration data, and the required AIX and DCE accounts (if they do not already exist) for the SSM user. Due to the AIX limit, the length of the

user ID must not exceed eight characters. If this step is not desired at this point, refer to Section 6.12.1 for instructions to create an SSM user session at a later time.

Even though all SSM User IDs and their prerequisite accounts can now be created, we recommend that only an administrative SSM ID be created at this time. The hpssuser utility should be invoked to create a new HPSS account after the HPSS system is operational so that all desired accounts (AIX, DCE, SSM, or FTP) for a new HPSS user can be created at the same time.

4.6.7 Start Up SSM Servers/User Session

If requested, the **mkhpss** script will start up the SSM servers and the SSM administration session so that the user can configure the HPSS system. If this step is not desired at this point, refer to Section 5.2.1 for instructions to start up the SSM servers and the SSM user sessions at a later time.

4.6.8 Re-run hpss_env()

If requested, the **mkhpss** script will re-read the environment variable values defined in the **hpss_env** file and the **hpss_env_defs.h** file.

4.6.9 Re-configure HPSS

If requested, the **mkhpss** script will delete the existing configuration and release any resources allocated by the configuration. After the existing configuration is deleted successfully, the script allows the administrator to configure a new configuration.

4.6.10 Additional Options

The **Add-on** and **Un-do** options are not available from the menu. They are prompted to the user when **mkhpss** detects that the HPSS infrastructure is already configured.

Add-on

This option allows the administrator to add an additional configuration to the existing infrastructure. The user is allowed to select any configuration options from the menu. This option should be used with care since an attempt to reconfigure an existing configuration may corrupt or duplicate the existing configuration.

Un-do

This option allows the administrator to delete the existing configuration and release any resources allocated by the configuration.

4.7 Configure Encina SFS Server

This section describes the guidelines and procedures to manually configure, start and stop the TXSeries 4.2 Encina SFS server. While the Encina enconsole interface can be used to configure the SFS server, this creates Encina components not needed by HPSS, which creates unnecessary overhead and adds additional complexity to the Encina product. We recommended that the following procedures be used to configure the SFS servers for HPSS.

4.7.1 Planning for the Encina SFS Server Configuration

All examples below use the machine name "datastore", cell name "dce.other.org", and refer to volume groups "optvg" & "sfsvg" that are RAID protected. If a mirrored configuration for the SFS server is desired, additional file system and logical volume create parameters must be specified. Also, the discussion below describes the overall structure to build the SFS environment. It does not try to optimize performance from the adapter/disk resources available. Each site should analyze their configuration and consult with Transarc and HPSS support representatives to customize a solution that best utilizes their system resources.

The SFS configuration requires several file systems to be created. The server also requires available space on a volume group to create the logical volumes for the SFS log volume and SFS data volumes. Assuming that two RAID volume groups, **optvg** and **sfsvg**, are to be used to configure the SFS server **/.:/encina/sfs/hpss**, allocate the file systems and the required logical volumes as follows:

1. Create the following file systems on the "**optvg**" volume group:

   /opt/encinalocal

   /opt/encinamirror

   /opt/sfs_backup_area

The "**encinamirror**" file system can be very small since its size is never expected to grow. If possible, the **encinamirror** and **encinalocal** file systems should be created on separate volume groups.

The "**/opt/sfs_backup_area**" will contain the backups of the data volumes (TRB files generated from the "**tkadmin backup lvol**" command). This area must be large enough to accommodate the TRB files until they are safely copied to other media.

The "**encinalocal**" file system can become quite large depending upon the number of transactions the site may generate and upon whether the archives subdirectory is a separate file system. By default, Encina places log archive backup files (MRAs) in **/opt/encinalocal/encina/sfs/hpss/archives** (for SFS server **/.:/encina/sfs/hpss**). On very active systems, a great number of MRA files will be generated, consuming a great deal of file system space. These MRA files must not be removed until they have been backed up to other media. They can be compressed to save space, but caution must be used to make sure the current active MRA file is never compressed or altered until it is completely written and the next MRA file is started. During initial Encina configuration, it is best to configure a separate file system (e.g., **/archive**) for the MRA files and to create **/opt/encinalocal/encina/sfs/hpss/archives** as a symbolic link to **/archives**. In this case, the "**encinalocal**" file system itself will not grow very much.

2. On the "**sfsvg**" volume group, the logical volumes associated with the SFS log and data volumes are to be created. (See note in foreword about performance. Also, SFS log and data volumes should not share physical disks in a mirrored configuration.) For ease of administration, the HPSS SFS files should be spread across 3-5 (or more) SFS data volumes. This will help make backup files smaller and easier to manage. It will help later, when additional space is needed to expand the system. However, since SFS only tracks up to a maximum of 512 TRB files, care should be taken to balance the number of data volumes, size of backup files and number of TRB files required for backing up SFS.

   Refer to Section 2.9.2 for more information on the sizing of the HPSS metadata files and how to use the Metadata Sizing Worksheet to determine the total disk space requirement for the HPSS metadata. The example in the following discussion creates 5 data volumes as specified below:

SFS Data Volume	LV Type	PPs
hpss_data0	raw	2
hpss_data1	raw	64
hpss_data2	raw	64
hpss_data3	raw	16
hpss_data4	raw	64

   Where the "**sfsvg**" PP size is 32MB, so "**hpss_data1**", "**hpss_data2**", and "**hpss_data4**" data volumes are each 2GB in size. The "**hpss_data0**" and "**hpss_data3**" are 64MB and 512MB, respectively. The allocation of HPSS files in these SFS data volumes using **managesfs** may be created as follows:

   File allocation for the hpss_data0 data volume:

accounting	bfs	cartridge_stk	nsconfig
logclient	logdaemon	logpolicy	migpolicy
mmonitor	mountd	mover	moverdevice
mpchkpt	mps	nfs2	purgepolicy
pvl	pvlactivity	pvldrive	pvljob
pvlpv	pvr	serverconfig	ss
dmg	dmgfileset	acctlog	site
lspolicy	ndcg		

   File allocation for the hpss_data1 data volume:

bfcoschange	bfdiskallocrec	bfdisksegment	bfsssegchkpt
bfssunlink	bftapesegment	bfpurgerec	bitfile
cos	hierarchy	storageclass	bfmigrrec
filefamily	acctsum		

   File allocation for the hpss_data2 data volume:

nsacls	nsobjects	nstext	nsfilesetattrs
nsglobalfilesets			

File allocation for the hpss_data3 data volume:

sspvdisk storagemapdisk storagesegdisk vvdisk

File allocation for the hpss_data4 data volume:

storagesegtape sspvtape storagemaptape vvtape

4.7.2 Configure the Encina SFS server

The following are the procedures to configure the SFS server to manage HPSS metadata. These steps are based on the configuration procedures described in the Encina Administration Guide, *Volumes 1 & 2.*

1. Login as "**root**". Make sure that the login session has only the DCE "**self**" privileges. Use DCE "**klist**" command to display the current **dce_login** information.

2. Create the **/opt/encinalocal/encina/sfs/hpss** and the **/opt/encinamirror/encina/sfs/hpss** directories as follows:

   # mkdir -p /opt/encinalocal/encina/sfs/hpss

   # mkdir -p /opt/encinamirror/encina/sfs/hpss

3. dce login as **cell_admin** to create the principal and group for **encina_admin** and the SFS server.

   # dce_login cell_admin

   Enter Password: XXXXXX

   **# dcecp -c group create encina_admin_group -fullname **

   {"Encina Administrators Group"}

   **# dcecp -c group create encina_servers_group -fullname **

   {"Encina Servers Group"}

   # dcecp -c principal create encina_admin

   # dcecp -c principal create encina/sfs/hpss

   # dcecp -c group add encina_admin_group -member encina_admin

   # dcecp -c group add encina_servers_group -member encina/sfs/hpss

   # dcecp -c organization add none -member encina_admin

   # dcecp -c organization add none -member encina/sfs/hpss

4. Create the "**encina_admin**" and set its password. It should be well protected since it typically has the same power as "**cell_admin**", only in the Encina world. The "**mypwd**" password entry is for "**cell_admin**".

   # dcecp

   **# dcecp> account create encina_admin -group encina_admin_group **

   -organization none -mypwd XXXXXX -password XXXXXXXX

   Use a temporary password like "**foo**" for the server account. The password will be overwritten a couple of steps later.

   **# dcecp> account create encina/sfs/hpss -group encina_servers_group **

   -organization none -mypwd XXXXXXXX -password foo

   # dcecp> quit

   # exit (Purge **cell_admin** context)

5. Create a keytab entry in DCE and the local restart files. Make sure the login session only has the "**root**" and DCE "**self**" privileges for the next two steps.

   **# dcecp -c keytab create /.:/hosts/datastore/config/keytab/hpss **

   **-attr {{storage /opt/encinalocal/encina/sfs/hpss/keyfile} **

   {data {encina/sfs/hpss plain 1 foo}}}

   The following step takes the existing keytab entry and creates a random password that only the server will know ("foo" is now overwritten). It also registers the DCE keytab entry with DCE so the server can run unattended.

   **# dcecp -c keytab add /.:/hosts/datastore/config/keytab/hpss **

   -member encina/sfs/hpss -random -registry

6. Log back in as "**cell_admin**" to create the Encina CDS entries and set up the ACLs for the administrators and the HPSS servers.

   # dce_login cell_admin

   Enter Password: XXXXXX

   # dcecp -c directory create /.:/encina

   # dcecp -c acl modify /.:/encina -add {group encina_admin_group rwidtca}

   # dcecp -c acl modify /.:/encina -add {group encina_servers_group rwidt}

   # dcecp -c acl modify /.:/encina -ic -add {group encina_admin_group rwidtca}

   # dcecp -c acl modify /.:/encina -ic -add {group encina_servers_group rwidt}

 # dcecp -c acl modify /.:/encina -io -add {group encina_admin_group rwidtca}

 # dcecp -c acl modify /.:/encina -io -add {group encina_servers_group rwidt}

7. Sanity checks. Issue the following commands to make sure the ACLs are correct.

 # dcecp -c acl show /.:/encina

{unauthenticated r--t---}
{user cell_admin rwdtcia}
{user hosts/datastore/cds-server rwdtcia}
{group subsys/dce/cds-admin rwdtcia}
{group subsys/dce/cds-server rwdtcia}
{group encina_admin_group rwdtcia}
{group encina_servers_group rwdt-i-}
{any_other r--t---}

 # dcecp -c acl show /.:/encina -ic

{unauthenticated r--t---}
{group subsys/dce/cds-admin rwdtcia}
{group subsys/dce/cds-server rwdtcia}
{group encina_admin_group rwdtcia}
{group encina_servers_group rwdt-i-}
{any_other r--t---}

 # dcecp -c acl show /.:/encina -io

{unauthenticated r--t---}
{group subsys/dce/cds-admin rwdtc--}
{group subsys/dce/cds-server rwdtc--}
{group encina_admin_group rwdtcia}
{group encina_servers_group rwdt-i-}
{any_other r--t---}

8. Create both the “trpc” and “sfs” entries.

 # dcecp -c directory create /.:/encina/trpc

 # dcecp -c directory create /.:/encina/sfs

9. Be sure to use “rti” for the permissions on the “trpc” entry.

 # dcecp -c acl modify /.:/encina/trpc -change {any_other rti}

 # dcecp -c acl show /.:/encina/trpc

{unauthenticated r--t---}

{user cell_admin rwdtcia}
{user hosts/datastore/cds-server rwdtcia}
{group subsys/dce/cds-admin rwdtcia}
{group subsys/dce/cds-server rwdtcia}
{group encina_admin_group rwdtcia}
{group encina_servers_group rwdt-i-}
{any_other r--t-i-}

10. If the previous step returned an error due to a missing **any_other** ACL entry, it will have to be created. Otherwise, skip this step.

    # dcecp -c acl modify /.:/encina/trpc -add {any_other rti}

    # dcecp -c acl show /.:/encina/trpc

    {unauthenticated r--t---}
    {user cell_admin rwdtcia}
    {user hosts/datastore/cds-server rwdtcia}
    {group subsys/dce/cds-admin rwdtcia}
    {group subsys/dce/cds-server rwdtcia}
    {group encina_admin_group rwdtcia}
    {group encina_servers_group rwdt-i-}
    {any_other r--t-i-}

11. Grant **encina_admin** write permission on the DCE clearinghouse.

    # dcecp -c acl modify /.:/datastore_ch -add {user encina_admin w}

    # dcecp -c acl show /.:/datastore_ch

    {unauthenticated r--t-}
    {user encina_admin -w---}
    {group subsys/dce/cds-admin rwdtc}
    {group subsys/dce/cds-server rwdtc}
    {group acct-admin rwdtc}
    {any_other r--t-}

12. Additional sanity checks.

    # dcecp -c directory list /.:/encina -simplename

    sfs trpc

    # dcecp -c account show encina/sfs/hpss

    {acctvalid yes}
    {client yes}
    {created /.../dce.other.org/cell_admin 1998-02-08-16:55:47.000-05:00

```
{description {}}
{dupkey no}
{expdate none}
{forwardabletkt yes}
{goodsince 1998-02-08-16:55:47.000-05:00I-----}
{group encina_servers_group}
{home /}
{lastchange /.../dce.other.org/encina/sfs/hpss 1998-02-08-16:58:
{organization none}
{postdatedtkt no}
{proxiabletkt no}
{pwdvalid yes}
{renewabletkt yes}
{server yes}
{shell {}}
{stdtgtauth yes}
```

```
# dcecp -c keytab catalog /.:/hosts/datastore
```

```
/.../dce.other.org/hosts/datastore/config/keytab/self
/.../dce.other.org/hosts/datastore/config/keytab/hpss
```

```
# dcecp -c keytab show /.:/hosts/datastore/config/keytab/hpss
```

```
{uuid c23788ee-a0cf-11d1-953e-08005a471c9d}
{annotation {}}
{storage /opt/encinalocal/encina/sfs/hpss/keyfile}
{/.../dce.other.org/encina/sfs/hpss des 1}
{/.../dce.other.org/encina/sfs/hpss des 2}
```

```
# exit
```

13. Make sure the login session only has **root** and DCE "**self**" privileges. Use the "**klist**" command to verify the DCE credentials.

4.7.3 Set up SFS Startup files

1. Create an "**/opt/encinalocal/encina/sfs/hpss/rc.encina_cold**" script to manually start an SFS server in cold (administrative) mode. Following is an example of the **rc.encina_cold** file:

```
#!/bin/ksh
ENCINA_DIR=/opt/encinalocal/encina/sfs/hpss
ENCINA_OUT=encina.out
ENCINA_OUT_OLD="${ENCINA_OUT}.`date +%y%m%d.%H%M`"
```

```
# Disable TCP; use UDP only
export RPC_SUPPORTED_PROTSEQS=ncadg_ip_udp
# Change directory to pickup files and for placement of core file
cd ${ENCINA_DIR}
# Save the previous output file
mv ${ENCINA_OUT} ${ENCINA_OUT_OLD}
# Start SFS in cold mode
/usr/bin/sfs -n /.:/encina/sfs/hpss \
        -p encina/sfs/hpss \
        -k keyfile \
        -N 2 \
        -Z 1 \
        -v restart:/opt/encinamirror/encina/sfs/hpss/restart.bak \
        -A encina_admin > encina.out 2>&1 &
```

2. Create an "**/opt/encinalocal/encina/sfs/hpss/rc.encina**" script to manually start an SFS server in warm (normal) mode. Following is an example of the **rc.encina** file:

```
#!/bin/ksh
ENCINA_DIR=/opt/encinalocal/encina/sfs/hpss
ENCINA_OUT=encina.out
ENCINA_OUT_OLD="${ENCINA_OUT}.`date +%y%m%d.%H%M`"
# Disable TCP; use UDP only
export RPC_SUPPORTED_PROTSEQS=ncadg_ip_udp
# Change directory to pickup files and for placement of core file
cd ${ENCINA_DIR}
# Save the previous output file
mv ${ENCINA_OUT} ${ENCINA_OUT_OLD}
# Start SFS in warm mode
/usr/bin/sfs -n /.:/encina/sfs/hpss \
        -p encina/sfs/hpss \
        -k keyfile \
        -l sfslogvol/sfslogfile \
        -N 2 \
        -Z 1 \
        -v restart:/opt/encinamirror/encina/sfs/hpss/restart.bak \
        -b 4096 \
        -c en_US \
        -i 60:5000:100000 \
        -d 60 \
```

-P 375:10 > encina.out 2>&1 &

If desired, define the ENCINA_TK_SERVER and ENCINA_SFS_SERVER variables in the /etc/environment file. The machine must be rebooted for the "/etc/environment" changes to take effect!

4.7.4 Start SFS server in "cold" Mode

1. Use the **rc.encina_cold** script to start the SFS server in "**cold**" mode to run with exclusive authority for "**encina_admin**" as the only user.

   # cd /opt/encinalocal/encina/sfs/hpss

   # rc.encina_cold

2. View the Encina SFS output file to ensure that SFS is ready for additional configuration. Press the **Control-C** keys to quit the tail session.

   # tail -f /opt/encinalocal/encina/sfs/hpss/encina.out

   1 13452 98/02/08-17:29:32.253494 502c3468 A Ready for additional configuration
   ... Sun Nov 8 17:29:32 EST 1998

4.7.5 Create the SFS Log Volume and Log File

1. Create and initialize the SFS log volume. Note that it is highly recommended that the SFS log volume be created on a mirrored or RAID disk. Assuming that the PP size of the volume group (**sfsvg**) is 32 MB and the log file to be created is 128 MB, issue the commands as follows.

   # dce_login encina_admin

   Enter Password: XXXXXXXX

   # mklv -y'sfsloglv' -t'raw' sfsvg 4 hdisk5

   # export ENCINA_TK_SERVER=/.:/encina/sfs/hpss

   # export ENCINA_SFS_SERVER=/.:/encina/sfs/hpss

   # tkadmin map lvol sfslogvol sfsloglv 64

   # mkdir /opt/encinalocal/encina/sfs/hpss/archives

   **# tkadmin init logvol sfslogvol **

   FILE:/opt/encinalocal/encina/sfs/hpss/archives

Make sure that there is sufficient space in the /opt/encinalocal/encina/sfs/hpss/archives directory to store the TRB files. Refer to Section 4.7.1 for more information.

2. Create the SFS log file in the SFS log volume then enable it so the SFS server can start using it.

   # tkadmin create logfile sfslogvol/sfslogfile

   # tkadmin enable logfile sfslogvol/sfslogfile

   # tkadmin recover lvols

   # tkadmin enable mediaarchiving

   # tkadmin enable server

4.7.6 Set Up ACLs for SFS Server

The SFS server is now up and running, but don't exit out until the ACLs are set; otherwise the set up procedures will have to be started all over. The last ACL for "**self**" is not mandatory, but does allow a site to run automated SFS backups from **cron** without DCE credentials. (The files created with "**managesfs**" will also need to have permissions set for "self" to run backups.)

 # dcecp -c acl modify /.:/encina/sfs/hpss -add '{group encina_admin_group ACQ}'

 # dcecp -c acl modify /.:/encina/sfs/hpss -add '{group hpss_server ACQ}'

 # dcecp -c acl modify /.:/encina/sfs/hpss -add '{user encina_admin ACQ}'

 # dcecp -c acl modify /.:/encina/sfs/hpss -add '{user hosts/<host_name>/self ACQ}'

Note that **<host_name>** from the last **dcecp** command should be replaced by the name of the host machine.

4.7.7 Configure SFS Data Volumes

1. Create the 5 SFS data volumes defined in the planning sections. It is recommended that each SFS data volume be created on a separate disk.

   # mklv -y'hpss_data0' -t'raw' sfsvg 2 hdisk0

   # mklv -y'hpss_data1' -t'raw' sfsvg 64 hdisk1

   # mklv -y'hpss_data2' -t'raw' sfsvg 64 hdisk2

   # mklv -y'hpss_data3' -t'raw' sfsvg 16 hdisk3

   # mklv -y'hpss_data4' -t'raw' sfsvg 64 hdisk4

2. Map these volumes to Encina

   # tkadmin map lvol hpss_data0 sfs_hpss_data0 64

   # tkadmin map lvol hpss_data1 sfs_hpss_data1 64

 # tkadmin map lvol hpss_data2 sfs_hpss_data2 64

 # tkadmin map lvol hpss_data3 sfs_hpss_data3 64

 # tkadmin map lvol hpss_data4 sfs_hpss_data4 64

3. Enable SFS data volumes for Encina

   # tkadmin enable lvol sfs_hpss_data0

   # tkadmin enable lvol sfs_hpss_data1

   # tkadmin enable lvol sfs_hpss_data2

   # tkadmin enable lvol sfs_hpss_data3

   # tkadmin enable lvol sfs_hpss_data4

4. Let SFS know about the SFS data volumes

   # sfsadmin add lvol sfs_hpss_data0

   # sfsadmin add lvol sfs_hpss_data1

   # sfsadmin add lvol sfs_hpss_data2

   # sfsadmin add lvol sfs_hpss_data3

   # sfsadmin add lvol sfs_hpss_data4

5. Clear the exclusive authority setting and shut down the SFS server.

   # tkadmin clear exclusiveauthority

   # tkadmin stop server

4.7.8 Starting the SFS Server in "Warm" Mode

1. Use the **rc.encina** script to start the SFS server in "**warm**" mode.

   # cd /opt/encinalocal/encina/sfs/hpss

   # rc.encina

2. View the Encina SFS output file to ensure that SFS initialized successfully. Press the **Control-C** keys to quit the tail session.

   # tail -f /opt/encinalocal/encina/sfs/hpss/encina.out

   19 13452 98/02/08-17:41:40.616185 502c5448 A Initialized ... Sun Nov 8 17:41:40 EST 1998

4.7.9 Verify the SFS Server Configuration

Perform the following verification to ensure that the SFS server was configured correctly:

 # dce_login encina_admin

Enter Password: XXXXXXXX

 # sfsadmin list lvol

Volumes:
  sfs_hpss_data0
  sfs_hpss_data1
  sfs_hpss_data2
  sfs_hpss_data3
  sfs_hpss_data4

 # sfsadmin list files

No files.

 # sfsadmin query server

Collating language: (null)
Log file name: sfslogvol/sfslogfile
Buffer pool size: 4096
TRPC authentication level: 1
Minimum authentication level: 2
Checkpoint Interval Info:
  Interval time in seconds: 60
  Minimum number of log records: 5000
  Maximum number of log records: 100000
Default idle timeout: 60

 # exit

4.7.10 Create SFS metadata files required by HPSS

At this point, the HPSS metadata files can be created using the **managesfs** utility. Refer to Section I.36 for more information about **managesfs**.

When multiple SFS data volumes are configured, the administrator must decide which SFS metadata files will reside on which SFS data volume. An SFS metadata file cannot span SFS data volumes.When using managesfs to create SFS metadata files, the administrator must explicitly specify the SFS Data Volume name prior to creating files to ensure that the files are created on the correct volume.

4.8 Customize DCE Keytabs Files

As part of the HPSS configuration with DCE, the **mkhpss** script created the following default DCE principals:

- hpss_bfs
- hpss_client_api
- hpss_cns
- hpss_dmg
- hpss_ftpd
- hpss_log
- hpss_ls
- hpss_mm
- hpss_mountd
- hpss_mps
- hpss_mvr
- hpss_ndcg
- hpss_nfs2
- hpss_pvl
- hpss_pvr
- hpss_ssm
- hpss_ss

To adhere to the site local password policy, these principals were created with known keys and subsequently changed with randomized keys. The HPSS keytab files (**/krb5/hpss.keytabs** and **/krb5/hpssclient.keytab**) hold both versions of the keys; however, the DCE Registry holds only the randomized keys.

Periodically, the HPSS administrator should change the passwords in the two HPSS keytab files.

The procedure to change the passwords is as follows:

1. List the contents of the two keytab files:
    - Become root.
    - **dce_login** as an entity allowed access to the DCE Registry for changing passwords, usually **cell_admin**.
    - Issue the commands:

        % rgy_edit

rgy_edit=> ktl -f /krb5/hpss.keytabs

rgy_edit=> ktl -f /krb5/hpssclient.keytab

2. Update the keytab files:

   - For each entry in /krb5/hpss.keytabs do:

     rgy_edit=> kta -f /krb5/hpss.keytabs -p <entry_name> -r -a

   - For each entry in /krb5/hpssclient.keytab do:

     rgy_edit=> kta -f /krb5/hpssclient.keytab -p <entry_name> -r -a

     where **<entry_name>** refers to an entry in the keytab file; e.g., hpss_ssm.

3. See the discussion immediately following this step! Propagate the resulting keytab files to every HPSS server machine. Note that the most secure mechanism for performing this is "**footnet**". If FTP is used, be sure to specify the "**bin**" option. The keytab files on every HPSS system should have the following ownership and permissions set:

   /krb5/hpss.keytabs hpss hpss rw- rw- ---

   /krb5/hpssclient.keytab hpss hpss rw- rw- ---

It is strongly recommended that both keytab files be generated on a single HPSS server machine and securely propagated to every other HPSS server machine; however, a customer may prefer to create appropriate keytab files which contain only the entries required for a specific HPSS server machine. This, however, is strongly discouraged because it can create a "Catch 22" condition in which the encryption keys on one or more HPSS systems cannot be set to match the keys stored in the DCE Registry!

If a customized keytab file is used on every different HPSS server system, steps 1 and 2 above must be performed on each system.

If the key for a server on one machine is changed, do not change the key on another machine since this will de-synchronize the entry on the first system changed!

Chapter 5

HPSS Configuration

5.1 Overview

This chapter provides instructions for creating the configuration data to be used by the HPSS servers. This includes creating the server configuration, defining the storage policies, defining the storage characteristics and creating the storage space. The configuration data can be created, viewed, modified, or deleted using the HPSS SSM GUI windows. In addition to using SSM to configure the HPSS servers, some UNIX configuration will also need to be performed. Refer to Appendix H for more information on how to use SSM. The UNIX configuration required by HPSS is discussed in Section 5.8.

To create or modify the HPSS configuration data, we recommend that the administrator be familiar with the information described for the HPSS planning process as documented in Chapter 2. In addition, we recommend that the planning worksheets provided in the Appendix C be used to document the decisions made during the HPSS planning process. The procedures described in this chapter assume that the HPSS installation and infrastructure configuration have been completed.

5.1.1 HPSS Configuration Roadmap

The following steps summarize the configuration for the HPSS system. It is very important that the steps be performed in the order listed. Each step is discussed in more detail in the referenced section.

1. Configure HPSS (Sections 5.2 through 5.7)
   - Configure and start up SSM (Section 5.2)
   - Create a basic configuration entry for each HPSS server (Section 5.3)
   - Configure HPSS storage policies (Section 5.4)
   - Configure HPSS storage characteristics (Section 5.5)
   - Create a specific configuration entry for each HPSS server (Section 5.6)
   - Configure MVR devices and PVL drives (Section 5.7)
2. Configure additional UNIX data (Section 5.8)

3. Start up the HPSS servers (Section 5.9)
4. Unlock the PVL drives (Section 5.9.2)
5. Create HPSS storage space (Section 5.9.3)
   - Import volumes into HPSS (Section 6.4.1.1)
   - Create Storage Server resources (Section 6.4.1.2)
6. Create Filesets and Junctions (Section 6.5.1)

5.1.2 HPSS Configuration Limits

The following configuration limits are imposed by SSM and / or the HPSS servers:

Server

- Maximum number of HPSS servers: 1024

Storage Policy

- Total Accounting Policies: 1
- Total Migration Policies: 64
- Total Purge Policies 64

Storage Characteristic

- Total Storage Classes: 384
- Total Storage Hierarchies: 64
- Total Classes Of Service: 64
- Maximum File Families: 1024

Mover Device/PVL Drive

- Maximum Devices / Drives: Unlimited
- Total Devices per Mover: 64

Storage Space

- Import: 1000 cartridges per SSM import request

- Export: 1000 cartridges per SSM export request
- Create: 1000 virtual volumes per SSM create request
- Delete: 1000 physical volumes per SSM delete request

5.1.3 Using SSM for HPSS Configuration

The HPSS server and resource configuration data may be created, viewed, updated, or deleted using the SSM windows. The configuration data are kept in Encina SFS files.

When you submit a request to configure a new server, SSM displays all fields with the appropriate default data. If the default values were not used during the installation and the infrastructure configuration phases or are not desired, type over the displayed data with the desired values. Press the **Enter** key after entering the data so that the new value is read by SSM.

As each server general configuration entry is created, SSM keeps track of the configured server to aid the user in defining the remainder of the HPSS configuration and to allow the user to monitor and manage the servers during the HPSS operational phase.

This guide presents and describes the configuration data displayed in the SSM configuration windows in the form of tables. Each table lists the SSM window display field names and, for each field, gives a description of the variable, acceptable values for the variable, and the default value of the variable if one exists.

Many of the fields in the tables have an advice box that provides additional information about the field.

Using the Help Window

All of the SSM windows provide a pop-up help window that contains detailed information about the fields on the window. To display the help window, move the cursor to any blank space on the window, hold down the **Shift** key, and click the right-most mouse button.

Using the HPSS Configuration Windows

All HPSS configuration data can be created, viewed, updated, and deleted using the appropriate HPSS Configuration windows. You obtain these windows from the HPSS Health and Status window that appears when you log on to the system (as discussed in Section 5.2.2). When the HPSS Health and Status window (shown in Figure 5-1) appears, click on the **Admin** menu, select the **Configure HPSS** option and click on the appropriate configuration option. The desired configuration window will be displayed. Refer to the appropriate sections in this chapter for more information on how to use the HPSS configuration windows to configure an HPSS system.

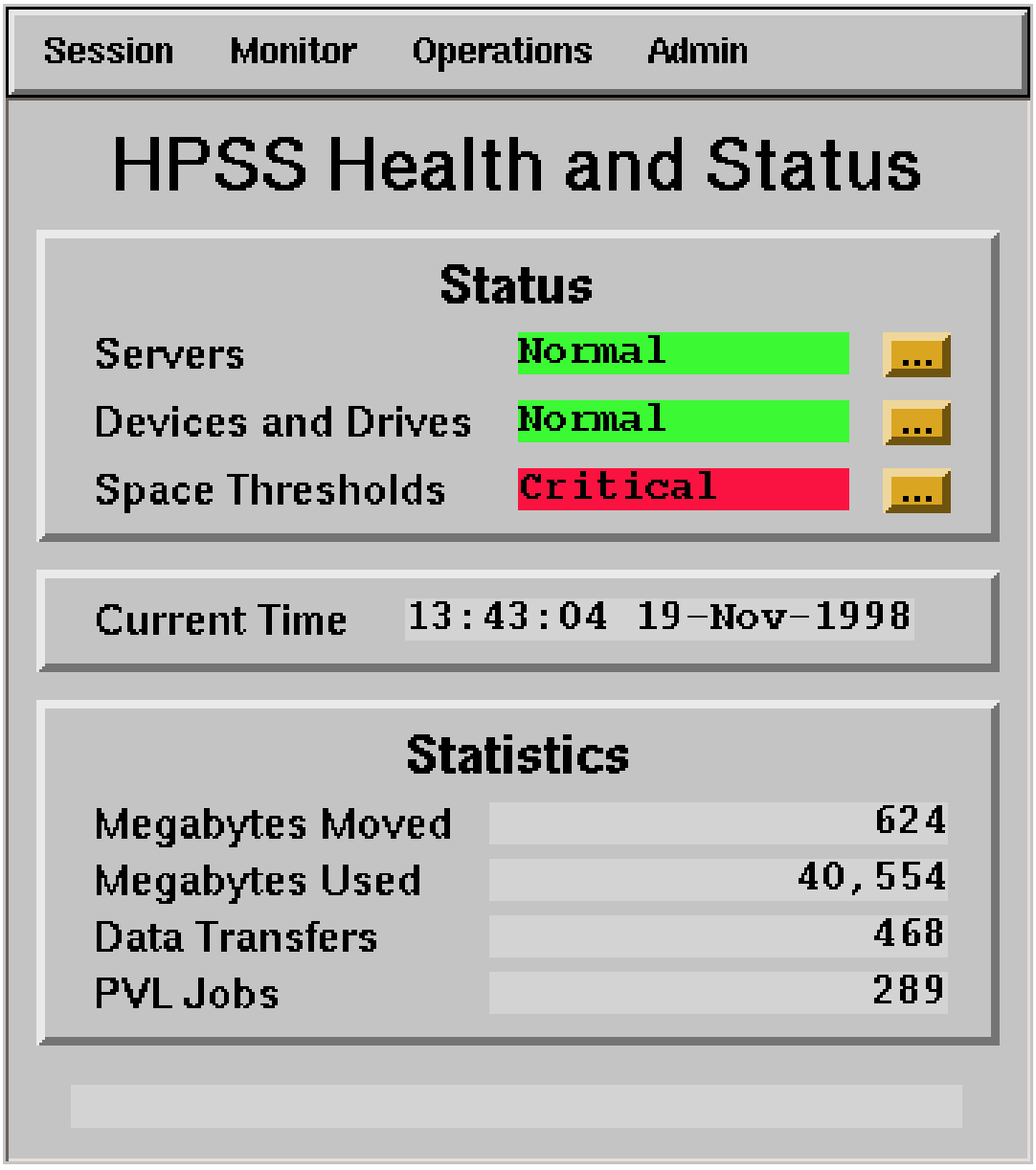


Figure 5-1 HPSS Health and Status Window

5.1.4 Server Reconfiguration and Reinitialization

HPSS servers read their respective configuration file entries during startup to set their initial running conditions. Note that modifying a configuration file while a server is running does not immediately change the running condition of the server. Servers will need to perform a reinitialization or restart to read any newly modified configuration data. This reinitialization is usually accomplished by stopping and restarting the server. For servers that have the ability to reinitialize without restarting (i.e., the Metadata Monitor, Log Daemon, and Log Client) this can be accomplished by using the **Reinitialize Server** feature in the appropriate SSM windows.

5.2 SSM Configuration and Startup

If the SSM servers and the SSM administrative session were configured and started up during the HPSS infrastructure configuration phase, the administrator does not have to perform the steps described in this section. However, the information described in this section will be useful for the subsequent restart of the SSM servers or for the startup of other SSM sessions.

The SSM configuration consists of creating the configuration entry for the SSM System Manager, starting up the SSM servers, and bringing up one or more SSM user sessions. An SSM server startup script, **start_ssm**, is provided to bring up the SSM System Manager and the SSM Data Server. Another provided script, **start_ssm_session**, can be used to bring up an SSM user session.

5.2.1 SSM Server Configuration and Startup

The SSM System Manager will automatically create an SSM configuration entry, if one does not already exist, using the environment variables defined in the **/usr/lpp/hpss/config/hpss_env** file. This configuration entry may later be modified, if necessary, by the administrator. The SSM Data Server also uses the environment variables defined in the **/usr/lpp/hpss/config/hpss_env** file, but does not require a configuration entry. Refer to Table 4-1 for more information on the SSM variables.

The SSM server startup script, **start_ssm**, as supplied with HPSS can be used to invoke the SSM System Manager and the SSM Data Server. Both servers require a number of environment variables to be used as their command line arguments and to be used to configure and manage the HPSS servers. These variables are defined in the **/usr/lpp/hpss/config/hpss_env** file. They should have been edited to reflect the site's configuration during the HPSS infrastructure configuration phase.

To start up the SSM servers, invoke the **start_ssm** script. The System Manager will be brought up first, followed by the Data Server. The **start_ssm** script can also be invoked with the **-s** (start the System Manager only) option or the **-d** (start the Data Server only) option. Once the SSM servers are up and running, one or more SSM sessions can be started to manage HPSS.

Before starting up the SSM servers, ensure that DCE, Encina, and Sun RPC servers are up and running.

The **start_ssm** script will not start an SSM server if that server is already running on the same node. Ensure that only one copy of each configured SSM System Manager and Data Server is running at any one time. If there are multiple SSM servers running with the same configuration or environment data, there may be data loss for the SSM windows. Typically, there will be only one System Manager and one Data Server configured and both are run on the same node. If this setup is not desired, refer to Appendix H (Section H.6) for more information on other possible configurations.

5.2.2 SSM User Session Configuration and Startup

An SSM administrator User ID should have been already created as part of the HPSS infrastructure configuration phase. If the User ID does not exist, refer to Section 6.12.1 for instructions on adding users and creating an SSM account before proceeding with this section.

SSM supports the following SSM security levels:

1. **admin.** This security level is normally assigned to an HPSS administrator. This SSM user can view all SSM windows and perform all control functions provided by SSM.

2. **operator.** This security level is normally assigned to an HPSS operator. This SSM user can view all SSM windows and perform all SSM control functions except for changing the HPSS configuration.

3. **privileged.** This security level is normally assigned to a privileged user such as an HPSS system analyst. This SSM user can view most of the SSM windows but cannot perform any SSM control functions.

4. **user.** This security level is normally assigned to an user who may have a need to monitor some HPSS functions. This user can view a limited set of the SSM windows but cannot perform any of the SSM control functions.

Refer to Appendix H for more information on setting up an SSM account and the list of SSM windows accessible to each SSM user type.

Before bringing up the SSM session, ensure that the Sammi license key provided by Kinesix has been properly set up. If it is not, refer to Section 3.4.3 for instructions on setting up the Sammi license.

In addition to the above, the SSM session may also need a Motif key bindings file to ensure that certain keyboard keys are defined in specific ways expected by Sammi. The key bindings file is named **.motifbind** and must be placed in the user's home directory on the host where the user will be displaying the SSM windows. Refer to Appendix H for more information on how to set up the Motif key bindings file.

The SSM user session **start_ssm_session** script as supplied with HPSS can be used to bring up any configured SSM session. To start up an SSM user session, invoke the startup script, **start_ssm_session,** and enter the requested data when prompted. (The **start_ssm_session** script will not start an SSM session if that session is already up and running.) After a brief interval, the SSM Logon window (shown in Figure 5-2) will be displayed to allow the user to log on. Enter the valid DCE **User ID** and **Password** before pressing the **Enter** key. After the user is authenticated by DCE, the HPSS Health and Status window will be displayed. From this window, the user can proceed to perform the HPSS configuration.

Before starting up the SSM user session, ensure that the Sun RPC servers and the SSM servers are up and running.

There is no limit on the number of SSM sessions that can be brought up at the same time. However, depending on the system workload, too many running SSM sessions may degrade the overall system performance.

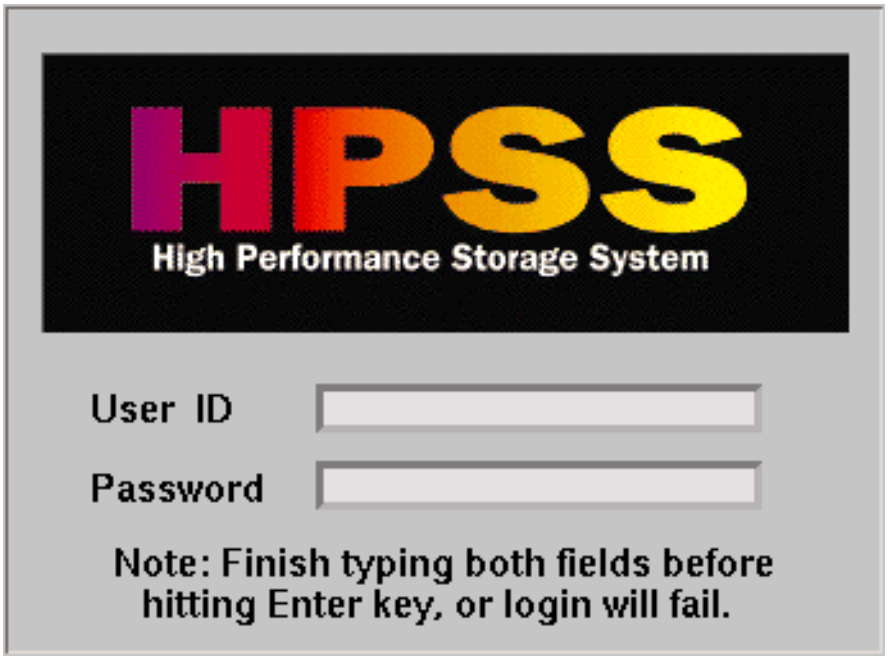


Figure 5-2 HPSS Logon Window

5.3 Basic Server Configuration

All HPSS servers use a similar metadata structure for the basic server configuration. A basic server configuration entry must be created for each of the following server types:

- Name Server
- Bitfile Server
- Storage Server
- Migration / Purge Server
- Location Server
- DMAP Gateway
- Physical Volume Library
- Physical Volume Repository
- Mover
- Log Daemon
- Log Client

- Metadata Monitor
- NFS Daemon
- NFS Mount Daemon
- HPSS Startup Daemon
- Non-DCE Client Gateway

Before configuring the HPSS servers, the planning guidelines for the HPSS servers as described in Section 2.6 should be carefully considered. The worksheets in Appendix C should be used to document the essential planning data for the to-be-configured servers to aid with the upcoming configuration tasks.

The SSM System Manager basic configuration entry is usually created when the server is started up for the first time using the environment variables defined in the /usr/lpp/hpss/config/hpss_env file. If the default values are not appropriate for the site's configuration, the administrator should modify them using the Basic Server Configuration window described in the following section and restart the System Manager rather than re-create a new entry. Since each configured HPSS server references the SSM ID in its basic configuration entry, the SSM configuration entry should not be deleted after the HPSS servers are configured; otherwise, the servers cannot communicate with SSM. In addition, it is recommended that only one SSM System Manager configuration entry be created to avoid possible loss of communication between the HPSS servers and SSM.

5.3.1 Configure the Basic Server Information

A basic server configuration entry can be created using the Basic Server Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through this window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 5-3.

To configure a new server, click on the **New Server...** button on the HPSS Servers window. A blank Basic Server Configuration window is displayed along with the server-type popup list. Select the desired type of server and the Basic Server Configuration window will be filled in as shown in Figure 5-4 with default values. If the default data is not desired, change the field with the desired value. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the desired server entry on the HPSS Servers window and click on the **Basic...** button from the **Configuration** button group. The Basic Server Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the desired server displayed on the HPSS Servers window and click on the **Basic...** button from the **Configuration** button group. The Basic Server Configuration window will be displayed with the configured data. Verify that the associated server- specific configuration, if exists, is already deleted then click on the **Delete** button to delete the basic configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

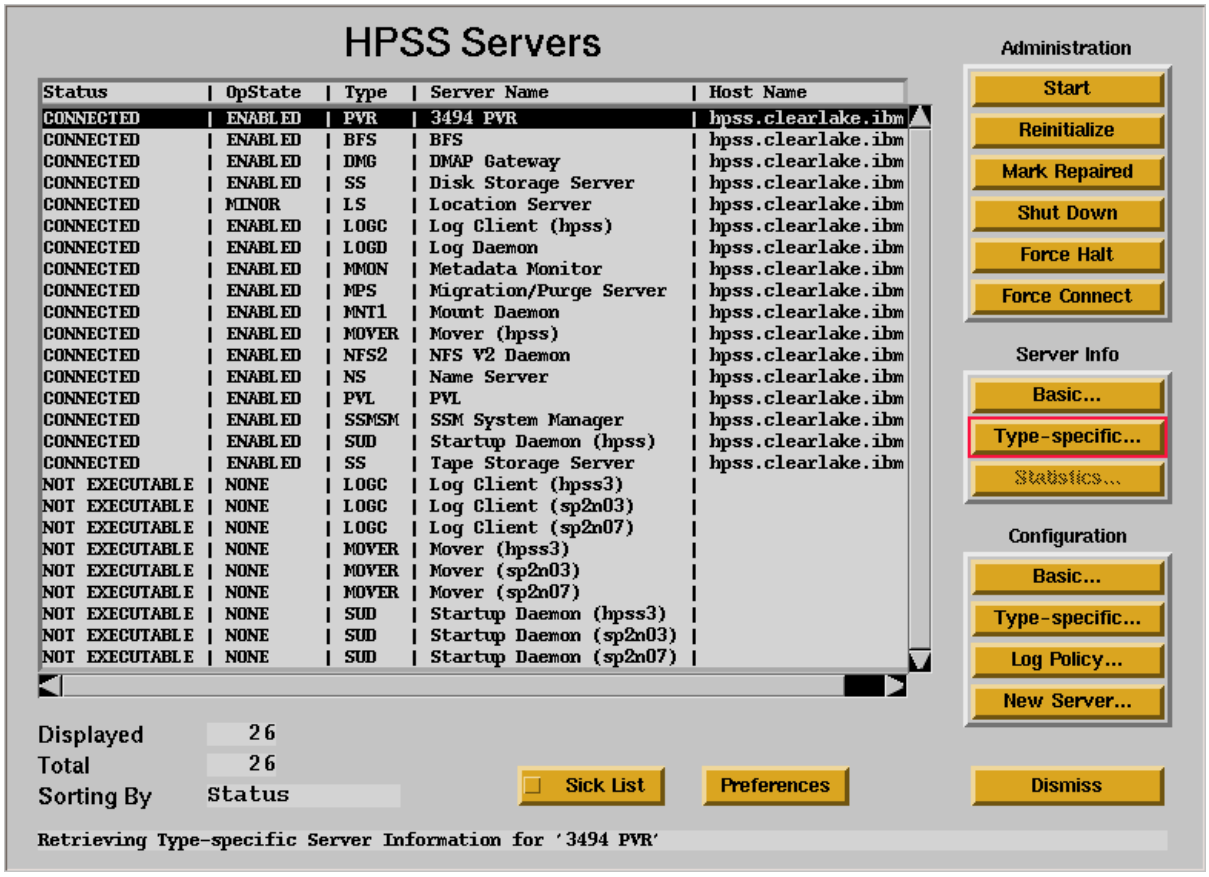


Figure 5-3 HPSS Servers Window

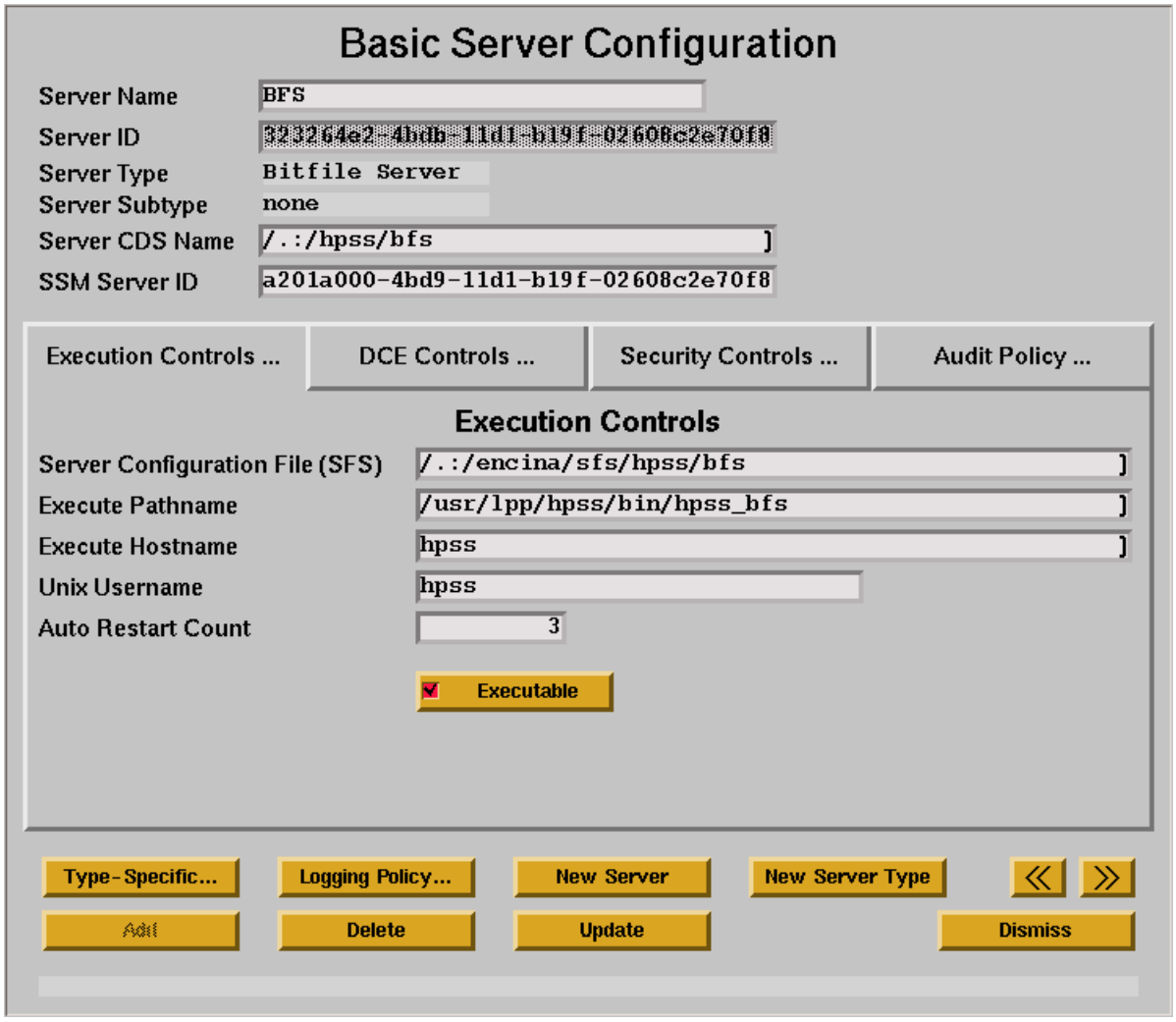


Figure 5-4 Basic Server Configuration Window

Basic Server Configuration Variables

The fields in the Server Configuration window describe information necessary for servers to successfully operate. The fields also contain information that is necessary for successful interaction with the SSM component.

In addition, each HPSS basic server configuration includes Security Information and Audit Policy fields that determine the server's security environment. The security of the HPSS system will be a result of how these fields are set. HPSS defaults for the Security Information are set to provide authenticated communication between servers, authorization of clients to HPSS interfaces and objects, enforcement of authorization permissions, and audit of authentication and file operation failures.

The fields in the Basic Server Configuration window are grouped into five categories:

- General Information fields
- Execution Controls fields
- DCE Controls fields
- Security Controls fields
- Audit Policy fields

To save window space, the last four categories are presented in "layers", and each layer has its name displayed on a "tab". To access a different layer, click on the appropriate tab. Table 5-1 lists the fields on the Basic Server Configuration window in the approximate order that they appear on the window.

Table 5-1 Basic Server Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
General Information. The following six fields describe the server's general information.			
Server Name	The descriptive name used to look up a server's configuration parameters in Encina SFS. As with UUIDs, each descriptive name in HPSS must be unique.	Any character string up to 31 bytes in length.	Based on the server type and the number of servers of that type that already exist.
	Advice: Ensure that the Descriptive Name is unique. A server's descriptive name should be meaningful to local site administrators and operators, in contrast to the server's corresponding UUID, which has meaning for DCE and HPSS.		
Server ID	A unique ID used to distinguish servers in the HPSS. Each server must have a unique Server ID.	Any valid non-null UUID.	A new and unique ID is generated by SSM during the configuration process as a default.
	Advice: Ensure that each server has a unique Server ID. If a server ID is not unique, SSM will not be able to distinguish servers and will not be able to correctly map between human-oriented descriptive names and machine-oriented UUIDs for display purposes. UUIDs are meaningful primarily to the storage system and DCE. Normally, they will not need to be directly specified or changed from SSM.		
Server Type	The type of HPSS server.	This field is filled in with the server type selected by the user as part of the window's selection. It is not changeable.	Server type selected by the user.

Table 5-1 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Server CDS Name	The name of a Cell Directory Service (CDS) directory in which a server creates additional CDS objects and registers its endpoint information. Every server must have a different directory.	Any legal CDS directory name is permitted. The directory must exist when this particular server is started.	Based on the selected server type and number of servers of that type that already exist.
	Advice: (1) Different names must be used for each server. If this is not observed, servers will have trouble contacting each other, or will talk to different servers than anticipated. (2) The directories must be writable by the process that uses them. (3) It is useful to have a convention for naming these directories. For example, all directories might be subdirectories of a common directory such as **/.:/hpss.** The directories might be named by the role that the corresponding server plays in the CDS—e.g., **/.:/hpss/disk_ss** and **/.:/hpss/tape_ss** might be the names used by two instances of the Storage Server. (4) If there are multiple Startup Daemons, Movers, and Log Clients running, use names that reflect the hosts they are running on. For example, a Startup Daemon running on a host named **timelord** might be named **/.:/hpss/hpssd_timelord.** (5) HPSS does not make any assumptions about which names are used. (6) SSM creates the specified CDS directory and the associated CDS Security Object when the server is configured from this window. The parent directory (e.g. **/.:/hpss**) must already exist, otherwise the creates will fail. SSM also creates the default ACLs for the server's CDS directory and the CDS Security object as follows: CDS Directory (for all servers:) **unauthenticated:r--t-** **user:hosts/sp2n06/cds-server:rwdtc** **user:hpss_ssm:rwdtc** **group:subsys/dce/cds-admin:rwdtc** **group:subsys/dce/cds-server:rwdtc** **group:hpss_server:rwdtc** **any_other:r--** CDS Security object (for all servers): **unauthenticated:r--t-** **user:hpss_ssm:rwdtc** **group:subsys/dce/cds-admin:rwdtc** **group:subsys/dce/cds-server:rwdtc** **group:hpss_server:rwdt- (not defined for NS and BFS)** **any_other:r--t-** Additional ACL for the BFS Security object: **user:hpss_mps:rwdtc** **user:hpss_nfs2:rwdtc** **user:hpss_dmg:rwdtc** **user:hpss_cns:rwdtc** Additional ACL for the NS Security object: **user:hpss_dmg:rwdtc** **user:hpss_ftp:r-dtc** **user:hpss_bfs:r-dtc** **user:hpss_nfs2:r-dtc**		

Table 5-1 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Server Subtype	The subtype of the selected server.	This field is filled in with the server subtype selected by the user as part of the window's selection. It is not changeable.	Server subtype selected by the user.
SSM Server ID	The Server ID that SSM is running under. The field is used by the server to validate that requests are coming from the SSM.	Any valid non-null SSM UUID.	The first SSM UUID found in the /.:/encina/sfs/hpss/serverconfig file.
	Advice: For the server to communicate with SSM, it must be configured with the proper SSM ID. There should be only one SSM Server ID created in the HPSS system.		
Execution Controls. The following five fields describe the server's execution control configuration.			
Server Configuration File (SFS)	The name of the Encina SFS file containing configuration data for this server.	Any valid SFS filename.	Based on the selected server type.
Execute Pathname	The UNIX file system path name to a server's executable. If servers are running on several hosts, the name must be the name of a file on the host that is running the server.	Any legal UNIX file name.	/usr/lpp/hpss/bin/<server-specific executable name>. Based on the selected server type.
	Advice: Use the full UNIX path name; otherwise, the Startup Daemon will try to start the file out of the current working directory of the Daemon.		
Execute Hostname	The hostname on which a particular server is to run. SSM uses this field to locate the Startup Daemon that will execute the server.	Any legal hostname, such as a name that might be obtained using the UNIX **hostname** command.	Local hostname.
	Advice: A Startup Daemon must be running on this host at the time when the server is to be started. This field must match exactly the Execute Hostname field specified in the configuration of the Startup Daemon running in the same host. For Movers which will execute in non-DCE mode, this field should contain the hostname corresponding to the node on which the Mover DCE/Encina processes will run.		

Table 5-1 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
UNIX Username	The UNIX user name that the server runs under. If servers are running on several hosts, the name must be the name that is used on the host that the server will run on. The name must be registered in the local UNIX authentication database (e.g., **/etc/passwd**).	Any legal user name for the host that the server runs on.	**root** for the Startup Daemon, the NFS V2 Damon, and the NFS Mount Daemon. **hpss** for all other servers (see Advice below).
	Advice: If there is no such user, the Startup Daemon will not be able to start the servers. The Startup Daemon, NFS Daemon and the NFS Mount Daemon should also have their UNIX Usernames set to **root**.		
Auto Restart Count	The HPSS Startup Daemon uses this field to control automatic server restarts. If the server shuts down unexpectedly, the Startup Daemon will restart the server, without any intervention by SSM, up to this many times; after that, the Startup Daemon will not restart it again. If the server runs for more than an hour without failing, the Startup Daemon will set its failure count for that server back to zero. A negative value in this field will be interpreted as an infinite count; the Startup Daemon will always restart the server if it fails.	Any 32-bit integer value.	3
Executable	A flag that indicates whether a HPSS server can be started up by SSM.	ON, OFF	ON
DCE Controls. The following two fields describe the server's DCE controls information.			

Table 5-1 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Maximum Connections	The highest number of connection contexts this server can establish.	Any positive 32-bit integer value.	NS - 200 BFS - 200 SS - 20 MPS - 10 PVL - 20 PVR - 20 MVR - 20 Log Daemon- 10 Log Client - 10 NFS Daemon - 20 Mount Daemon - 10 MMON - 10 SSM - 100 HPSS Startup Daemon - 10 DMG- 100 NDCG -20
	Advice: This value should be set based on the anticipated number of concurrent clients. Too large a value may slow down the system.		

Table 5-1 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Thread Pool Size	The highest number of threads this server can spawn in order to handle concurrent requests.	Any positive 32-bit integer value.	NS - 200 BFS - 200 SS - 50 MPS - 10 LS - 20 DMG - 100 PVL - 100 PVR - 30 MVR - 20 Log Daemon- 10 Log Client - 10 NFS Daemon - 20 Mount Daemon - 10 MMON - 10 SSM - 100 HPSS Startup Daemon - 10 NDCG - 20
	Advice: If necessary, the default values can be changed when defining servers. Too large a value may slow down the system. The Thread Pool Size should be equal to or larger than the value used for Maximum Connections. The SSM System Manager does not use this field. It uses the **ENCINA_TPOOL _SIZE** environment variable defined in the **/usr/lpp/hpss/config/hpss_env** file. The **ENCINA_TPOOL_SIZE** variable is initially set to 100 for the System Manager and can be changed if desired. The number of threads for MPS can be calculated as follows: 1 + (# of disk storage classes) * (2 + copy count from Migration Policy) + (# of tape storage classes) * (1 + total # Storage Servers in this storage class). Ensure that the value of the **HPSS_SFS_THREADS** environment variable defined in the **/usr/lpp/hpss/config/hpss_env** file is greater than the Thread Pool Size value defined for the Bitfile Server and the Name Server or Encina SFS may run out of threads.		
Security Controls. The following eight fields describe the server's security configuration.			

Table 5-1 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Principal Name	The DCE principal name defined for the server during the infrastructure configuration phase.	The principal name must exist in the DCE registry.	NS: hpss_cns BFS: hpss_nfs SS: hpss_ss MPS: hpss_mps LS: hpss_ls DMG: hpss_dmg PVL: hpss_pvl PVR: hpss_pvr MVR: hpss_mvr Log Daemon: hpss_log Log Client: hpss_log NFS Daemon: hpss_nfsd Mount Daemon: hpss_mntd MMON: hpss_mmon SSM: hpss_ssm HPSS Startup Daemon: hpss_hpssd
	Advice: Ensure that the key table named by the **Keytab Pathname** field contains an entry for this principal; otherwise, authentication will fail.		
Protection Level	The amount of encryption that will be used in communication with peer applications.	Default, None, Connect, Call, Packet, Packet Integrity, Packet Privacy	Connect
	Advice: The higher the level of protection, the more encryption and overhead required in communications with peers. A level of **None** corresponds to no authentication and no authorization. The minimum suggested level is **Connect**. Refer to the description of the DCE Protection Levels included in the **rpc_binding_set_auth_info** call in the *OSF/DCE Application Development Reference Manual*.		

Table 5-1 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Authorization Service	The authorization service to use when passing identity information in communications to HPSS components.	None, Name, DCE	DCE
	Advice: The recommended authorization server is **DCE**. This ensures that the most complete identity information about the client is sent to the server. If **None** is used, anyone is authorized to call the HPSS components. This means you are in a trusted environment. If the Authentication Service value is **None**, the only acceptable Authorization Service value is **None**. If the Authentication Service value is **Secret**, the Authorization Service value can be **Name** or **DCE**. Refer to the description of the DCE Authorization Services included in th**e rpc_binding_set_auth_info** call in the *OSF/DCE Application Development Reference Manual*.		
Authentication Service	The authentication service to use in communications.	None, Secret, Public, Default	Secret
	Advice: **Secret** provides for authenticated communications between HPSS components. The authentication service is the basis for all HPSS security. Refer to the description of the DCE Authentication Services included in the **rpc_binding_set_auth_info** call in the *OSF/DCE Application Development Reference Manual*.		
Security Site Name	The security services site name.	NULL or DCE site name where the security services may be obtained.	NULL
	Advice: If **NULL** is used, the local security site is used.		
Security Registry Scope	The security registry search scope for HPSS principal names.	NULL or DCE security registry directory.	NULL
	Advice: If **NULL** is used, the entire security registry is searched for the principal name. If your site has other DCE applications separate from HPSS, you may want to limit the search scope to an HPSS security registry directory.		
Keytab Pathname	The absolute pathname of the UNIX file containing the keytab entry that will be used by the server when setting up its identity.	Any legal UNIX file name can be used as long as it is the name of a keytable file.	/krb5/ hpss.keytabs
	Advice: The server must have read access to this file. Do not set other access permissions on this file or your security can be breached. *Notes:* (1) Each server can have its own key file, or all the servers can share a single key file. It is recommended that one key file be used for all of the servers on any given platform.(2) To use the standard DCE system wide key file, set this value to **/krb/v5srvtab** (not recommended).		

Table 5-1 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Authentication Service Arg	The argument passed to the authentication service indicated by the Authentication Service configuration variable and used by the authentication service to validate communications. Currently, the only authentication services supported are **none** and **dce**.	NULL or any UNIX pathname that points to a key file.	/krb5/ hpss.keytabs
	Advice: If **dce** authentication is to be used and the **Keytab Pathname** is **/krb/ v5srvtab**, set this variable to **NULL**. If **dce** authentication is to be used and the **Keytab Pathname** is not **/krb/v5srvtab**, set this variable to the value of **Keytab Pathname**. In either case, the server must have read access to the file. Do not set other permissions on this file or your security can be breached. If no authentication is to be used, set this value to **NULL**.		
Audit Policy. The following 20 fields describe the server's audit policy configuration.			
AUTH	The Security Audit Policy for Authentication events. If set, security audit messages will be sent to the logging subsystem.	NONE- No audit messages will be generated. FAILURE - Audit messages will only be generated when there are errors. ALL - Audit messages will be generated for all related operations.	FAILURE
	Advice: Sites that must audit all login type events should set this value to **ALL**.		
AOPEN	The Security Audit Policy for Audit File Open events. If set, security audit messages will be sent to the logging subsystem. Not currently used.	NONE, FAILURE, ALL	NONE
CHMOD	The Security Audit Policy for Name Server Object Permissions events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Name Server; NONE for other servers
CHOWN	The Security Audit Policy for Name Server Object Owner events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Name Server; NONE for other servers

Table 5-1 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
CREAT	The Security Audit Policy for Name Server Bitfile Creation events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Name Server; NONE for other servers
	Advice: Sites that must audit object creation should set the CREAT field to **ALL** for Name Server.		
LINK	The Security Audit Policy for Name Server Hard Link Creation events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Name Server; NONE for other servers
	Advice: Sites that must audit object creation should set the LINK field to **ALL** for Name Server.		
MKDIR	The Security Audit Policy for Name Server Directory Creation events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	NONE
	Advice: Sites that must audit object creation should set the MKDIR field to **ALL** for Name Server.		
OPEN	The Security Audit Policy for Bitfile Server Bitfile Open events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Bitfile Server; NONE for other servers
OPENDIR	The Security Audit Policy for Name Server Directory Open events. If set, security audit messages will be sent to the logging subsystem. Not currently used.	NONE, FAILURE, ALL	NONE
RENAME	The Security Audit Policy for Name Server Object Rename events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Name Server; NONE for other servers
	Advice: Sites that must audit object deletion should set the RENAME field to **ALL** for Name Server.		

Table 5-1 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
RMDIR	The Security Audit Policy for Name Server Directory Remove events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Name Server; NONE for other servers
	Advice: Sites that must audit object deletion should set the RMDIR field to **ALL** for Name Server.		
UNLINK	The Security Audit Policy for Name Server Object Delete events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Name Server; NONE for other servers
	Advice: Sites that must audit object creation should set the UNLINK field to **ALL** for Name Server.		
UTIME	The Security Audit Policy for Bitfile Time Modified events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Bitfile Server and Name Server; NONE for other servers
ACL_SET	The Security Audit Policy for Name Server Access Control List Modification events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Name Server; NONE for other servers
MAC_SET	The Security Audit Policy for Name Server Set Mandatory Access Control Information events. If set, security audit messages will be sent to the logging subsystem. Not currently used.	NONE, FAILURE, ALL	NONE
PRIV_SET	The Security Audit Policy for Name Server Set Privileged Attribute events. If set, security audit messages will be sent to the logging subsystem. Not currently used.	NONE, FAILURE, ALL	NONE

Table 5-1 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
CHDIR	The Security Audit Policy for Name Server Change Directory events. If set, security audit messages will be sent to the logging subsystem. Not currently used.	NONE, FAILURE, ALL	NONE
CHBFID	The Security Audit Policy for Name Server Change Bitfile Identifier events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Name Server; NONE for other servers
BFSETATTRS	The Security Audit Policy for Bitfile Server Set Bitfile Attribute events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Bitfile Server; NONE for other servers
HPSS4	Not currently used.	NONE, FAILURE, ALL	NONE

5.4 HPSS Storage Policy Configuration

The configuration of the HPSS Storage Policy includes creating the Migration Policy, the Purge Policy, the Accounting Policy, the Logging Policy, and the Location Policy. These policies will subsequently be used in configuring the HPSS storage classes and the HPSS servers.

Before configuring the HPSS Storage Policy, the planning guidelines as described in Section 2.7 should be carefully considered.

5.4.1 Configure the Migration Policies

The migration policy defines the criteria under which the HPSS files are moved from one level of storage hierarchy to lower levels in the hierarchy. A migration policy can be created using the Migration Policy window. After the configuration entry is created, it may be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Migration Policies** option.

The Migration Policy window will be displayed as shown in Figure 5-5. The fields are displayed with default values for a new policy. If the default data is not desired, change the field with the desired value. Click on the **Add** button to create the configuration entry.

To update an existing policy, select the **Load Existing** button on the Migration Policy window and select the desired policy from the popup list. The window will be refreshed with the configured data. After modifying the policy, click on the **Update** button to write the changes to the SFS file.

To delete an existing policy, select the **Load Existing** button on the Migration Policy window and select the desired policy from the popup list. The window will be refreshed with the configured data. Click on the **Delete** button to delete the policy.

Before changes made to the Migration Policy configuration take effect, the Migration/Purge Server must be restarted or told to re-read the policy from the MPS Storage Class Information window. However, changes to some fields also require the Bitfile Server to be restarted before they take effect. Consult Table 5-2 to determine changes to which fields require the Bitfile Server to be restarted.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

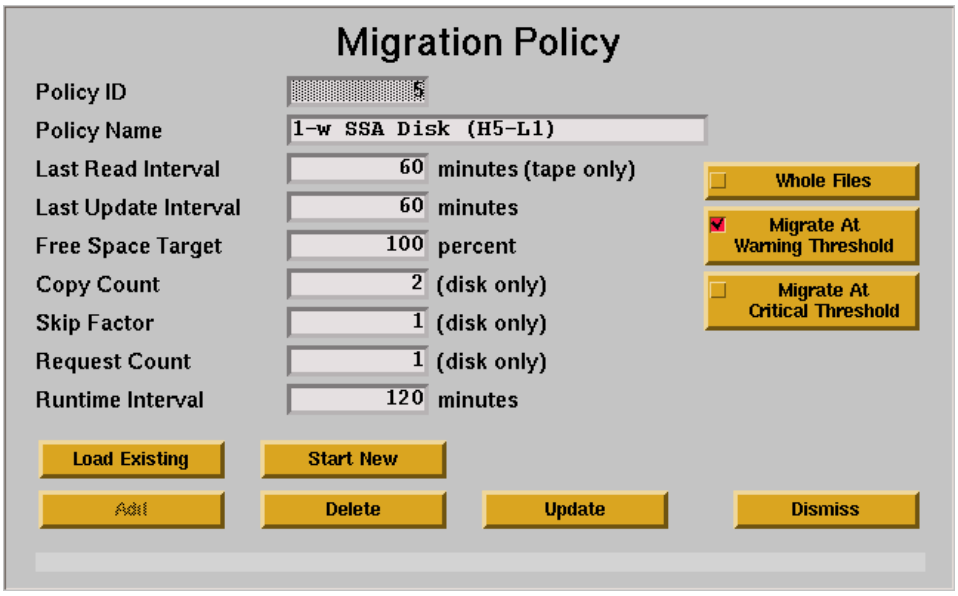


Figure 5-5 Migration Policy Configuration Window

Migration Policy Configuration Variables

Table 5-2 lists the fields on the Migration Policy window and provides specific recommendations for configuring the Migration Policy for use by HPSS.

Table 5-2 Migration Policy Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Policy ID	A unique ID associated with the Migration Policy.	Any unique, non-zero, positive integer value.	Last configured Migration Policy ID plus 1.
Policy Name	The descriptive name of a Migration Policy.	Any character string up to 31 bytes in length.	Migration Policy ID
	Advice: A policy's descriptive name should be meaningful to local site administrators and operators.		
Last Read Interval	An integer that determines whether a file is a candidate for migration. The file must not have been read in the last <interval> minutes. This value is only applicable for files residing on tapes.	Any positive 32-bit integer value.	60
	Advice: Refer to Section 2.7.1.2 for more information on tape migration.		
Last Update Interval	An integer that determines whether a file is a candidate for migration. The file must not have been written to in the last <interval> minutes to be a candidate.	Any positive 32-bit integer value.	60
Free Space Target	The desired percentage of purgeable disk or free tape space to have when migration is completed. The migration will be terminated when this goal is reached.	Any integer value between 1 and 100.	100
Copy Count	The number of copies of a file to create during migration from this level of the hierarchy. This value is only applicable for files residing on disk.	Any integer number between 0 and 4. A value of 0 is treated as 1.	0
	Note: After changing the value for this field, the Bitfile Server must be restarted for the change in policy to take effect. If the Bitfile Server is not restarted, change Class of Service requests for files will be performed based on the prior settings.		

Table 5-2 Migration Policy Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Skip Factor	An integer that indicates how many hierarchy levels to skip over when writing a second file copy. This value is only applicable for files residing on disk.	Any integer number between 0 and 4. A value of 0 is treated as 1.	0
	Note: After changing the value for this field, the Bitfile Server must be restarted for the change in policy to take effect. If the Bitfile Server is not restarted, change Class of Service requests for files will be performed based on the prior settings.		
Request Count	The number of parallel migration threads which will be run during migration. This value is only applicable for files residing on disk.	Any positive 32-bit integer value greater than 0.	1
Runtime Interval	An integer, in minutes, that dictates how often the migration process will occur within a storage class.	Any positive 32-bit integer value.	120 minutes
	Note: The value specifies the interval between the completion of one migration run and the beginning of the next.		
Whole Files	A flag that indicates whether an entire file or only file segments will be migrated. Whole file migration only applies to files residing on tape.	ON, OFF	OFF
Migrate At Warning Threshold	A flag that indicates a migration run should be started immediately when the storage class warning threshold is exceeded.	ON, OFF	OFF
Migrate At Critical Threshold	A flag that indicates a migration run should be started immediately when the storage class critical threshold is exceeded.	ON, OFF	OFF

5.4.2 Configure the Purge Policies

The purge policy defines the criteria under which files are deleted from a level in the storage hierarchy after they have been migrated to a lower level. A purge policy can be created using the Purge Policy window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Purge Policies** option.

The Purge Policy window will be displayed as shown in Figure 5-6. The fields are displayed with default values for a new policy. If the default data is not desired, change the field with the desired value. Click on the **Add** button to create the configuration entry.

To update an existing policy, select the **Load Existing** button on the Purge Policy window and select the desired policy from the popup list. The window will be refreshed with the configured data. After modifying the policy, click on the **Update** button to write the changes to the SFS file.

To delete an existing policy, select the **Load Existing** button on the Purge Policy window and select the desired policy from the popup list. The window will be refreshed with the configured data. Click on the **Delete** button to delete the policy.

Before changes made to the Purge Policy configuration take effect, the Migration/Purge Server must be restarted or told to re-read the policy from the MPS Storage Class Information window. However, changes to some fields also require the Bitfile Server to be restarted before they take effect. Consult Table 5-3 to determine changes to which fields require the Bitfile Server to be restarted.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

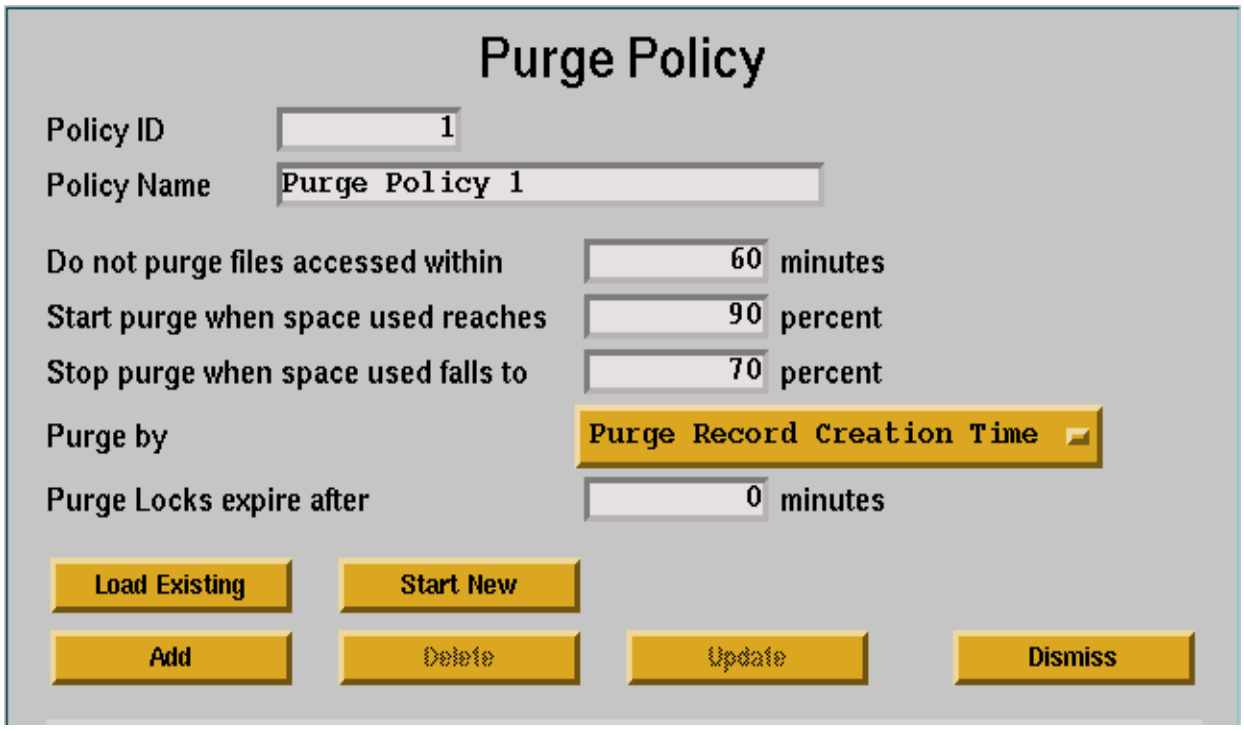


Figure 5-6 Purge Policy Window

Purge Policy Configuration Variables

Table 5-3 lists the fields on the Purge Policy window and provides specific recommendations for configuring the Purge Policy for use by HPSS.

Table 5-3 Purge Policy Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Policy ID	A unique ID associated with the Purge Policy.	Any unique, non-zero, positive integer value.	Last configured Purge Policy ID plus 1.
Policy Name	The descriptive name of a Purge Policy.	Any character string up to 31 bytes in length.	Purge Policy ID
	Advice: A policy's descriptive name should be meaningful to local site administrators and operators.		
Do not purge files accessed within	To be considered a candidate for purging, a file must have been migrated, and it must have remained unaccessed (for read or write) for the length of time specified by this field.	Any positive 32-bit integer value.	60
Start purge when space used reaches	Purging will begin for a storage class when the amount of its space used exceeds this threshold. Used space includes any file in the storage class, whether it has been migrated or not.	Any integer value between 0 and 100.	90
Stop purge when space used falls to	Purging will stop for a storage class when the amount of its space used drops to this threshold. Note that the purge may stop before this point if it runs out of files which are available for purging.	Any integer number between 1 and 100.	70
Purge by	The MPS uses this time attribute of a file to determine which files are eligible to be purged.	Purge Record Creation Time, File Creation Time, or Last Data Access Time	Purge Record Creation Time
	Note: After changing the value for this field, the Bitfile Server must be restarted for the change in policy to take effect.		

Table 5-3 Purge Policy Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Purge Locks expire after	Maximum number of minutes that a file may hold a purge lock. Purge locked files are not eligible for purging.	Any integer value between 0 and 3600. A value of 0 indicates that purge locks expire immediately.	0

5.4.3 Configure the Accounting Policy

The accounting policy defines how each HPSS user account will be charged for using HPSS resources. Refer to Section 2.7.3 for more information on setting and using the Accounting Policy. An accounting policy can be created using the Accounting Policy window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Accounting Policy** option.

The Accounting Policy window will be displayed as shown in Figure 5-7. Since only one Accounting Policy can be configured, the fields are displayed with default values for a new policy if one does not exist. Otherwise the fields will be displayed with data from the configured policy.

To add a new policy, change the default fields with the desired values and click on the **Add** button to create the configuration entry.

To update an existing policy, modify the policy data and click on the **Update** button to write the changes to the SFS file.

To delete the existing policy, **c**lick on the **Delete** button to delete the policy.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

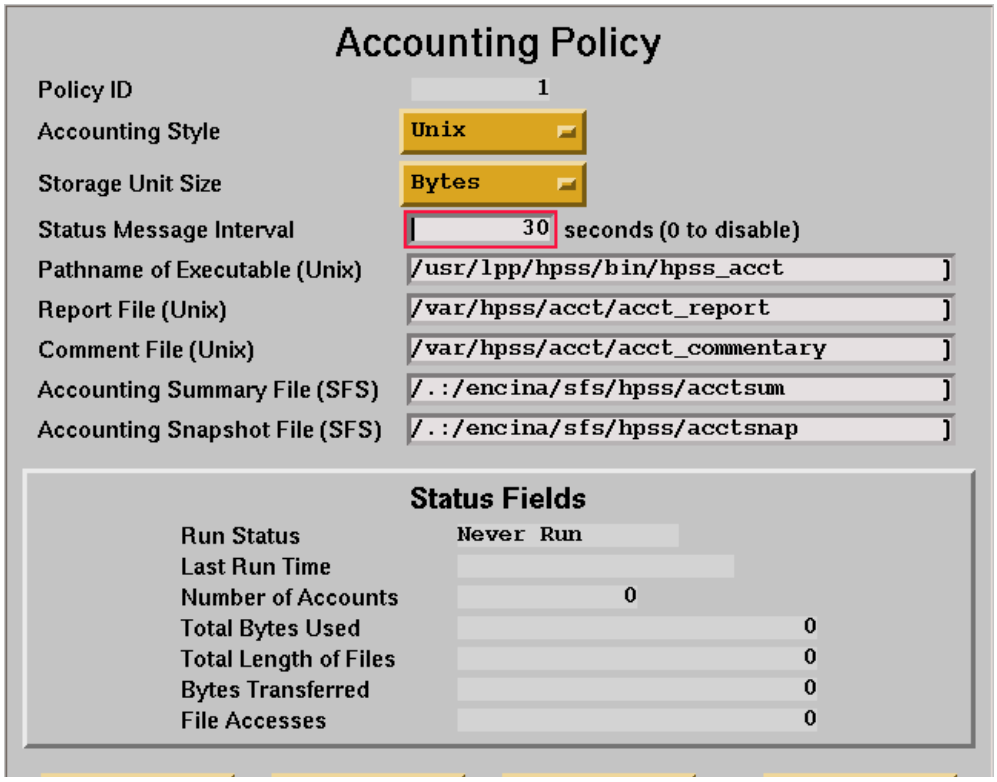


Figure 5-7 Accounting Policy Window

Accounting Policy Configuration Variables

Table 5-4 lists the fields on the Accounting Policy window and provides specific recommendations for configuring the Accounting Policy for use by HPSS.

Table 5-4 Accounting Policy Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Policy ID	The unique ID associated with the Accounting Policy.	Always 1. This field is not changeable.	1
	Advice: This number is always 1 for this version of HPSS since only one accounting policy is currently allowed.		
Accounting Style	Style of accounting that is used by the entire HPSS system.	Site, UNIX	UNIX
	Advice: This field must be set up properly before any files are written to the system. UNIX-style accounting obtains account number from the user's UID. Site-style accounting allows the account to be set specifically. Once the accounting policy is configured, this value cannot be changed. *Note*: In the current release, this field is not being used. The accounting style of a site is determined by the presence of the **AA= <default-acct-id>** string in any user entries in the DCE registry. If Site-style accounting is used, each user entry in the DCE registry must contain a string of the form **AA=<default-acct-id>** in its **GECOS** (Miscellaneous Info) field, where **<default-acct-id>** is the default account index to be used by HPSS when creating files for the user. If UNIX-style accounting is used, it is important that the **AA=<default-acct-id>** string does not appear in any GECOS fields.		
Storage Unit Size	The integral units to be displayed in the report file.	Bytes, Kilobytes, Megabytes, Gigabytes Terabytes	Bytes
Status Message Interval	The number of seconds between status messages sent to SSM by the Accounting utility during an accounting run.	Any positive 32-bit integer value.	300 seconds
	Advice: Set the interval to zero (0) if no accounting status messages are desired. Otherwise, for performance reasons, this value should be set to at least 15 seconds		
Pathname of Executable (Unix)	The UNIX path name of the accounting utility executable.	Valid UNIX path name.	/usr/lpp/hpss/bin/hpss_acct
	Advice: SSM should have execute privileges for the pathname.		

Table 5-4 Accounting Policy Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Report File (Unix)	The UNIX pathname where the generated accounting report will be stored.	Any legal UNIX pathname	/var/hpss/acct/acct_report
	Advice: SSM should have write access for the pathname.		
Comment File (Unix)	A UNIX pathname of an optional commentary text file.	Empty, or any legal UNIX pathname	/var/hpss/acct/acct_commentary
	Advice: This pathname is optional. If it exists, SSM must have read access to it.		
Accounting Summary File (SFS)	The Encina SFS filename of the Accounting Summary metadata.	Any valid Encina SFS filename.	/.:/encina/sfs/hpss/acctsum
Accounting Snapshot File (SFS)	The Encina SFS filename of the Accounting Snapshot metadata.	Any valid Encina SFS filename.	/.:/encina/sfs/hpss/acctsnap
Status Fields. The fields below are statuses reported for the current or last accounting run. They are not configuration fields, and thus are not changeable.			
Run Status	Status of the current or last accounting run	Never Run, Running, Failed, Completed	Never Run
	Advice: This value is set by the Accounting Utility when the status of an accounting run changes.		
Last Run Time	The starting timestamp of the current accounting run or the completion time of the last accounting run.	A date and time value.	0
	Advice: This time is set when an accounting run begins, and it is set again when the accounting run terminates.		
Number of Accounts	Total number of accounts in the system.	A positive 32-bit integer value.	0
Total Bytes Used	Total number of bytes accounted for in the system.	A positive 64-bit integer value.	0
Total Length of Files	The total length of bitfiles, specified by the defined **Storage Unit Size**, in the HPSS system.	A positive 64-bit integer value.	0
Bytes Transferred	The total number of bytes transferred in and out of the HPSS system since the last accounting run.	A positive 64-bit integer value.	0

Table 5-4 Accounting Policy Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
File Accesses	The total number of bitfiles accesses since the last accounting run.	A positive 64-bit integer value.	0

5.4.4 Configure the Logging Policies

For any HPSS server or utility which uses the HPSS logging facility, a Logging Policy can be configured to specify which log message types get sent to the HPSS logs, and which message types get sent to SSM for display. All that is required to configure a Logging Policy is a server name. For configured servers, this is the same as the Server Name field on the Basic Server Configuration window (see Section 5.3). For servers and utilities which do not have configurations (such as the SSM Data Server), Logging Policies can be configured using the name by which the server identifies itself to the logging facility. These names can be obtained by examining server startup scripts or HPSS log entries.

If no Logging Policy is configured for a server, the server will log all message types with the exception of Trace messages.

The following log message types can be specified for logging and / or SSM display:

- Alarm—defines a high-level error condition of interest to the administrator. The default policy is to send alarms to both the log and to SSM for displaying in the HPSS Alarms and Events window (Chapter 6, Figure 6-47).

- Event—defines an informational message (e.g., subsystem initializing, subsystem terminating). The default policy is to send events to both the log and to SSM for displaying in the HPSS Alarms and Events window (Figure 6-47).

The following log message types can be specified for logging:

- Debug—defines a lower-level error message. For troubleshooting, these messages are important record types because they provide more detailed information about the root cause of an error. The default policy is to send these messages to the log.

- Request—used to log the fact that a particular client or server request is being processed. The default policy is to send these messages to the log.

- Security—used to log security related events (e.g., authorization failures). The default policy is to send these messages to the log.

- Accounting—used to log accounting information. This record type is not currently used.

- Trace—used to trace any type of entry / exit processing flows. The default policy is *not* to log these record types. Log policy must be set to enable this record type for logging.

The Logging Policy can be created using the Logging Policy window. After the Logging Policy is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Logging Policies** option.

The Logging Policy window will be displayed as shown in Figure 5-8. For configured server, click on the **Name of Server to Which Policy Applies** popup button and select the desired server. The Logging Policy for a configured server may also be accessed from the HPSS Servers window, and from the Basic Server Configuration window for that server. With either of the last two methods, the Logging Policy window opens with the server name already filled in.

For non-configured servers or utilities, simply type a server name into the **Name of Server to Which Policy Applies** field, and press **Enter**.

A Logging Policy can be created for any unique name which is typed into the name field on the Logging Policy window, regardless of whether or not the name refers to any actual server or utility. There is no way, from SSM, to list all of the existing Logging Policies for non-configured servers. Therefore, care should be taken when creating such policies. If too many are created in a haphazard way, some might be forgotten about, resulting in "lost" policy records in the HPSS metadata.

If the Logging Policy for the selected server does not exist, the fields will be displayed with default values for a new policy. Otherwise, the configured policy will be displayed.

To add a new policy, change the default fields as desired and click on the **Add** button to create the policy.

To update an existing policy, modify the desired fields and click on the **Update** button to write the changes to the SFS file.

To delete an existing policy, click on the **Delete** button to delete the policy.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

Once a Logging Policy is created or updated, it will not be in effect until its associated Log Client is started or reinitialized. The Log Client associated with a particular policy is the one which executes on the same host as the server to which the policy applies. The Reinitialize button on the HPSS Servers window(Chapter 6, Figure 6-2) can be used to reinitialize a running Log Client.

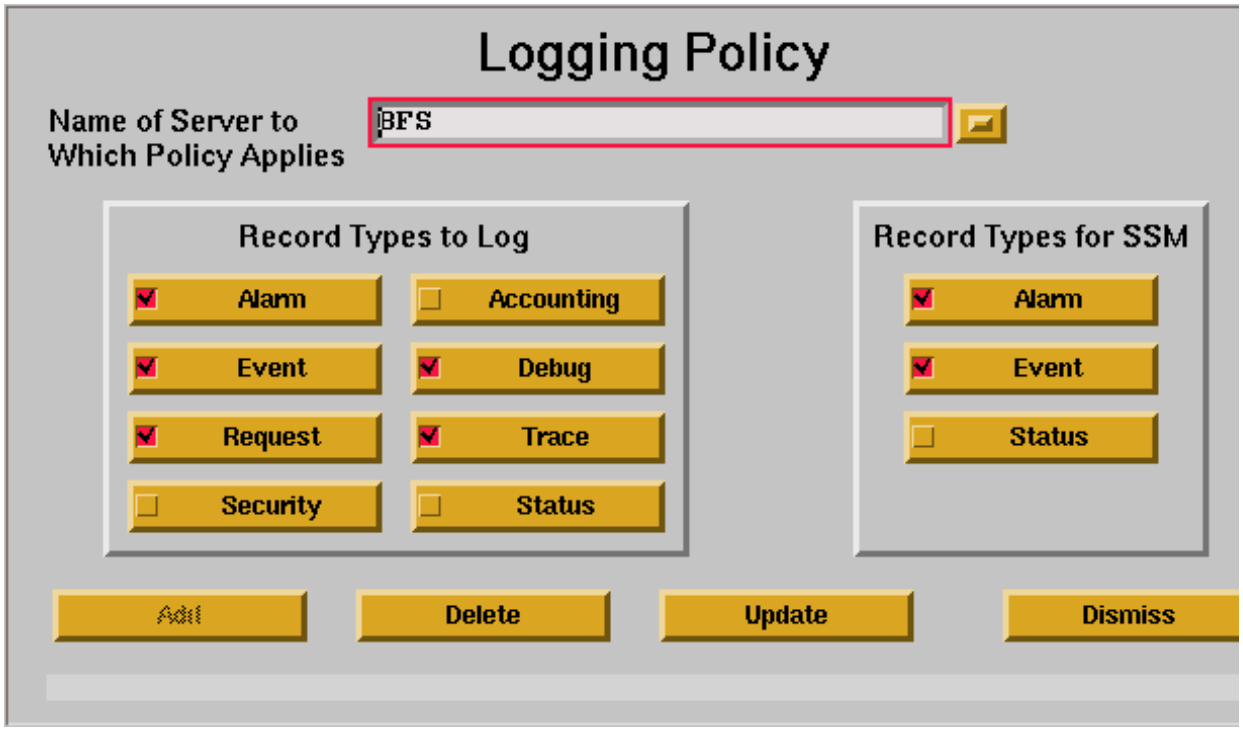


Figure 5-8 Logging Policy Window

Logging Policy Configuration Variables

Table 5-5 lists the fields on the Logging Policy window and provides Logging Policy configuration information.

Table 5-5 Logging Policy Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Name of Server to Which Policy Applies	The descriptive name of the HPSS server to which the Logging Policy will apply.	This field is filled in with the descriptive name of the server that the user had selected.	None if opened from the Health and Status menu; otherwise it is the name of the selected server.

Table 5-5 Logging Policy Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Record Types to Log	Record types that are to be logged for the specified server.	Any combination of the following: Alarm, Event, Request, Security, Accounting, Debug, Trace, Status.	Alarm, Event, Request, Security, Accounting, Debug, Status
	Advice: It is recommended that at least the Alarm, Event, Security and Status record types be selected for all servers while they are running normally. Additional record types for debugging purposes should be selected for servers experiencing problems. If system performance is being degraded by excessive logging, first filter out Trace, Debug and Request record types from all servers. Other record types can be filtered out as necessary. However, the HPSS administrator may lose useful debugging information when problems occurred.		
Record Types for SSM	Record types that are to be sent to SSM for display.	Any combination of the following: Alarm, Event, Status.	Alarm, Event, Status
	Advice: If SSM or system performance is degraded because excessive messages are being sent to SSM by a particular server(s), set the Logging Policy to filter out Alarm and Event record types. However, the HPSS administrator will lose visibility into any subsequent Alarm or Event messages from that server(s).		

5.4.5 Configure the Location Policy

All of the Location Servers at the local HPSS site share the same Location Policy. The Location Policy is used by the Location Servers to determine how and how often information should be updated. In general, most of the default values for the policy can be used as defined.

The Location Policy can be created using the Location Policy window. After the Location Policy is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Location Policy** option. The Location Policy window will be displayed as shown in Figure 5-9. If the Location Policy does not exist, the fields will be displayed with default values for a new policy. Otherwise, the configured policy will be displayed.

To add a new policy, change the default fields as desired and click on the **Add** button to create the policy.

To update an existing policy, modify the desired fields and click on the **Update** button to write the changes to the SFS file.

To delete an existing policy, click on the **Delete** button to delete the policy.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

Once a Location Policy is created or updated, it will not be in effect until all local Location Servers are started or reinitialized. The Reinitialize button on the HPSS Servers window(Chapter 6, Figure 6-2) can be used to reinitialize a running Location Server.

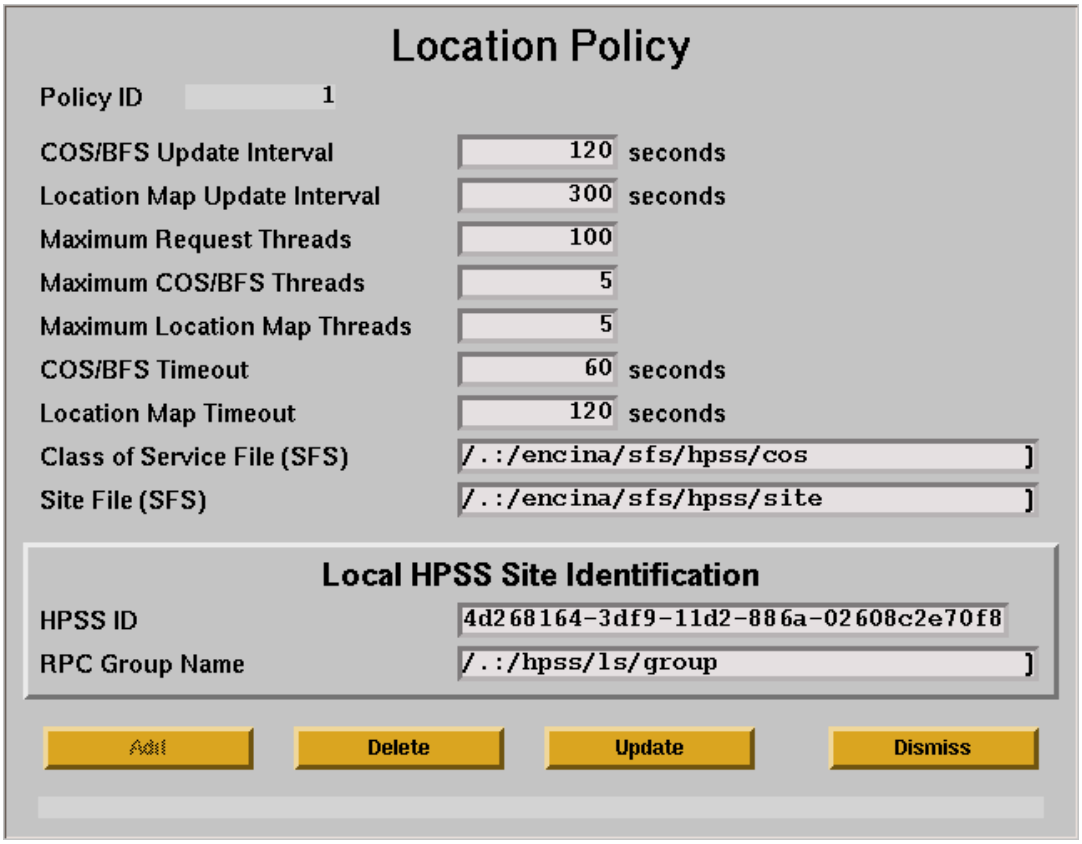


Figure 5-9 Location Policy Window

Location Policy Configuration Variables

Table 5-6 lists the fields on the Location Policy window and provides Location Policy configuration information.

Table 5-6 Location Policy Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Policy ID	The unique ID associated with a Location Policy.	This field is always one (1) and may not be changed.	1

Table 5-6 Location Policy Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
COS/BFS Update Interval	Interval in seconds that the LS rebuilds its COS storage statistics by contacting each local BFS for information.	Any positive integer value.	120
	Advice: If the value is set too low, this will cause an unnecessary load on each BFS. A value below 30 is not recommended. If set too high, COS selection will be based on old information. This value should be adjusted upward from the default if many local LS are defined or if many BFS are defined.		
Location Map Update Interval	Interval in seconds that the LS rereads general server configuration metadata.	Any positive integer value.	300
	Advice: If this value is set too low, a load will be put on SFS while reading configuration metadata and the LS will be unable to contact all remote LSs within the timeout period. If set too high, new servers will not be registered in a timely manner. Set this value higher if timeouts are occurring during remote LS communications.		
Maximum Request Threads	The maximum number of concurrent client requests allowed.	Any positive integer greater than 2 and less than or equal to 400	100
	Advice: If the LS is reporting heavy loads, increase this number. If this number is above 300, consider replicating the LS on a different machine. Note if this value is changed, the general configuration thread value should be adjusted as well.		
Maximum COS/BFS Threads	The maximum number of threads allocated to contact local BFSs concurrently.	Any positive integer value.	5
	Advice: This value does not need to be changed unless there are many local BFS defined and the system is experiencing timeout problems contacting the BFS.		
Maximum Location Map Threads	The maximum number of threads allocated to contact remote LSs concurrently.	Any positive integer value.	5
	Advice: This value does not need to be changed unless there are defined remote sites and the system is experiencing timeout problems contacting the remote LS.		
COS/BFS Timeout	The maximum amount of time to wait for a single BFS to respond when returning COS storage statistic information.	Any positive integer value.	60
	Advice: This value does not need to be changed unless the system is experiencing very long delays in contacting the BFS.		

Table 5-6 Location Policy Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Location Map Timeout	Maximum amount of time in seconds to wait for a single remote LS to return location map information.	Any positive integer value.	120
	Advice: This value should only be changed if the system is experiencing very long delays while contacting remote Location Servers.		
Class of Service File (SFS)	The Encina SFS filename of the Class of Service metadata.	Any valid Encina SFS filename.	/.:/encina/sfs/hpss/cos
Site File (SFS)	The Encina SFS filename of the Remote HPSS Site metadata.	Any valid Encina SFS filename.	/.:/encina/sfs/hpss/site
HPSS ID	The unique identifier (UUID) for this HPSS installation.	Any valid UUID value.	A random UUID is generated.
	Advice: If you change this value, make sure it is unique. This value is used by remote HPSS sites to connect to your HPSS site.		
RPC Group Name	The CDS pathname where the DCE RPC group containing local LS path information should be stored.	Any valid CDS pathname.	/.:/hpss/ls/group
	Advice: All clients will need to know this group name since it is used by them when initializing to contact the Location Server. If the default is not used, ensure that associated environment variable for this field is changed accordingly for all HPSS interfaces.		

5.5 HPSS Storage Characteristics Configuration

Before an HPSS system can be used, storage classes, storage hierarchies, and classes of service must be created. Before configuring them, the planning guidelines described in Section 2.8 should be carefully considered.

5.5.1 Configure the Storage Classes

Storage class information must be created for each storage class that is to be supported by the HPSS system. A storage class can be created using the HPSS Storage Class window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Storage Classes** option.

The Storage Class Configuration window will be displayed as shown in Figure 5-10. The fields are displayed with default values for a new disk storage class. If a tape storage class is desired, click on the **Tape** button to display the default tape storage class (Figure 5-11) before modifying any other fields. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing storage class, select the **Load Existing** button on the Storage Class Configuration window and select the desired storage class from the popup list. The window will be refreshed with the data from the selected storage class. After modifying the data, click on the **Update** button to write the changes to the SFS file. Refer to Section 6.4.10.1 for more guidelines on changing a storage class configuration.

To delete an existing storage class, select the **Load Existing** button on the Storage Class Configuration window and select the desired storage class from the popup list. The window will be refreshed with the configured data. Click on the **Delete** button to delete the storage class. Refer to Section 6.4.11.1 for more guidelines on deleting a storage class configuration.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

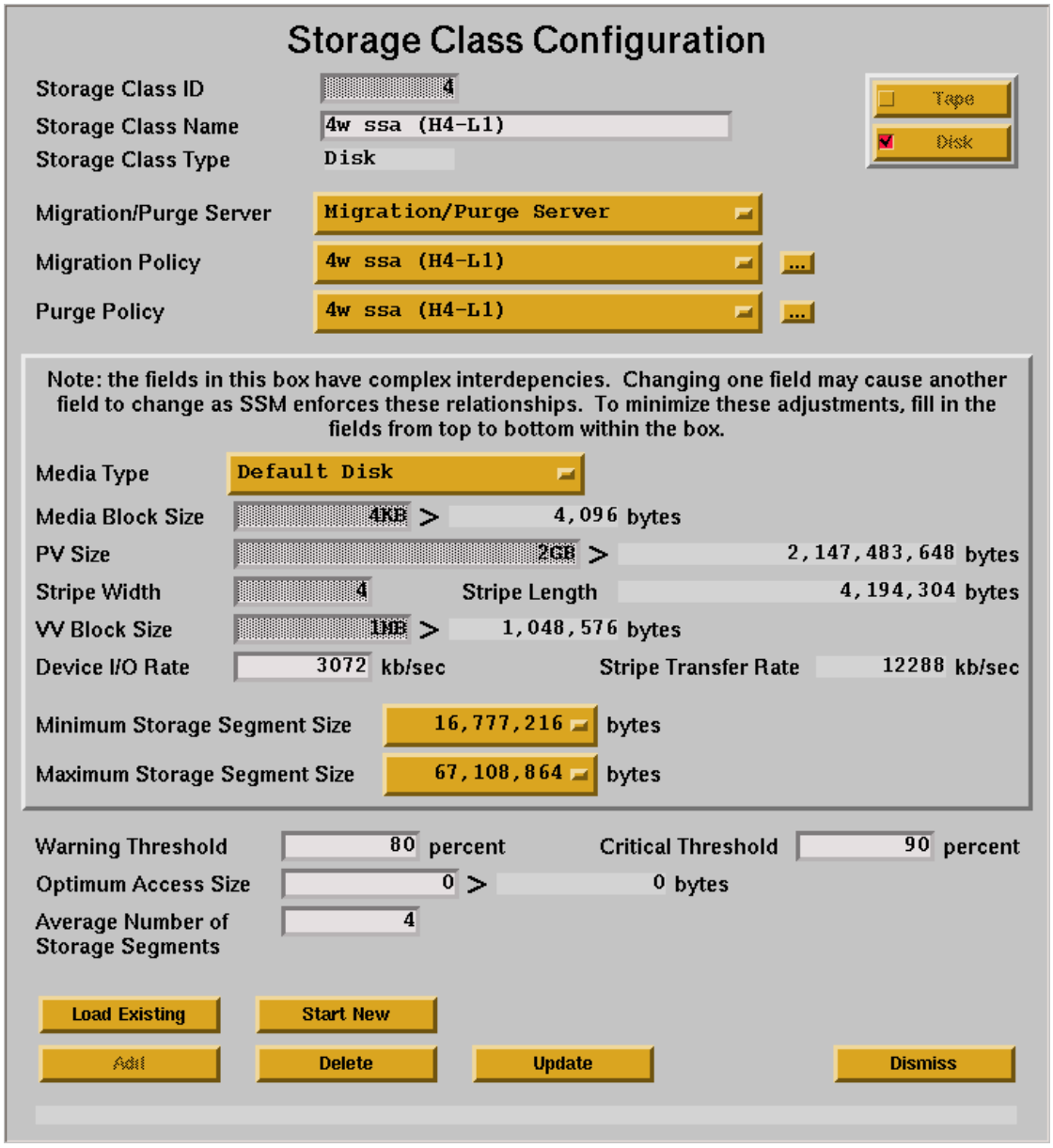


Figure 5-10 Disk Storage Class Configuration Window

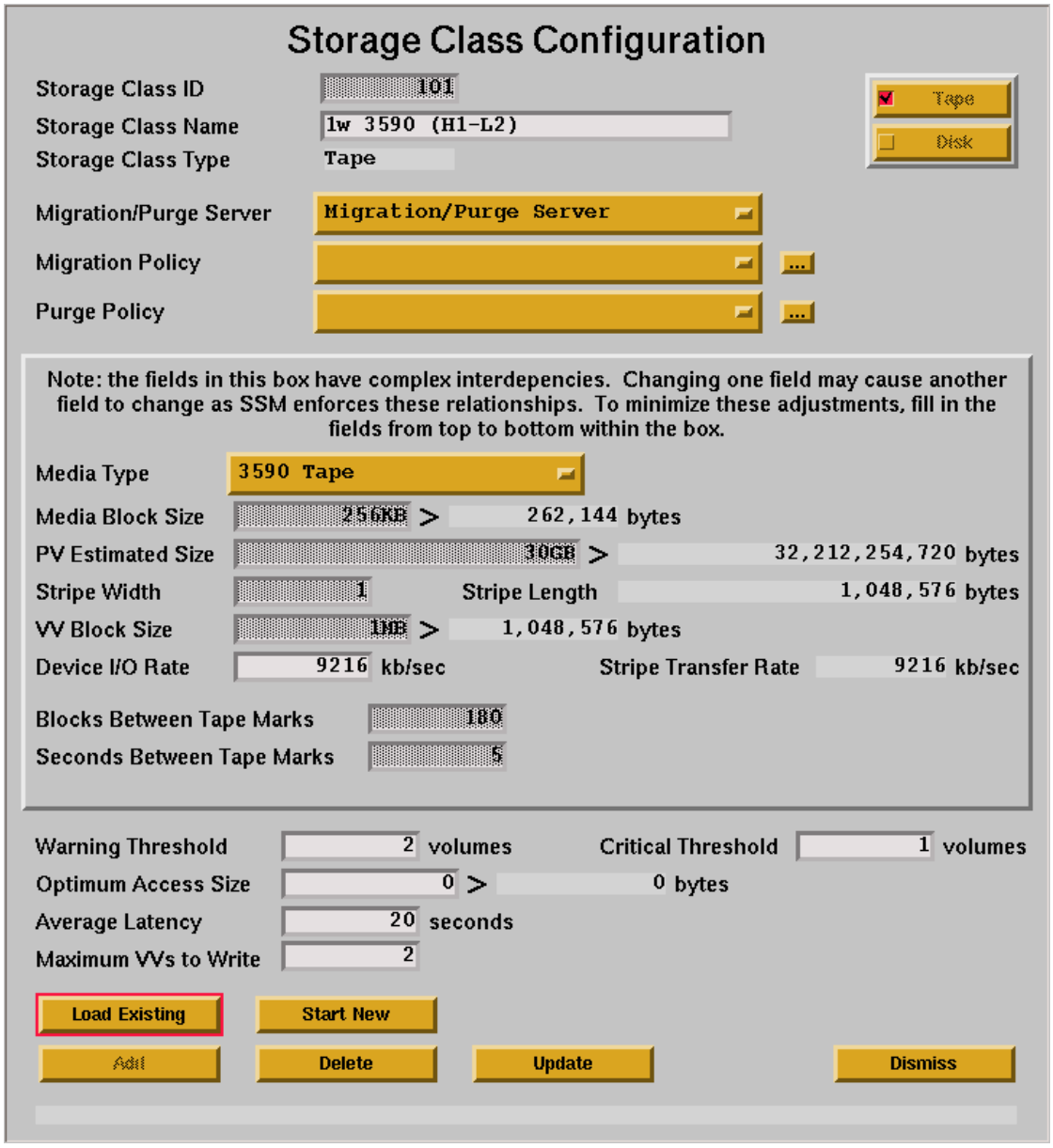


Figure 5-11 Tape Storage Class Configuration Window

Storage Class Configuration Variables

Table 5-7 lists the fields on the Storage Class Configuration window and provides specific recommendations for configuring the storage class for use by HPSS.

Table 5-7 Storage Class Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Storage Class ID	A unique numeric ID associated with this storage class.	Any non-zero, positive 32-bit integer value.	Last configured ID plus 1.
Storage Class Name	A text string used to describe this storage class.	Any character string up to 31 bytes in length.	Storage Class ID
	Advice: Choose a Storage Class Name that describes the function of the storage class. Good examples are 4-Way striped 3490 or Non-Striped SCSI Disk.		
Storage Class Type	An indicator of whether the media to be associated with this storage class are tapes or disks.	This field is set with the Tape / Disk toggle buttons and cannot otherwise be modified.	Tape or Disk based on the toggle option selected.
Migration/Purge Server	The descriptive name of the MPS, if data is to be migrated or purged from this storage class	Any configured MPS from the pop-up list. Must have a valid MPS name even though migration and / or purge is not desired for this storage class. If field is blank, this storage class cannot be monitored through SSM.	Blank.
Migration Policy	The name of the migration policy associated with this storage class, if data is to be migrated from this storage class.	Any configured migration policy name from the pop-up list. May be left blank if migration is not desired for this storage class.	Blank.
	Advice: Do not configure a migration policy for a storage class at the lowest level in a hierarchy.		
Purge Policy	The name of the purge policy associated with this storage class, if data is to be purged from this storage class.	Any configured purge policy name from the pop-up list. May be left blank if purge is not desired for this storage class.	Blank.
	Advice: Do not configure a purge policy for a tape storage class or a storage class which does not support migration.		
Media Type	The type of media comprising this storage class.	Any valid media type from the pop-up list.	Default Tape or Default Disk based on Storage Class Type.

Table 5-7 Storage Class Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Media Block Size	The size of the data block on the physical volume in bytes.	Any positive 32-bit integer value.	Based on selected Media Type.
	Advice: The block size should be set to a value appropriate for the volume type. See Section 2.8.1.1 for more details.		
PV Estimated Size/ PV Size (for Disk)	The estimated amount of data that can be written on the physical volume.	Any positive 64-bit integer value. Must be a multiple of the VV Block Size.	Based on selected Media Type
	Advice: See Section 2.8.1.10 for considerations on selecting a good value for PV Estimated Size. For disk, **PV Size** must be less than or equal to the **Bytes on Device** value from Section 5.7.		
Stripe Width	The number of physical volumes in a virtual volume.	Any value from 1 to 256.	1.
Stripe Length	The number of bytes in a stripe.	This value is calculated by SSM. It is the product of the specified VV Block Size and Stripe Width fields.	Same as VV Block Size.
VV Block Size	The size of the logical data block on the new virtual volume.	A 32-bit integer value.	First multiple of Media Block Size which equals or exceeds 1MB.
	Advice: The VV Block Size chosen will determine the performance characteristics of the virtual volume. This value must meet the following restraining requirements: (1) it must be an integer multiple of the Media Block Size. (2) For tape, it must be an even divisor of the virtual volume section size (Media Block Size * Blocks Between Tape Marks). For example, if the Media Block Size is 64 K, and the Blocks Between Tape Marks is 512, the virtual volume section size is 32 MB. The VV Block Size could be 64 K, 128 K, 256 K, or larger, but not 192 K. (3) For disk, the VV Block Size has to be large enough that there are no more than 16K VV blocks in a VV. In other words, PV Size * Stripe Width / VV Block Size cannot exceed 16,384. See Sections 2.8.1.2 and 2.8.1.3 for more details on selecting a good value for VV Block Size.		
Device I/O Rate	The approximate data transfer speed, in kilobytes per second, which can be achieved by devices corresponding to "Media Type". For Tape, this field is used in calculating a reasonable setting for "Blocks Between Tape Marks" (see below). For Disk, this field is mostly informational.	Any positive 32-bit integer value The value should, however, be as close as possible to the actual I/O speed of the device.	Based on selected Media Type.

Table 5-7 Storage Class Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Stripe Transfer Rate	The approximate data transfer rate for the entire stripe.	This value is calculated by SSM. It is the product of the Device I/O Rate and Stripe Width fields.	Same as Device I/O Rate.
Blocks Between Tape Marks	The maximum number of data blocks that can be written on a tape between adjacent tape marks. This field is applicable to tape storage class only.	Any positive 32-bit integer value.	Based on selected Media Type.
	Advice: The number of blocks between the physical volume tape marks should be chosen to use the media efficiently while allowing for quick positioning to a tape mark. See Section 2.8.1.5 for more details.		
Seconds Between Tape Marks	The number of seconds between tape mark writes during a tape write operation derived from Blocks Between Tape Marks and Device I/O Rate. This field is applicable to tape storage class only.	Any positive 32-bit integer value.	Based on selected Media Type.
	Advice: Enter a value into the Blocks Between Tape Marks field, and use Seconds Between Tape Marks value for feedback on the resulting time interval; or enter a value into the Seconds Between Tape Marks and let SSM calculate the Blocks Between Tape Marks value. Changing either field automatically modifies the other one. As a rule of thumb, it is best to choose "Blocks Between Tape Marks" such that 5 to 30 seconds elapsed.		
Storage Segment Size	The smallest allocatable unit of disk space. This field is used for disk storage class only.	Click on the option menu button to pop up a list of acceptable sizes.	Same as VV Block Size.
	Advice: Care should be taken in selecting the Storage Segment Size. Consult Section 2.8.1.6 for considerations related to this selection.		
Maximum Storage Segment Size	When this field is larger than the Storage Segment Size, the Bitfile Server is allowed to use multiple storage segment sizes for disks in this storage class, up to this maximum.	Click on the option menu button to pop up a list of acceptable sizes.	Same as Storage Segment Size.
	Advice: If a value larger than 10 percent of PV Size is specified, the user will be warned of non-optimal use of disk space.		

Table 5-7 Storage Class Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Warning Threshold	Low threshold for the percentage of space used in this storage class. For disk this is the percentage of total space used. For tape this is the number of free VVs remaining. Alarms will be sent to SSM periodically when the threshold is exceeded and the **Space Thresholds** field on the SSM Health and Status window is changed to **Warning**.	For disk, any integer value between 1 and 100. For tape, any positive integer value.	Disk: 80, Tape: 10
Critical Threshold	High threshold for the percentage of space used in this storage class. For disk this is the percentage of total space used. For tape this is the number of free VVs remaining. Alarms will be sent to SSM periodically when the threshold is exceeded and the **Space Thresholds** field on the SSM Health and Status window is changed to **Critical**.	For disk, any integer value between 1 and 100. For tape, any positive integer value.	Disk: 90, Tape: 5
Optimum Access Size	The optimal transmission size to be used for a transfer request using this storage class. Not currently used.	Any positive 32-bit integer value.	0.
Average Latency	The average time (in seconds) that elapses when a data transfer request is scheduled and the time the data transfer begins. This field is only applicable to the tape storage class.	Any positive 32-bit integer value.	Based on selected Media Type.
	Advice: For additional information relating to the selection of storage system characteristics fields, refer to Section 2.8.3.5.		

Table 5-7 Storage Class Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Maximum VVs to Write	The number of tape virtual volumes in the storage class that a Tape Storage Server will use for concurrent writes. This field is only applicable to a the tape storage class.	Any positive 32-bit integer value.	10.
	Advice: Small values in this field restrict files being written in the storage class to a small number of tapes, reducing the number of tape mounts. The number of tape drives used to write files in the storage class will be limited to approximately the value of this field times the stripe width. Read operations are not limited by this value.		
Average Number of Storage Segments	If the Maximum Storage Segment Size field is greater than the Storage Segment Size field, the Bitfile Server attempts to select a storage segment size for a bitfile such that the bitfile can be stored in this number of storage segments.	Any positive 32-bit integer value limited by the maximum number of VVs created for the storage class.	4.

5.5.2 Configure the Storage Hierarchies

Storage hierarchy information must be created for each storage hierarchy that is to be supported by the HPSS system. An HPSS storage hierarchy can be created using the HPSS Storage Hierarchy window. After the configuration entry is created, it can be viewed, updated or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Storage Hierarchies** option.

The Storage Hierarchy Configuration window will be displayed as shown in Figure 5-12. The fields are displayed with default values for a new storage hierarchy. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing storage hierarchy, select the **Load Existing** button on the Storage Hierarchy Configuration window and select the desired storage hierarchy from the popup list. The window will be refreshed with the data from the selected storage hierarchy. After modifying the data, click on the **Update** button to write the changes to the SFS file. Refer to Section 6.4.10.2 for more guidelines on changing a storage hierarchy configuration.

To delete an existing storage hierarchy, select the **Load Existing** button on the Storage Hierarchy Configuration window and select the desired storage hierarchy from the popup list. The window will be refreshed with the configured data. Click on the **Delete** button to delete the storage hierarchy. Refer to Section 6.4.11.2 for more guidelines on deleting a storage hierarchy configuration.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

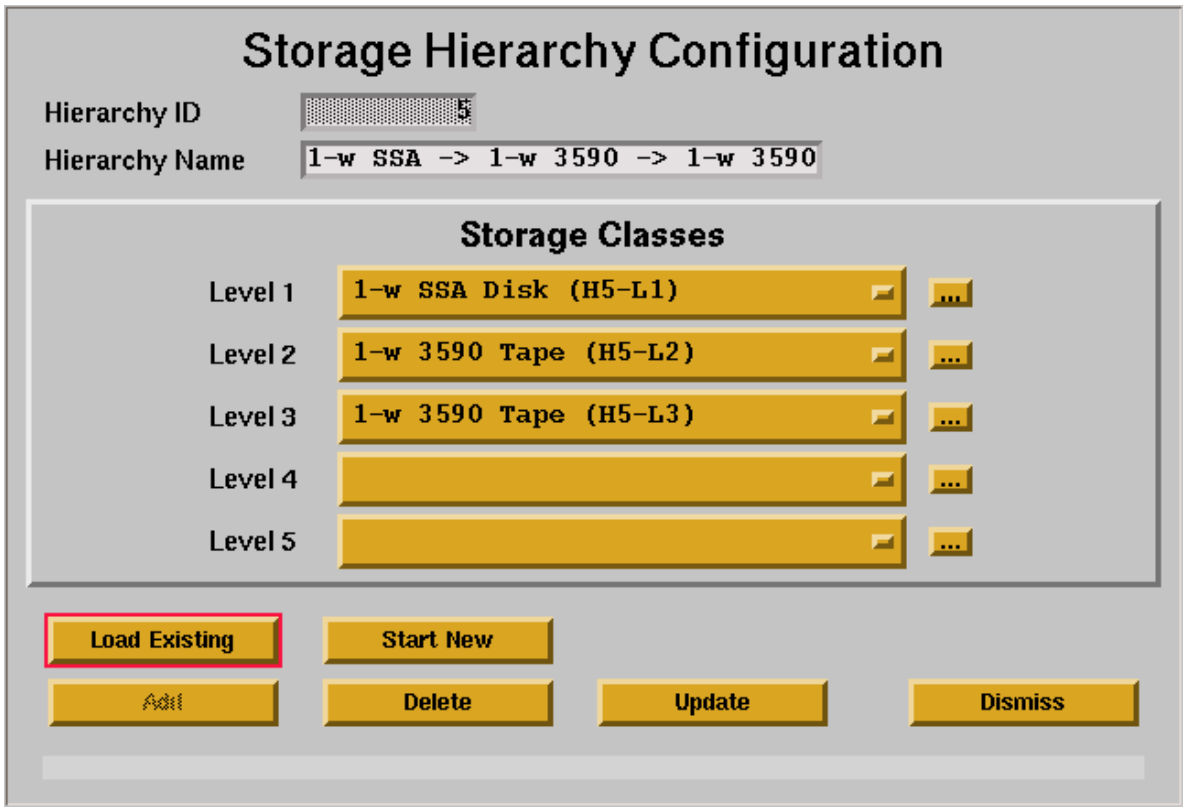


Figure 5-12 Storage Hierarchy Configuration Window

Storage Hierarchy Configuration Variables

Table 5-8 lists the fields on the Storage Hierarchy Configuration window and provides specific recommendations for configuring the Storage Hierarchy for use by HPSS.

Table 5-8 Storage Hierarchy Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Hierarchy ID	The unique, numeric ID associated with this hierarchy.	Any unique, non-zero, positive 32-bit integer value.	Last configured ID plus 1.
Hierarchy Name	The descriptive name associated with this hierarchy.	Any character string up to 31 bytes in length.	Hierarchy ID
	Advice: A hierarchy's descriptive name should be meaningful to local site administrators and operators.		
Level 1	The name of the storage class associated with level 1 in the hierarchy.	Any configured storage class name.	None
Level 2	The name of the storage class associated with level 2 in the hierarchy.	Any configured storage class name.	None
Level 3	The name of the storage class associated with level 3 in the hierarchy.	Any configured storage class name.	None
Level 4	The name of the storage class associated with level 4 in the hierarchy.	Any configured storage class name.	None
Level 5	The name of the storage class associated with level 5 in the hierarchy.	Any configured storage class name.	None
Advice: No two hierarchy levels can reference the same storage class.			

5.5.3 Configure the Classes of Service

Class of Service (COS) information must be created for each class of service that is to be supported by the HPSS system. A COS can be created using the HPSS Class of Service window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Classes of Service** option.

The Class of Service Configuration window will be displayed as shown in Figure 5-13. The fields are displayed with default values for a new class of service. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing class of service, select the **Load Existing** button on the Class of Service Configuration window and select the desired class of service from the popup list. The window will be refreshed with the configured data. After modifying the data, click on the **Update** button to write

the changes to the SFS file. Refer to Section 6.4.10.3 for more guidelines on changing a class of service configuration.

To delete an existing class of service, select the **Load Existing** button on the Class of Service Configuration window and select the desired class of service from the popup list. The window will be refreshed with the configured data. Click on the **Delete** button to delete the class of service. Refer to Section 6.4.11.3 for more guideline on deleting a class of service configuration.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

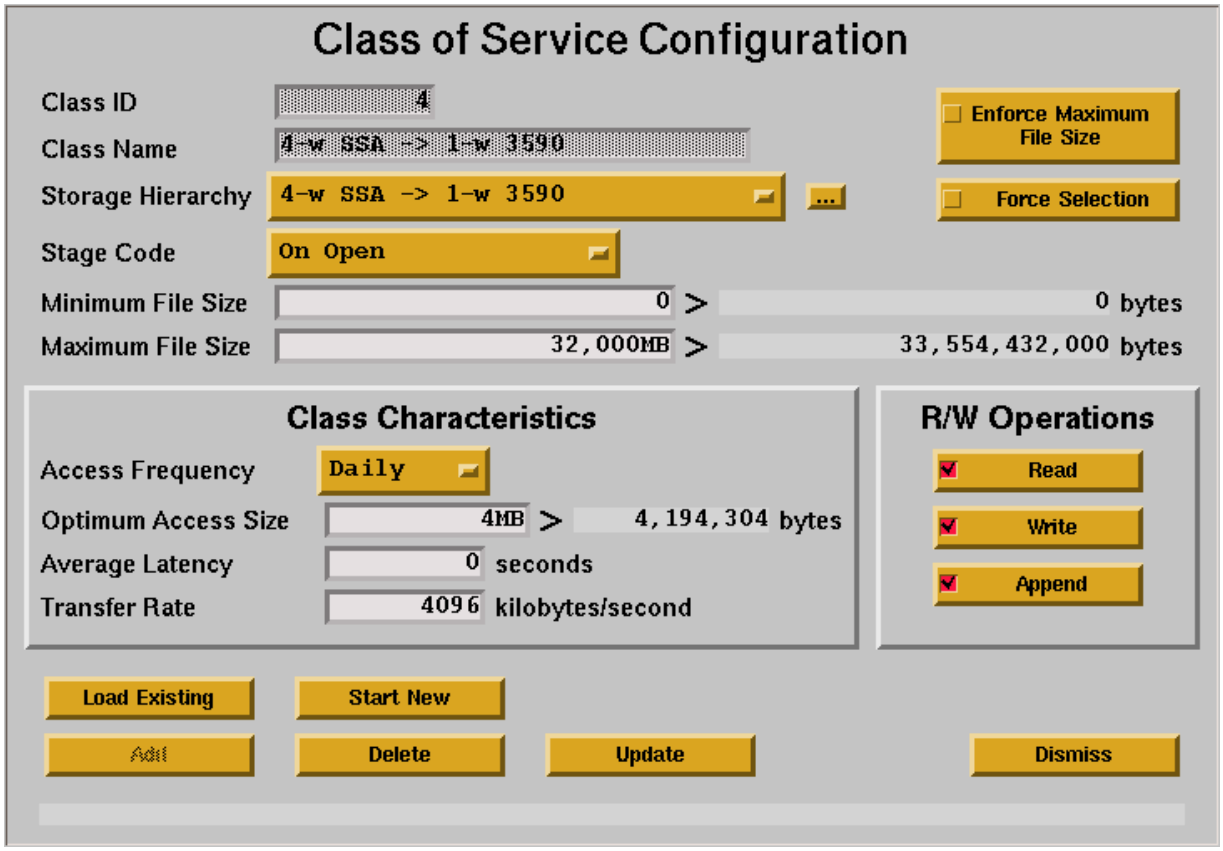


Figure 5-13 Class of Service Configuration Window

Class of Service Configuration Variables

Table 5-9 lists the fields on the HPSS Class of Service window and provides Class of Service configuration information.

Table 5-9 Class of Service Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Class ID	An unique integer ID for the COS.	Any non-zero, positive 32-bit integer value.	Last configured COS ID plus 1.
Class Name	The descriptive name of the COS.	A character string up to 31 bytes in length.	None
	Advice: Select a name that describes the COS in some functional way. A good example would be High Speed Disk Over Tape.		
Storage Hierarchy [P]	The name of the storage hierarchy associated with this COS.	Any configured hierarchy name.	Same as last configured COS.
Stage Code	A code that indicates the file staging options.	On Open, On Open Async, No Stage On Open Background	Same as last configured COS.
	Advice: Refer to Section 2.8.3.3 for more information on selecting the appropriate stage code.		
Minimum File Size	The size, in bytes, of the smallest bitfiles supported by this COS.	Any positive 64-bit integer value.	Same as last configured COS.
Maximum File Size	The size, in bytes, of the largest bitfile supported by this COS.	Any positive 64-bit integer value	Same as last configured COS.
Enforce Maximum File Size	A flag that indicates that a bitfile larger than the Maximum File Size cannot be created in this COS.	ON, OFF	Same as last configured COS.
Force Selection	A flag to determine how a COS will be selected. If ON, a client must explicitly select this COS in order to have a file assigned to it; if the client merely supplies general COS hints for a file, this COS will not be selected	ON, OFF	Same as last configured COS.
R / W Operations	The operations supported for this COS.	READ, WRITE, APPEND	Same as last configured COS.
Class Characteristics. The fields below describe the characteristics of a COS.			

Table 5-9 Class of Service Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Access Frequency	The frequency, on average, for accessing files in this COS.	Hourly, Daily, Weekly, Monthly, Archive	Daily
Optimum Access Size	The suggested number of bytes that should be written at one time for maximum efficiency. Not currently used.	Any positive 32-bit integer value.	Same as last configured COS.
Average Latency	The average time (in seconds) that elapses between the time a transfer request is accepted for processing and the time the data transfer begins.	Any positive 32-bit integer value.	Same as last configured COS.
Transfer Rate	The average throughput (in KB per second) that can be transferred using this COS.	Any positive 32-bit integer value.	Same as last configured COS.

5.5.4 File Family Configuration

File family information must be created for each file family that is to be supported by the HPSS system. A file family can be created using the File Family Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **File Families** option.

The File Family Configuration window will be displayed as shown in Figure 5-14. The fields are displayed with default values for a file family. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing file family, select the **Load Existing** button on the File Family Configuration window and select the desired family from the popup list. The window will be refreshed with the configured data. After modifying the data, click on the **Update** button to write the changes to the SFS file.

To delete an existing file family, select the **Load Existing** button on the File Family Configuration window and select the desired family from the popup list. The window will be refreshed with the configured data. Click on the **Delete** button to delete the configuration.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

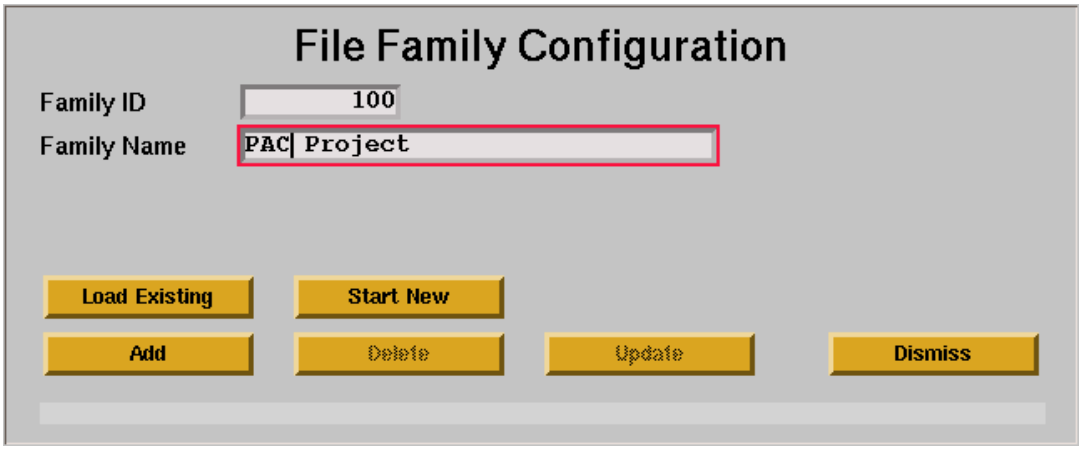


Figure 5-14 File Family Configuration Window

Table 5.10 describes the create junction variables.

Table 5-10 Configure File Family Variables

Display Field Name	Description	Acceptable Values	Default Value
Family ID	An unsigned non-zero integer which serves as a unique identifier for this file family. A unique default value is provided, which may be overwritten if desired. However, if an ID which is already in use by another file family is specified, the Add request will fail. This field may only be modified in Add mode.	Any non-zero, positive 32-bit integer value.	Last configured family ID plus 1
Family Name	A text string which serves as a descriptive identifier for this file family. The name should be unique among all file families. The name should also be informative.	A character string up to 31 characters in length.	File Family "ID "

5.6 Specific Server Configuration

In addition to creating the basic server configuration entries, a specific server configuration entry must be created for each HPSS server of the following types:

- Name Server
- Bitfile Server
- Storage Server
- Migrate / Purge Server
- DMAP Gateway
- Physical Volume Library
- Physical Volume Repository
- Mover
- Log Daemon
- Log Client
- Metadata Monitor
- NFS Daemon
- NFS Mount Daemon
- Non-DCE Client Gateway

Sections 5.6.1 through 5.6.14 describe the specific configuration for each of the above servers.

The SSM servers, the Location Servers and the Startup Daemons do not have specific configurations.

5.6.1 Configure the Name Server Specific Information

The NS specific configuration entry can be created using the Name Server Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 5-3.

To add a new specific configuration, select the Name Server entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The Name Server Configuration window will be displayed as shown in Figure 5-15 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Name Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Name Server Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the Name Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Name Server Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

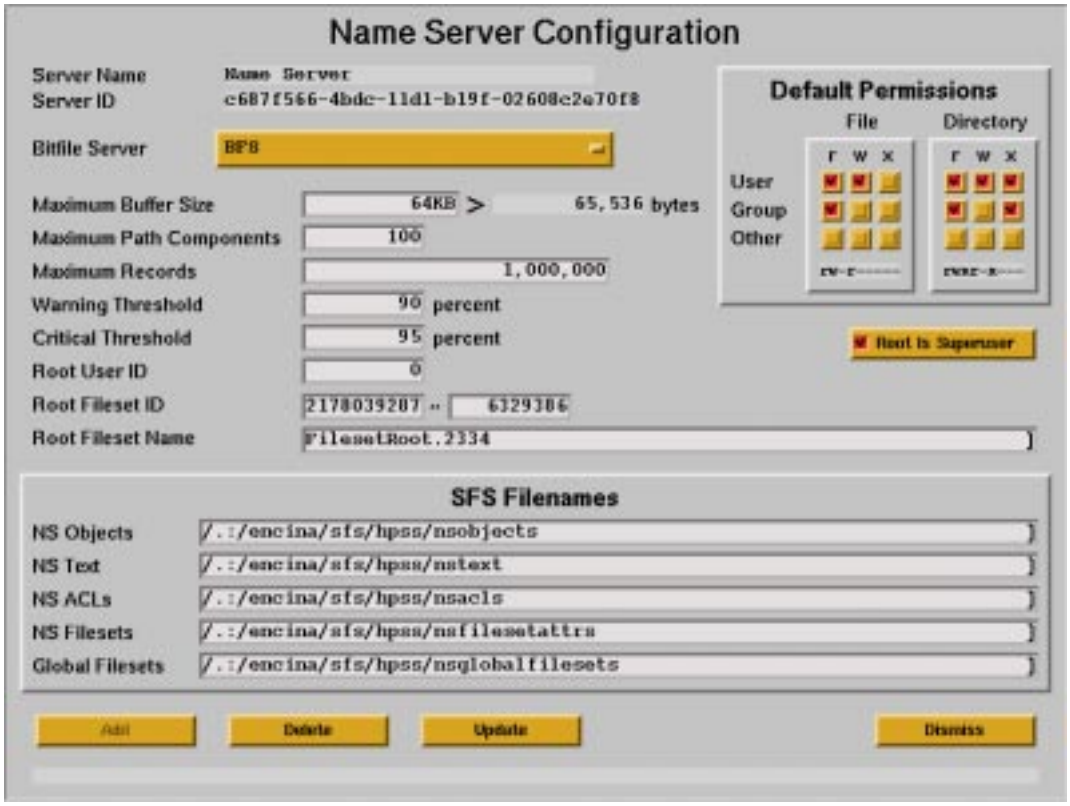


Figure 5-15 Name Server Configuration Window

Name Server Configuration Variables

Table 5-11 lists the fields on the Name Server Configuration window and provides specific recommendations for configuring the NS for use by HPSS.

Table 5-11 Name Server Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the NS. This name is copied over from the NS general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected NS descriptive name
Server ID	The UUID of the NS. This ID is copied over from the NS general configuration entry.	This field cannot be modified. It is displayed for reference only.	Extracted from the NS general configuration entry.
Bitfile Server	The descriptive name of the BFS.	Any configured BFS name from the pop-up list. This field is required.	Name of the first BFS found in the SFS file.
Maximum Buffer Size	The maximum number of bytes that the NS can use when returning data from the Read Directory call.	Any positive integer value between 1 and 64 KB.	32 KB
	Advice: This value must not exceed 64 KB.A buffer size of 64K should be able to hold a list of about 80 entries. See Section 2.6.1 for additional details.		
Maximum Path Components	The maximum number of components permitted in a path name.	Any positive integer value between 1 and 512.	100
Maximum Records	The maximum number of SFS records that can be used by the NS to store metadata objects.	Any positive 64-bit integer value.	1,000,000
	Advice: This limit must consider the maximum anticipated number of files and directories in the system, the upper limit for SFS, and the amount of disk space available to SFS. See Section 2.9.2.2 for additional details.		
Warning Threshold	A percentage value indicating when to issue warning messages. When the percentage of used space exceeds this value, the NS will issue warning messages to SSM.	Any integer value between 1 and 100.	90

Table 5-11 Name Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Critical Threshold	A percentage value indicating when to issue critical messages. When the percentage of used space exceeds this value, the NS will issue critical messages to SSM.	Any integer value between 1 and 100.	95
Root User ID	The UID of the user who has root access privileges to the NS database, if the Root Is Superuser flag is set to ON.	Valid Root User UID	0
Root Fileset ID	The fileset ID to be assigned to the Name Server's local root fileset. A fileset ID consists of two numbers (high and low) separated by a double comma.	The highest part must be an unsigned 32-bit number greater than 2147483648. The low part can be any unsigned 32-bit number.	Generated automatically by SSM
Root Fileset Name	The name to be assigned to the Name Server's local root fileset. The name must be unique among all filesets within the DCE cell.	A character string of up to 127 bytes in length.	Generated automatically by SSM
Root Is Superuser	A flag that indicates whether root privileges are enabled for the UID specified in the User ID field. Root access privileges grant the specified user the same access rights to a name space object as the owner of that object.	ON, OFF	ON
Default Permissions. The following two fields are default permissions that will only be used if no permissions are supplied as attributes when an object is being created.			
File	The default permissions that will be given to files if no permissions are supplied.	Any valid combination of the UNIX file permissions.	rw-r-----
	Advice: The default permissions will apply to the owner, the group associated with the file, and other authenticated DCE users of the DCE cell where the files are created.		

Table 5-11 Name Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Directory	The default permissions that will be given to directories if no permissions are supplied.	Any valid combination of the UNIX directory permissions.	rwxr-x---
	Advice: The default permissions will apply to the owner, the group associated with the directory, and other authenticated DCE users of the DCE cell where the directory is created.		
SFS Filenames. The fields below list the names of the SFS files used by the NS.			
NS Objects	The path name to the SFS file containing the metadata for the NS objects.	Any valid Encina file name.	/.:/encina/sfs/hpss/nsobjects
NS Text	The path name to the SFS file containing the overflow text metadata for the NS objects.	Any valid Encina file name.	/.:/encina/sfs/hpss/nstext
NS ACLs	The path name to the SFS file containing the overflow ACL entries for the NS objects.	Any valid Encina file name.	/.:/encina/sfs/hpss/nsacls
NS Filesets	The path name to the SFS file containing the metadata for the NS filesets.	Any valid Encina file name.	/.:/encina/sfs/hpss/nsfilesetattrs
Global Filesets	The path name to the SFS file containing the metadata for the global fileset information.	Any valid Encina file name.	/.:/encina/sfs/hpss/nsglobalfilesets

5.6.2 Configure the Bitfile Server Specific Information

The Bitfile Server specific configuration entry can be created using the Bitfile Server Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 5-3.

To add a new specific configuration, select the Bitfile Server entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The Bitfile Server Configuration window will be displayed as shown in Figure 5-16 with default values. If the default

data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Bitfile Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Bitfile Server Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the Bitfile Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Bitfile Server Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

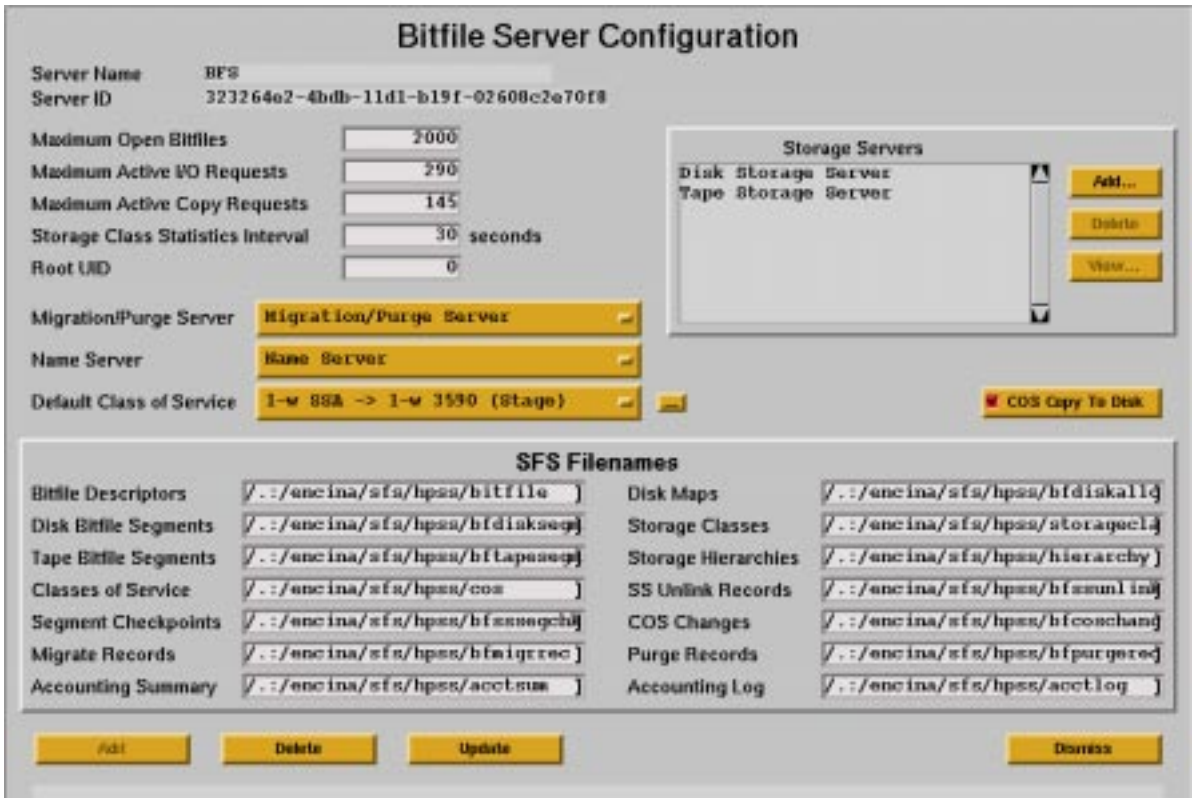


Figure 5-16 Bitfile Server Configuration Window

Bitfile Server Configuration Variables

Table 5-12 lists the fields on the Bitfile Server Configuration window and provides specific recommendations for configuring the BFS for use by HPSS.

Table 5-12 Bitfile Server Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	Descriptive name of the BFS. This name is copied over from the BFS general configuration entry.	This field cannot be modified. It is displayed for reference only.	Selected BFS descriptive name.
Server ID	The UUID of the Bitfile Server. This ID is copied over from the BFS general configuration entry.	This field cannot be modified. It is displayed for reference only.	Extracted from the BFS general configuration entry.
Migration/Purge Server	The descriptive name of the MPS.	Any configured MPS name from the pop-up list.	Name of first configured MPS found in the SFS file.
Name Server	The descriptive name of the NS.	Any configured NS name from the pop-up list. This field is required. If it is not specified, the creation of this configuration will fail.	Name of first defined NS found in the SFS file.
Default Class of Service	The default COS assigned to bitfiles that do not specify COS information on creation.	Any configured COS name from the pop-up list.	First COS name found in the COS SFS file.
Storage Servers	An array of descriptive names of Storage Servers for the BFS to connect to.	Any configured SS name from the pop-up list.	Names of all defined Storage Servers found in the SFS file.
Maximum Open Bitfiles	The maximum number of bitfiles that can be open concurrently in the BFS.	Any positive 32-bit integer value.	2000
Maximum Active I/O Requests	The maximum number of I/O type requests that can be open concurrently in the BFS.	Any positive 32-bit integer value.	190
	Advice: This value includes all requests that might result in data movement except migrate requests. This includes read, write, copy file and all non-background type stage requests. If an open request results in any stage other than a background stage, the open request is also counted. This value should be set high enough to cover the maximum number of active and queued requests expected in the system. This value cannot be higher than the maximum number of BFS threads -4. Since migrate requests are not counted, the maximum number of concurrent migrate requests expected should be computed and this value should be set no higher than the number BFS threads - 4 - max migrate requests. Each storage class that is in the process of being migrated will generate 1 migrate request for each copy being created.		

Table 5-12 Bitfile Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Maximum Active Copy Requests	The maximum number of copy type requests that can be active concurrently in the BFS.	Any positive 32-bit integer value.	95
	Advice: This value counts only copy file requests and various forms of stage requests. For copy file requests and non-background type stage requests, this limit results in the requests being given a BUSY error when the request is generated. Background stage requests are queued internally to BFS without using thread resources and up to 2000 can be queued before a BUSY error will be returned. This value will limit the actual number of threads that are busy in the background processing the background stage requests. The copyfile command that is generated by the background COS change thread is included in this count. This count does not include migrate requests.		
Storage Class Statistics Interval	An interval in seconds that indicates how often the BFS needs to contact each SS to get up-to-date statistics on each storage class that the SS manages. This information is used in load balancing across multiple storage classes.	Any positive 32-bit integer value.	180
	Advice: For systems with mostly disk or small files, the number should be set to a short interval (between 30 and 60 seconds). For systems with mostly large or tape files, this value should be set higher (e.g., 1 to 5 minutes).		
Root UID	The UID used to create root credentials for special calls to the NS by the BFS.	Valid Root User UID. Must match the Root User UID defined in the NS Configuration.	0
COS Copy To Disk	A flag affecting the COS changes for a bitfile. By default, when the COS of a bitfile is changed, the BFS copies the file to the highest level tape storage class in the target hierarchy. If this flag is ON, and if the target hierarchy has a disk storage class as its highest level, the BFS will copy the file to that disk storage class. Otherwise, the bitfile will be copied to the highest tape level.	ON, OFF	OFF
	Advice: This flag should be ON when changing the COS of a file to a duplicate (2-copies) COS. Otherwise, only the first copy of the file will be created on tape. The second copy will not be created until the file is staged to disk, modified and then migrated to tape.		

Table 5-12 Bitfile Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
SFS Filenames. The fields below list the names of the SFS files used by the BFS.			
Bitfile Descriptors	The file name of the SFS file where the bitfile descriptor information is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/bitfile
Disk Bitfile Segments	The file name of the SFS file where the bitfile disk segment information is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/bfdiskseg-ment
Tape Bitfile Segments	The file name of the SFS file where the bitfile tape segment information is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/bftapeseg-ment
Classes of Service	The file name of the SFS file where the COS Information is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/cos
Segment Checkpoints	The file name of the SFS file where the BFS checkpoints are stored.	Valid Encina file name.	/.:/encina/sfs/hpss/bfsssegchkpt
Migrate Records	The file name of the SFS file where the migrate records are stored.	Valid Encina file name.	/.:/encina/sfs/hpss/bfmigrrec
Accounting Summary	The file name of the SFS file where accounting summary records are stored.	Valid Encina file name.	/.:/encina/sfs/hpss/acctsum
Disk Maps	The file name of the SFS file where the bitfile disk map information is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/bfdiskallocrec
Storage Classes	The file name of the SFS file where the storage class information is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/storageclass
Storage Hierarchies	The file name of the SFS file where the storage hierarchy information is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/hierarchy
SS Unlink Records	The file name of the SFS file where the information representing storage segments to be unlinked is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/bfssunlink

Table 5-12 Bitfile Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
COS Changes	The file name of the SFS file where the information indicating which bitfiles need to have the COS changed is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/bfcoschange
Purge Records	The file name of the SFS file where the purge records are stored.	Valid Encina file name.	/.:/encina/sfs/hpss/bfpurgerec
Accounting Log	The file name of the SFS file where accounting log records are stored.	Valid Encina file name.	/.:/encina/sfs/hpss/acctlog

5.6.3 Configure the Storage Server Specific Information

The Storage Server specific configuration entry can be created using the Storage Server Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 5-3.

To add a new specific configuration, select the Storage Server entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The Storage Server Configuration window will be displayed with default values (Figure 5-17 and Figure 5-18 shown the screen image of the Disk and Tape Storage Server configuration window, respectively). If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Storage Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Storage Server Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the Storage Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Storage Server Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

When configuring a Disk Storage Server, the five statistics fields (Total Virtual Volumes, Total Allocated Volumes, Total Bytes, Used Bytes, and Free Bytes) in the Storage Server Configuration window will not be displayed. These fields cannot be set for Disk Storage Servers because the server computes these values each time it initializes and does not use the fields in the specific configuration record.

When configuring a Tape Storage Server, the initial values of the statistics fields are normally zero, but other values can be used. The server reads the configuration record when it initializes and updates the record as the statistics change. For this reason, SSM will not allow changes to be made to the Storage Server specific configuration while the server is running. Adjustments can only be made to the statistics by modifying the statistics fields when the server is not running.

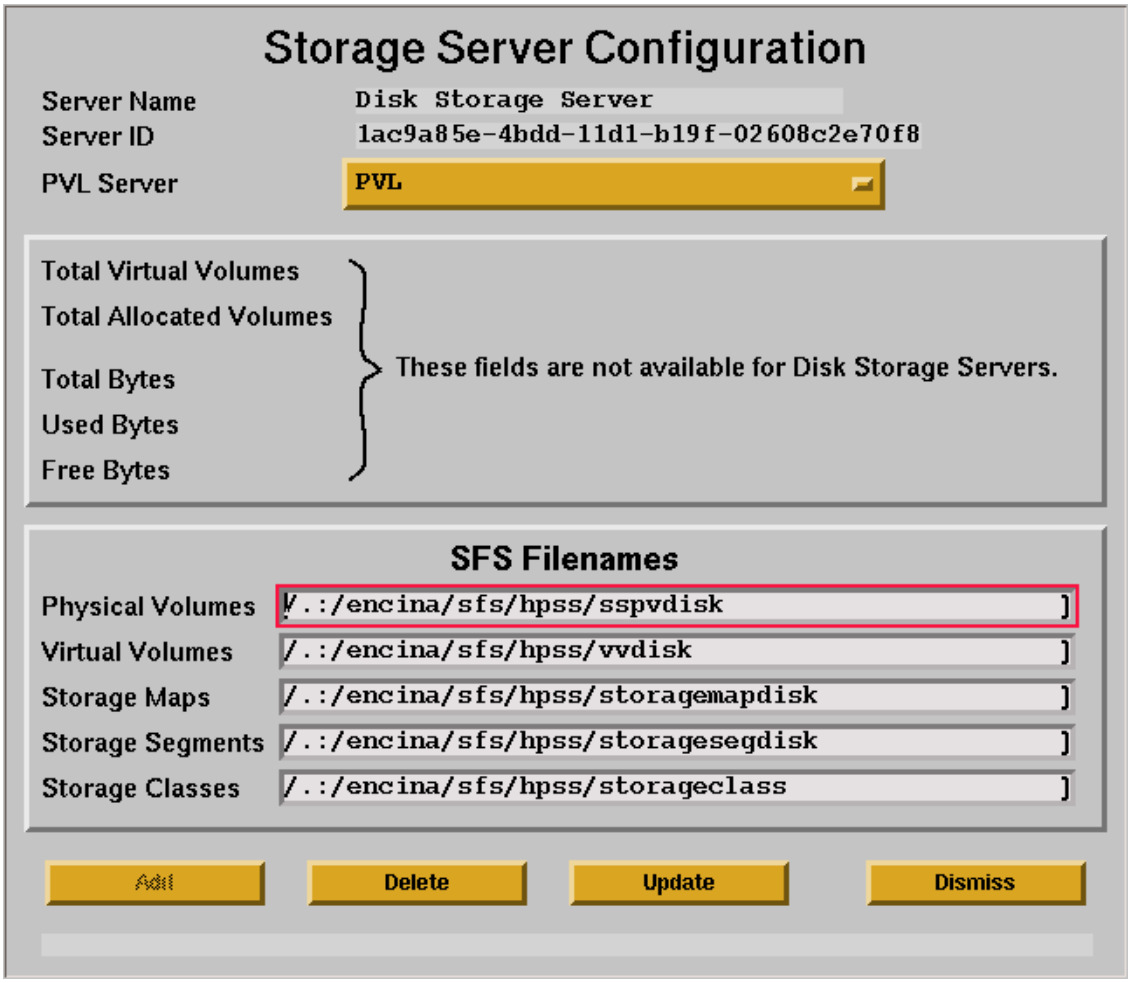


Figure 5-17 Disk Storage Server Configuration Window

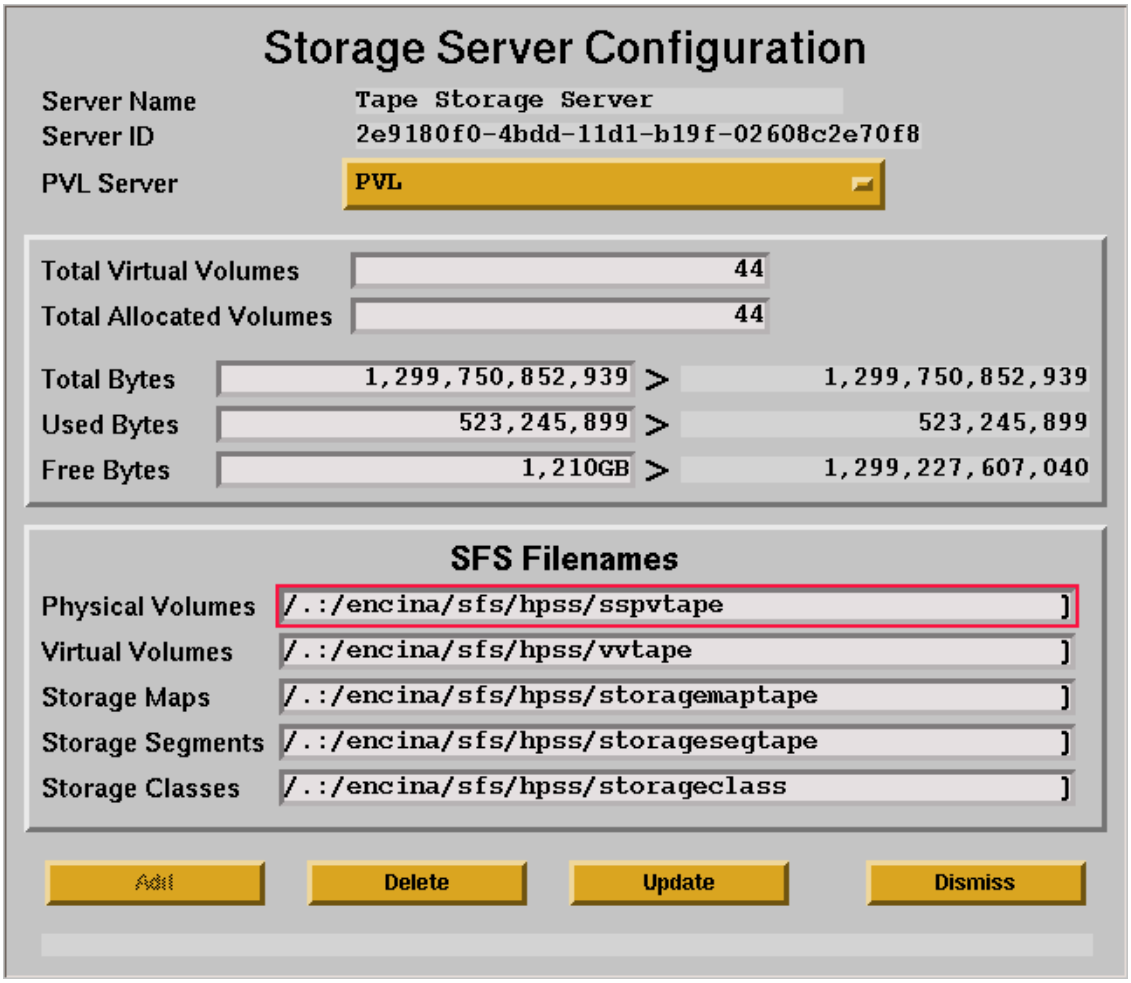


Figure 5-18 Tape Storage Server Configuration Window

Storage Server Configuration Variables

Table 5-13 lists the fields on the Storage Server Configuration window and provides specific recommendations for configuring a Storage Server for use by HPSS.

Table 5-13 Storage Server Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the SS. This name is copied from the SS general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected SS descriptive name

Table 5-13 Storage Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Server ID	The UUID of this SS. This ID is copied from the SS general configuration entry.	This field cannot be modified. It is displayed for reference only.	Extracted from the SS general server configuration entry.
PVL Server	The descriptive name of the PVL server to be used by the SS to communicate with the PVL.	Any configured PVL name from the popup list.	Name of the first PVL found in the SFS file.
Statistics Fields. The following five fields are displayed for Tape Storage Servers only.			
Total Virtual Volumes	The number of virtual volumes that are managed by this SS. This field is applicable to the Tape Storage Server only.	Any positive 64-bit integer value.	0
	Advice: This value should be set to zero when the specific configuration record is first created. The SS will calculate and update the configuration file as virtual volumes are added and deleted from the system. This value should only be changed when the SS is not running. See the documentation on "**settapestats**" for information about updating this value in established servers.		
Total Allocated Volumes	The number of virtual volumes that are either available for space allocation or full. This is the number of virtual volumes known to the server that are not in a "scratch" state. This field is applicable to the Tape Storage Server only.	Any positive 64-bit integer value.	0
	Advice: This value should be set to zero when the specific configuration record is first created. The SS will calculate and update the configuration file as virtual volumes are added and deleted from the system, and as volume states change. This value should only be changed when the SS is not running. See the documentation on "**settapestats**" for information about updating this value in established servers.		

Table 5-13 Storage Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Total Bytes	The total number of bytes of used and available storage known to the SS. This field is applicable to the Tape Storage Server only.	Any positive 64-bit integer value.	0
	Advice: This value should be set to zero when the specific configuration record is first created. The SS will calculate and update the configuration file as virtual volumes are added and deleted from the system. This number is an estimate of the amount of storage space, not an accurate number. This value should only be changed when the SS is not running. See the documentation on "**settapestats**" for information about updating this value in established servers.		
Used Bytes	The number of bytes written in storage segments. This field is applicable to the Tape Storage Server only.	Any positive 64-bit integer value.	0
	Advice: This value should be set to zero when the specific configuration record is first created. The SS will calculate and update the configuration file as storage segments are written, shortened, or deleted, and as volume state change. This value should only be changed when the SS is not running and only when the administrator is absolutely sure that the value is incorrect. See the documentation on "**settapestats**" for information about updating this value in established servers.		
Free Bytes	An estimate of the amount of unused storage space managed by the SS. This field is applicable to the Tape Storage Server only.	Any positive 64-bit integer value.	0
	Advice: This value should be set to zero when the specific configuration record is first created. The SS will calculate and update the configuration file as storage segments are written, shortened, or deleted (disk only), and as virtual volumes are added. This value should only be changed when the SS is not running. See the documentation on "**settapestats**" for information about updating this value in established servers.		
SFS Filenames. The fields below list the names of the SFS files used by the SS.			
Physical Volumes	The name of the Encina SFS file that contains the physical volume metadata.	Valid Encina file name. The file must not be shared with other Storage Servers.	/.:/encina/sfs/hpss/sspvdisk for Disk SS /.:/encina/sfs/hpss/sspvtape for Tape SS
	Advice: Use a name that is meaningful to the type of SS and metadata being stored.		

Table 5-13 Storage Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Virtual Volumes	The name of the Encina SFS file that contains the virtual volume metadata.	Valid Encina file name. The file must not be shared with other Storage Servers.	/.:/encina/sfs/hpss/vvdisk for Disk SS /.:/encina/sfs/hpss/vvtape for Tape SS
	Advice: Use a name that is meaningful to the type of SS and metadata being stored.		
Storage Maps	The name of the Encina SFS file that contains the storage map metadata.	Valid Encina file name. The file must not be shared with other Storage Servers.	/.:/encina/sfs/hpss/storagemapdisk for Disk SS /.:/encina/sfs/hpss/storagemaptape for Tape SS
	Advice: Use a name that is meaningful to the type of SS and metadata being stored.		
Storage Segments	The name of the Encina SFS file that contains the storage segment metadata.	Valid Encina file name. The file must not be shared with other Storage Servers.	/.:/encina/sfs/hpss/storagesegdisk for Disk SS /.:/encina/sfs/hpss/storagesegtape for Tape SS
	Advice: Use a name that is meaningful to the type of SS and metadata being stored.		
Storage Classes	The file name of the SFS file where the storage class information is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/storageclass

5.6.4 Configure the MPS Specific Information

The Migration / Purge Server specific configuration entry can be created using the Migration / Purge Server Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 5-3.

To add a new specific configuration, select the Migration / Purge Server entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The

Migration / Purge Server Configuration window will be displayed as shown in Figure 5-19 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Migration / Purge Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Migration / Purge Server Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the Migration / Purge Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Migration / Purge Server Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

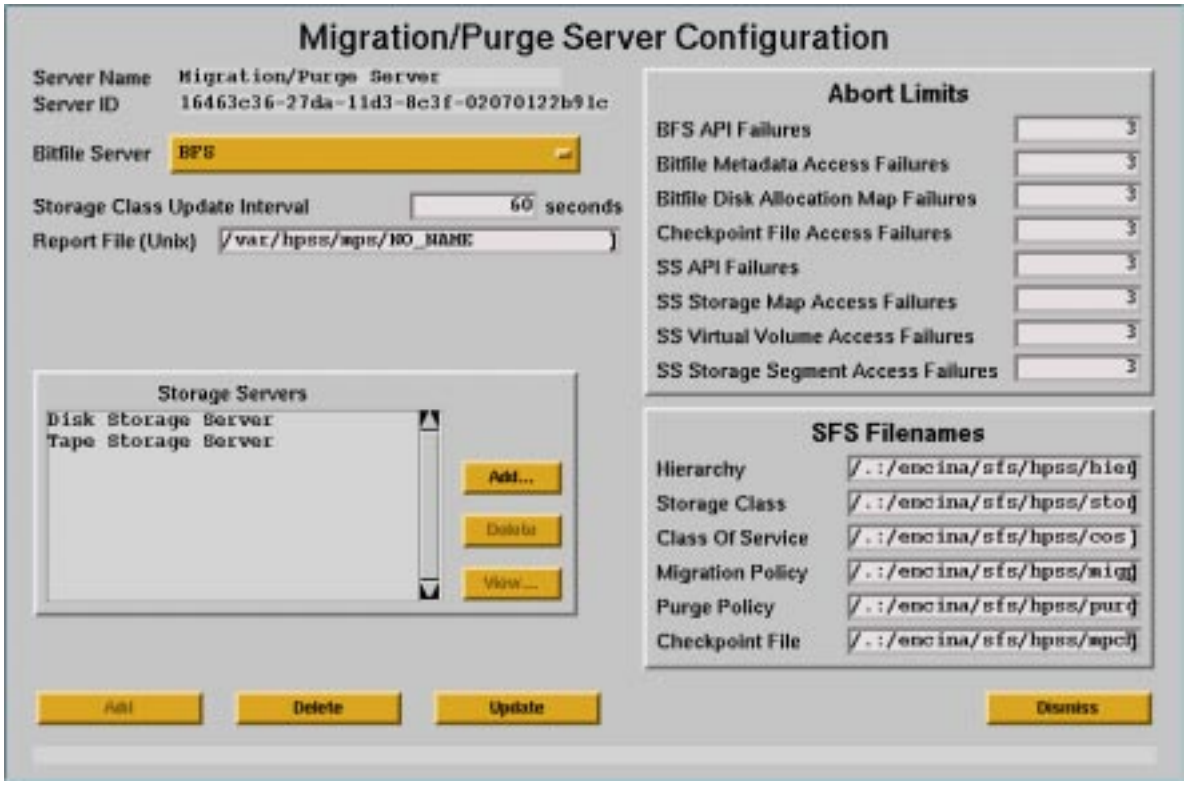


Figure 5-19 Migration / Purge Server Configuration Window

Migration/Purge Server Configuration Variables

Table 5-14 lists the fields on the HPSS Migration / Purge Server Configuration window and provides specific recommendations for configuring the MPS for use by HPSS.

Table 5-14 Migration/Purge Server Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the MPS. This name is copied over from the selected MPS general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected MPS descriptive name.
Server ID	The UUID of the MPS. This ID is copied over from the selected MPS general configuration entry.	This field cannot be modified. It is displayed for reference only.	Extracted from the MPS general configuration entry.
Bitfile Server	The descriptive name of the BFS that the MPS must communicate with.	Any configured BFS name from the pop-up list. This field is required.	First configured BFS name found in the Encina file.
Storage Class Update Interval	The interval, in seconds, that indicates how often the MPS will query each SS to get the latest data on each storage class that the SS manages. This is also the interval the MPS uses to check periodically to see whether it needs to initiate a purge operation on the storage class based on the associated purge policy.	Any positive integer between 10 and 600.	60 seconds
Report File (Unix)	A prefix string used by the MPS to construct a report filename. The full filename will consist of this string with a date string appended to it. MPS reports are generated every 24 hours. If the string specified is not an absolute path, the report file is created with respect to the current directory for the MPS process (usually the **/usr/lpp/hpss/bin** directory**)**. If the MPS reports are not desired, leave this field blank, or specify "/dev/null".	Any valid UNIX file name which is writable by MPS. If the reports are to be generated in a specific directory, use the full path and file name	blank - no reports will be generated.

Table 5-14 Migration/Purge Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Storage Servers	An array of the SS descriptive names with which the MPS must communicate.	Any SS name from the ADD list.	Names of all defined Storage Servers found in the Encina file.
Abort Limits. The following 8 fields describe the maximum failures before the MPS will terminate.			
BFS API Failures	MPS will terminate when the total number of consecutive failures of BFS API reaches this number.	Any positive number between 1 and 30.	3
Bitfile Metadata Access Failures	MPS will terminate when the total number of consecutive failures of reading the bitfile descriptor metadata reaches this number.	Any positive number between 1 and 30.	3
Bitfile Disk Allocation Map Failures	MPS will terminate when the total number of consecutive failures of reading the bitfile disk map metadata reaches this number.	Any positive number between 1 and 30.	3
CheckPoint File Access Failures	MPS will terminate when the total number of consecutive failures of reading or writing the MPS checkpoint metadata reaches this number.	Any positive number between 1 and 30.	3
SS API Failures	MPS will terminate when the total number of consecutive failures of SS API reaches this number.	Any positive number between 1 and 30.	3
SS Storage Map Access Failures	MPS will terminate when the total number of consecutive failures of reading the storage map metadata reaches this number.	Any positive number between 1 and 30.	3
SS Virtual Volume Access Failures	MPS will terminate when the total number of consecutive failures of reading the virtual volume metadata reaches this number.	Any positive number between 1 and 30.	3

Table 5-14 Migration/Purge Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
SS Storage Segment Access Failures	MPS will terminate when the total number of consecutive failures of reading the storage segment metadata reaches this number.	Any positive number between 1 and 30.	3
SFS Filenames. The fields below list the names of the SFS files used by the MPS.			
Hierarchy	The name of the SFS file where the storage hierarchy information is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/hierarchy
Storage Class	The name of the SFS file where the storage class information is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/storageclass
Class Of Service	The name of the SFS file where the COS information is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/cos
Migration Policy	The name of the SFS file where the migration policy information is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/migpolicy
Purge Policy	The name of the SFS file where the purge policy information is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/purgepolicy
Checkpoint File	The name of the SFS file where the migration / purge checkpoint information is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/mpchkpt

5.6.5 Configure the DMAP Gateway Specific Information

The DMAP Gateway specific configuration entry can be created using the DMAP Gateway Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 5-3.

To add a new specific configuration, select the DMAP Gateway entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The DMAP Gateway Configuration window will be displayed as shown in Figure 5-20 with default values. If

the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the DMAP Gateway entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The DMAP Gateway Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the DMAP Gateway entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The DMAP Gateway Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

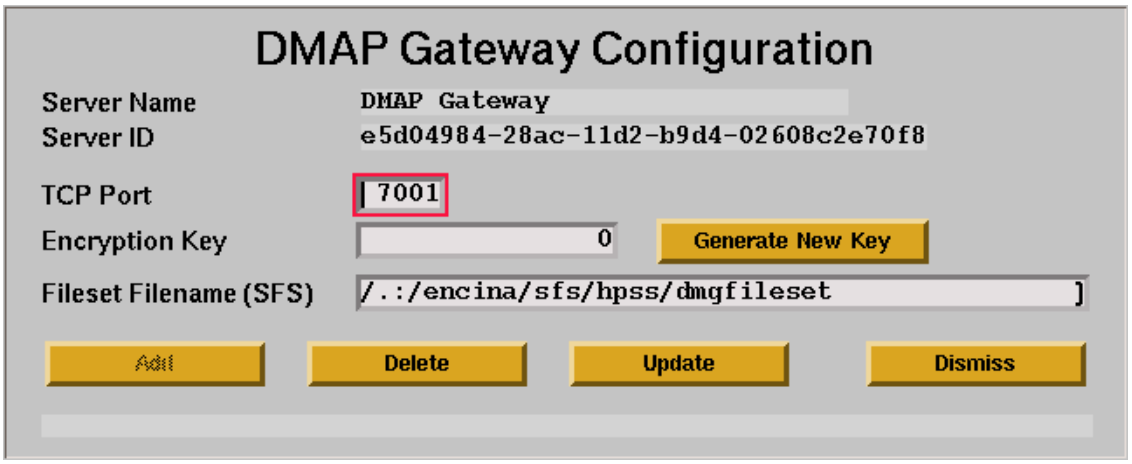


Figure 5-20 DMAP Gateway Configuration Window

DMAP Gateway Configuration Variables

Table 5-14 lists the fields on the HPSS DMAP Gateway Configuration window and provides specific recommendations for configuring the DMAP Gateway for use by HPSS.

Table 5-15 DMAP Gateway Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the DMAP Gateway. This name is copied over from the selected DMAP Gateway general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected DMAP Gateway descriptive name.
Server ID	The UUID of the DMAP Gateway. This ID is copied over from the selected DMAP Gateway general configuration entry.	This field cannot be modified. It is displayed for reference only.	Extracted from the DMAP Gateway general configuration entry.
TCP Port	The TCP port number used by the DMAP Gateway to listen for requests from HPSS/DMAP servers.	Any integer from 1 to 65,535.	7001
Encryption Key	A number used as an encryption key in message passing. A specific value can be typed in, or the **Generate New Key** button can be clicked to generate a random key value.	Any positive 64-bit integer, displayed as hexadecimal.	0
	Advice: Should not use key 0 which implies no protection. This key must be identical to the one defined in the **/var/hpss/hdm/hdm<id>/config.dat** file.		
Fileset Filename (SFS)	The name of the Encina SFS file containing the DMAP Gateway fileset information.	Valid Encina file name.	/.:/encina/sfs/hpss/dmgfileset

5.6.6 Configure the PVL Specific Information

The PVL specific configuration entry can be created using the PVL Server Configuration window. After the configuration entry is created, it can be viewed, updated or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 5-3.

To add a new specific configuration, select the PVL Server entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The PVL Server Configuration window will be displayed as shown in Figure 5-21 with default values. If the default

data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the PVL Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The PVL Server Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the PVL Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The PVL Server Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

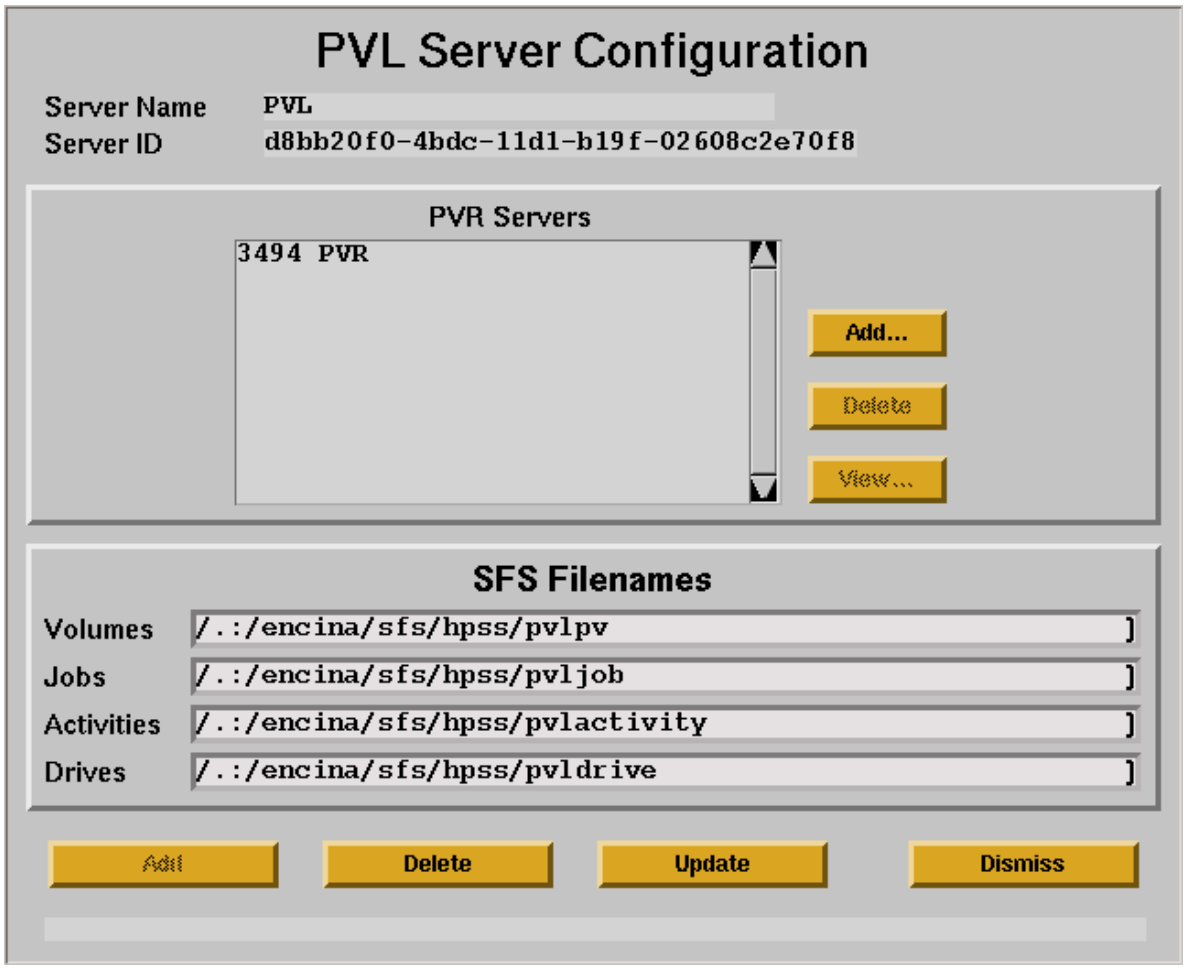


Figure 5-21 PVL Server Configuration Window

Physical Volume Library Configuration Variables

Table 5-16 lists the fields on the PVL Server Configuration window and provides specific recommendations for configuring the PVL for use by HPSS.

Table 5-16 Physical Volume Library Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the PVL. This name is copied over from the PVL general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected PVL server descriptive name.
Server ID	The UUID of the PVL. This ID is copied over from the PVL general configuration entry.	The UUID of the PVL. This field cannot be modified. It is displayed for reference only.	Extracted from the PVL general server configuration entry.
PVR Servers [P]	An array of descriptive names of PVRs for PVL to connect to.	Any configured PVR name from the ADD list.	All currently-defined PVRs found in the SFS file.
SFS Filenames. The fields below list the names of the SFS files used by the PVL Server.			
Volumes	The name of the Encina SFS file that maintains volume metadata for the PVL.	Valid Encina file name.	/.:/encina/sfs/hpss/pvlpv
	Advice: Use a name that is meaningful to the type of metadata being stored.		
Jobs	The name of the Encina SFS file that maintains job metadata for the PVL.	Valid Encina file name.	/.:/encina/sfs/hpss/pvljob
	Advice: Use a name that is meaningful to the type of metadata being stored.		
Activities	The name of the Encina SFS file that maintains activity metadata for the PVL.	Valid Encina file name.	/.:/encina/sfs/hpss/pvlactivity
	Advice: Use a name that is meaningful to the type of metadata being stored.		
Drives	The name of the Encina SFS file that maintains drive metadata for the PVL.	Valid Encina file name.	/.:/encina/sfs/hpss/pvldrive
	Advice: Use a name that is meaningful to the type of metadata being stored.		

5.6.7 Configure the PVR Specific Information

The PVR specific configuration entry can be created using the PVR Server Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

If you are configuring a PVR for StorageTek or IBM 3494/3495 PVRs, before proceeding with PVR configuration you should read Appendix K (StorageTek PVR Information), Appendix J (IBM 3494/3495 PVR Information) or Appendix L (ADIC AML Information) as appropriate. These appendices provide additional vendor-specific advice on PVR/robot configuration.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 5-3.

To add a new specific configuration, select the PVR Server entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The PVR Server Configuration window will be displayed (as shown in Figure 5-22, Figure 5-23, Figure 5-24, or Figure 5-25, depending on the selected PVR type) with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the PVR Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The PVR Server Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the PVR Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The PVR Server Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

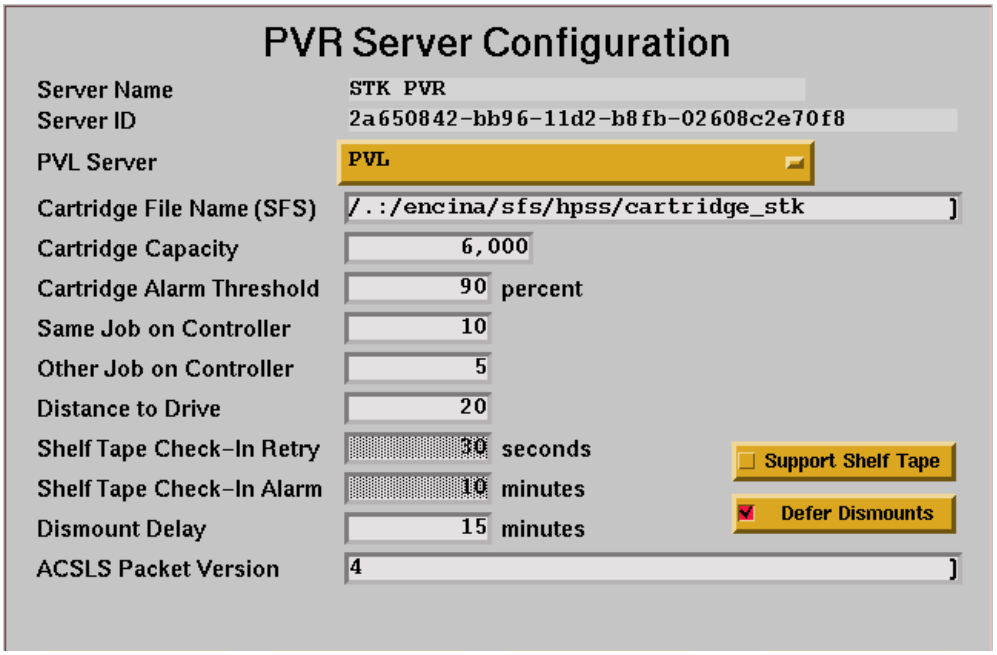


Figure 5-22 STK PVR Server Configuration Window

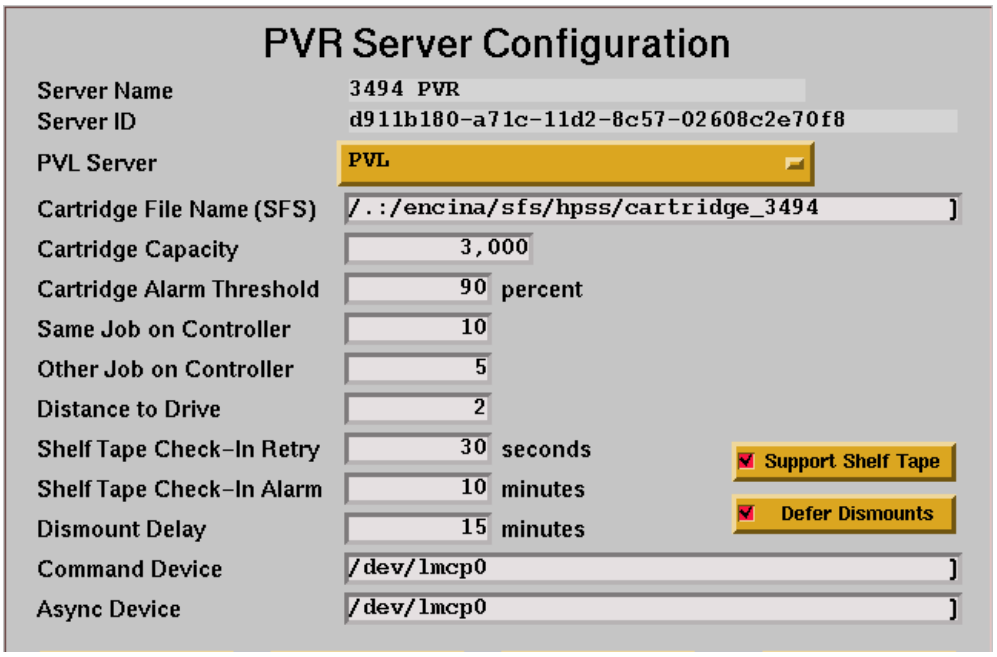


Figure 5-23 3494 PVR Server Configuration Window

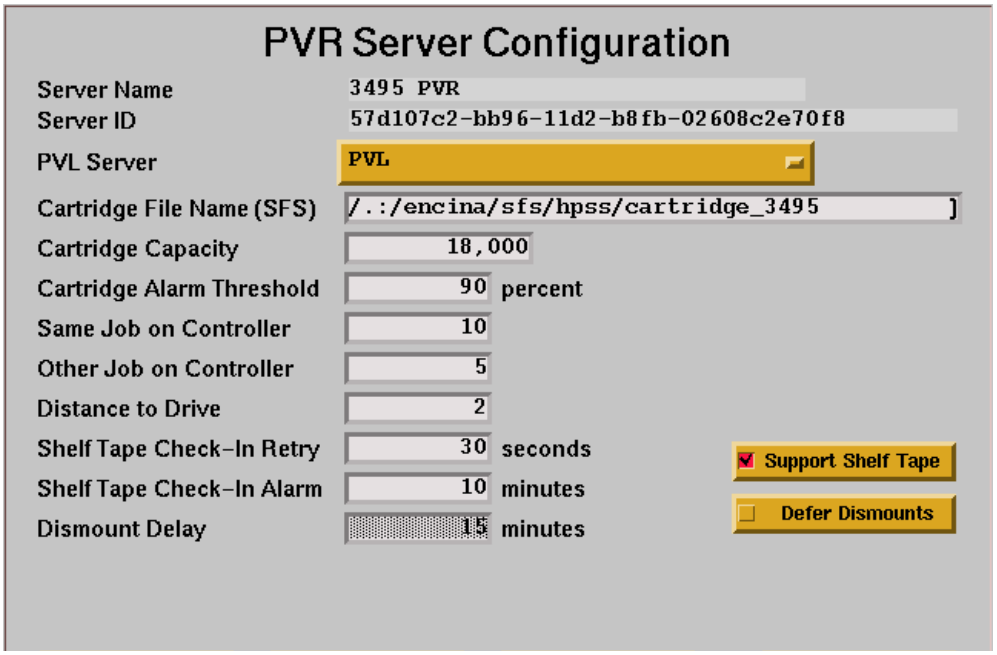


Figure 5-24 3495 PVR Server Configuration Window

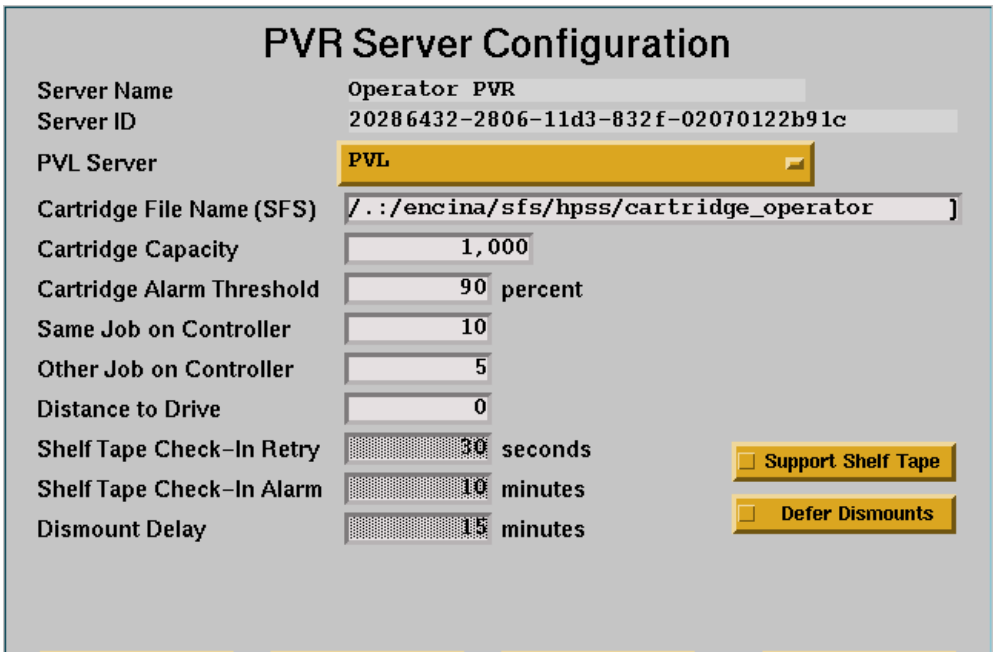


Figure 5-25 Operator PVR Server Configuration Window

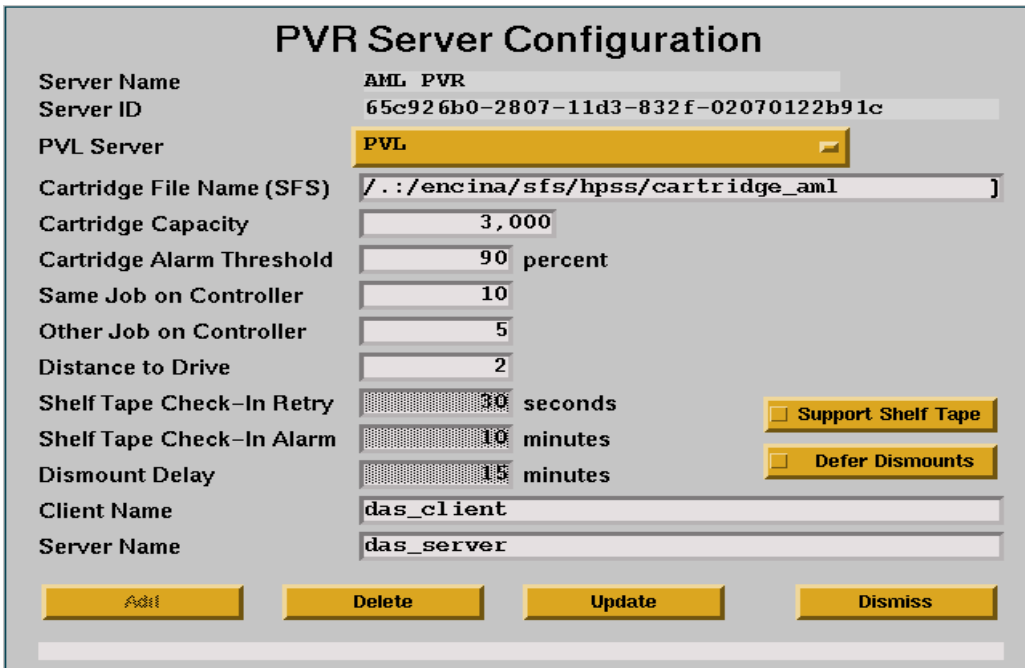


Figure 5-26 AML PVR Server Configuration Window

Physical Volume Repository Configuration Variables

Table 5-17 lists the fields on the PVR Server Configuration window and provides specific recommendations for configuring a PVR for use by HPSS.

Table 5-17 Physical Volume Repository Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the PVR. This name is copied over from the PVR general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected PVR server descriptive name.
Server ID	The UUID of this PVR. This ID is copied over from the PVR general configuration entry.	The UUID of this PVR. This field cannot be modified. It is displayed for reference only.	Extracted from the PVR general server configuration entry.
PVL Server	The descriptive name of PVL for the PVR to connect to.	Any configured PVL name from the pop-up list.	First PVL name found in the SFS file.

Table 5-17 Physical Volume Repository Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Cartridge File Name (SFS)	The name of the Encina file that maintains PVR cartridge metadata.	Valid Encina file name.	/.:/encina/sfs/hpss/cartridge_<robot>
	Advice: The cartridge file name for each PVR must be unique. Multiple PVRs cannot share the same SFS file for cartridges. Suggested names are: cartridge_operator for manual-mount PVRs cartridge_stk for StorageTek-based PVRs cartridge_3494 for IBM 3494 PVRs cartridge_aml for ADIC AML PVRs If two robots of the same type are managed by two different PVRs, be sure that each one has a different SFS file for cartridges.		
Cartridge Capacity	The total number of HPSS cartridges to be stored in this PVR. This is not a hard limit, but rather the point at which major alarms will be generated.	A positive 32-bit integer.	3495 - 18000 3494 - 3000 STK - 6000 Operator - 1000
Cartridge Alarm Threshold	The maximum percentage of cartridge capacity allowed in a PVR before a minor alarm is generated.	Any integer value between 1 and 100.	90
	Advice: Depending on operational policy, in some sites it is wise to leave a small number of free slots in a PVR to allow room for a small number of cartridge injections without the need for corresponding ejects.		
Same Job On Controller	The number of cartridges from this job mounted on this drive's controller. The larger the number, the harder the PVR will try to avoid mounting two tapes in the same stripe set on drives attached to the same controller. See Advice below for more information.	Any positive 32-bit integer value.	10
Other Job On Controller	The number of cartridges from other jobs mounted on this drive's controller. The larger the number, the harder the PVR will try to avoid mounting any two tapes on drives attached to the same controller. See Advice below for more information.	Any positive 32-bit integer value.	5

Table 5-17 Physical Volume Repository Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Distance to Drive	The number of units of distance from the cartridge to the drive. The larger the number, the harder the PVR will try to mount tapes on the closest drive to the tape's current location. See Advice below for more information.	Any positive 32-bit integer value.	3495 - 2 3494 - 2 STK - 20 Operator - 0
Advice: The Same Job on Controller, Other Job on Controller, and Distance to Drive values are used by the PVR when selecting a drive for a tape mount operation. The three values are essentially weights that are used to compute an overall score for each possible drive. After the score has been calculated, the drive with the lowest score is selected for the mount. If two or more drives tie for the lowest score, one drive is selected at random. The score is calculated as follows: Score = Weight 1 * Cartridges from this job mounted on this drive's controller + Weight 2 * Cartridges from other jobs mounted on this drive's controller + Weight 3 * Units of distance from the cartridge to the drive This method has the effect of distributing a striped tape mount across as many controllers as possible for the best performance. It also will try to pick controllers that are currently driving a minimum number of tapes. So, in an environment with many tape drives per controller, the best performance will be achieved by minimizing the load on any one controller.			
Shelf Tape Check-In Retry	The number of seconds the PVR will wait before asking the robot if a requested shelf tape has been checked-in. The PVR will continue checking at this interval until the tape is checked-in. This field is only active if the Support Shelf Tape button is checked	Any positive 32-bit integer value	30 seconds
Shelf Tape Check-In Alarm	The PVR will periodically log alarm messages when a requested shelf tape has not been checked-in. This field specifies the number of minutes between alarms. This field is only active if the Support Shelf Tape button is checked.	Any positive 32-bit integer value	10 minutes
Dismount Delay	The number of minutes that dismounts are delayed after the last data access. This field is only active if the Defer Dismounts button is checked.	Any positive 32-bit integer value	15 minutes

Table 5-17 Physical Volume Repository Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Support Shelf Tape	A toggle button. If ON, the PVR will support the removal of cartridges from the tape library. If OFF, the PVR will not support the removal of cartridges from the tape library.	ON, OFF	OFF
Defer Dismounts	A flag to determine whether all tape dismounts in the drives manages by the PVR will be delayed when the associated PVL jobs are completed. If ON, the PVL will delay the dismounting of a tape cartridge until the drive is required by another job or until an approximate 15 minutes time limit is exceeded.	ON, OFF	OFF
ACSLS Packet Version (STK Only)	The packet version used by STK's ACSLS software. See the *STK Automated Cartridge System Library Software (ACSLS) System Administrator's Guide* for details. The environment variable **ACSAPI _PACKET _VERSION** will override the value entered in this field. If neither are set, a default value of 3 is used.	3 or 4	4
	Advice: ACSLS 4.0 generally uses packet version 3; ACSLS 5.0 generally uses packet version 4.		
Command Device (3494/3495 Only)	The name of the device that the PVR can use to send commands to the 3494/3495 robot. The environment variable **HPSS_3494_COMMAND _DEVICE** will override the value entered in this field.	Any device; generally /dev/lmcpX for AIX systems; symbolic library name defined in **/etc/ibmatl.conf** for Solaris systems	/dev/lmcp0

Table 5-17 Physical Volume Repository Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Async Device (3494/3495 Only)	The name of the device that the PVR can use to receive replies from the 3494/3495 robot. The environment variable **HPSS_3494_ASYNC_DEVICE** will override the value entered in this field.	Any device; generally /dev/lmcpX for AIX systems; symbolic library name defined in **/etc/ibmatl.conf** for Solaris systems	/dev/lmcp0
	Advice: For Block Multiplexer Channel (BMUX)-attached IBM robots, the Async Device must be different from the Command Device. For TTY and LAN-attached robots, the devices can be the same.		
Client Name	The name of the client requesting authorization from the Distributed Automated Media Library Server.	Any alphanumeric string of length <= 64	none
	Additional Info: DAS software, which executes on the OS/2 controller PC, allows different clients to control the AML robotics system. DAS uses the client name to determine the access permission of the requesting client to the AML's storage positions, drives, and Insert/Eject units. Access configurations for clients are set in the configuration file C:\DAS\ETC\CONFIG on the OS/2 PC. The client name can be up to 64 alphanumeric characters in length and is case sensitive.		
Server Name	TCP/IP host name or IP address of the AML OS/2-PC DAS server.	Any alphanumeric string of length <= 64	none
	Additional Info: This value must be defined in the network domain server and must be resolvable during DAS start. The server name is set in the configuration file C:\CONFIG.SYS on the OS/2 PC. The server name can be up to 64 alphanumeric characters long and can include up to six dots ('.').		

5.6.8 Configure the Mover Specific Information

The Mover (MVR) specific configuration entry can be created using the Mover Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 5-3.

To add a new specific configuration, select the Mover entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The Mover Configuration window will be displayed as shown in Figure 5-27 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Mover entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Mover Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the Mover entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Mover Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

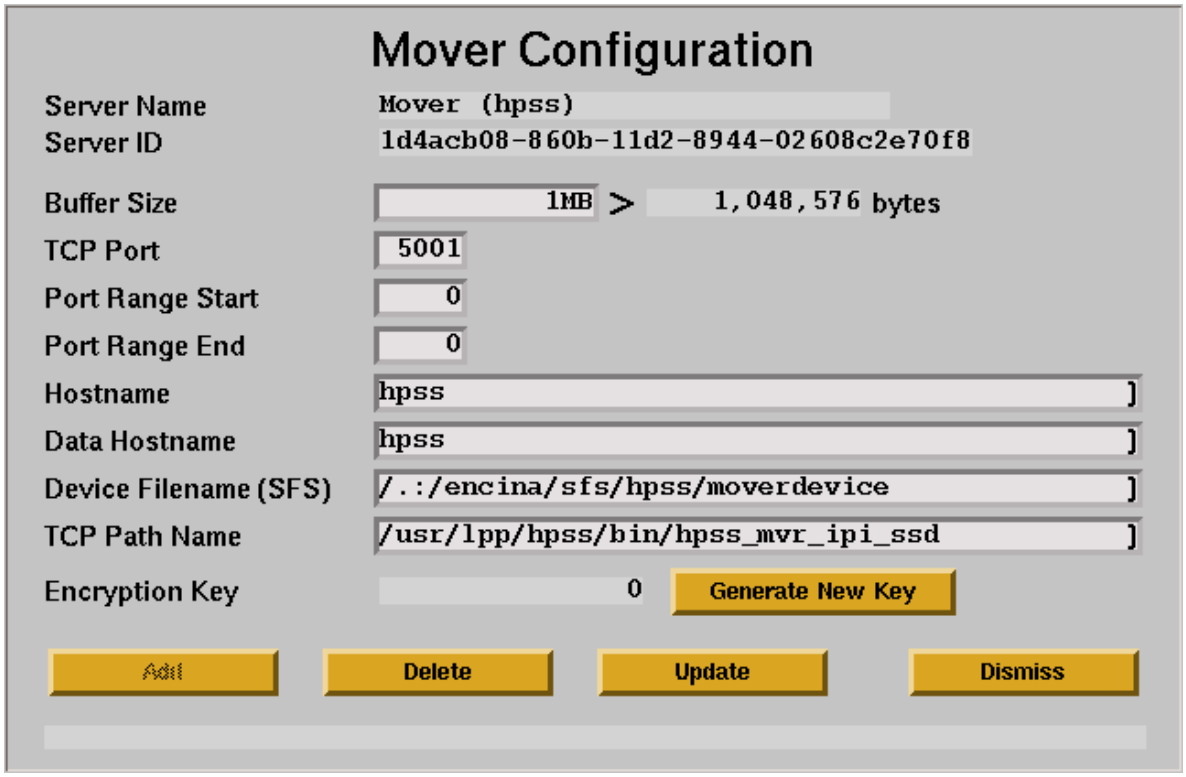


Figure 5-27 Mover Configuration Window

Mover Configuration Variables

Table 5-18 lists the fields on the Mover Configuration window and provides specific recommendations for configuring the MVR for use by HPSS.

Table 5-18 Mover Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the MVR. This name is copied over from the selected MVR general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected MVR descriptive name.
Server ID	The UUID of the MVR. This ID is copied over from the selected MVR general configuration entry.	The UUID of the MVR. This field cannot be modified. It is displayed for reference only.	Extracted from the MVR general configuration entry.
Buffer Size	The buffer size (of each buffer) used for double buffering during data transfers.	The minimum mover buffer size is the size of smallest block size for any device the Mover will handle. The maximum value will be bounded by the available system memory and the number of concurrent mover requests anticipated.	1,048,576
	Advice: This value should be tuned based on device and networking configuration and usage. The trade-off for this value is that large buffer sizes will use more system memory and may be inefficient for small transfers (e.g., if the mover buffer size is 4MB, but client requests are 512KB, the Mover will not achieve any double buffering benefit because the entire amount of the transfer fits in one mover buffer). A smaller buffer size will cause device and network I/O to be broken more often, usually resulting in reduced throughput rates for all but the smallest transfers.		
TCP Port	The TCP/IP port number used for the TCP/IP listen process to receive connections.	Should be a value over 5000.	5001
	Advice: The port number must be unique for MVRs running in the same node. For Movers running in non-DCE mode, the Mover will internally use the port one greater than the one configured in this field on the non-DCE/Encina platform during initialization (e.g., if 5001 is entered, port 5002 is used on the non-DCE/Encina node); therefore available port ranges on both nodes must be taken into consideration when selecting this value.		

Table 5-18 Mover Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Port Range Start	The beginning of a range of local TCP/IP port numbers to be used by the Mover when connecting to clients (required by some sites for communication across a firewall). If zero, the operating system will select the port number; otherwise the Mover selects a local port number between Port Range Start and Port Range End (inclusive).	Zero or any valid TCP port number to which the Mover may bind (that is less than or equal to the value of Port Range End).	0
	Advice: If non-zero, this field must be less than or equal to Port Range End. If this field is zero, Port Range End field must also be zero.		
Port Range End	Used in conjunction with Port Range Start (See above).	Zero or any valid TCP port number to which the Mover may bind (that is greater than or equal to the value of Port Range Start).	0
	Advice: If non-zero, this field must be equal or greater than Port Range Start. If this field is zero, Port Range Start field must also be zero.		
Hostname	The name of the host interface used by the TCP/IP listen process.	Valid host name for the network interfaces on the MVR machine.	Extracted from the MVR general configuration entry.
	Advice: If the Mover is running in non-DCE mode, this field must correspond to a network interface on the non-DCE/Encina node (to which the Storage Servers and PVL will connect when communicating with the Mover), whereas the Execute Hostname set in the Mover's basic configuration corresponds to a network interface on which the Mover DCE/Encina processes run.		
Data Hostname	The host network interface name to be used by the Mover when creating listen ports for data transfers (used during migration, repacking, and staging operations).	Valid host name for the network interfaces on the MVR machine.	Extracted from the MVR general configuration entry.
	Advice: If the Mover is running in non-DCE mode, this field must correspond to a network interface on the non-DCE/Encina node.		
Device Filename (SFS)	The name of the Encina SFS file containing MVR device configuration data.	Valid Encina file name.	/.:/encina/sfs/hpss/moverdevice
	Advice: Use a name that is meaningful to the type of metadata being stored. One device configuration file may be used to support multiple MVRs.		

Table 5-18 Mover Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
TCP Path Name	The pathname of the MVR TCP/IP listen executable.	Fully qualified file name of the MVR TCP/IP listen executable.	/usr/lpp/hpss/bin/hpss_mvr_tcp
	Advice: The TCP MVRs currently supported are listed below. All TCP MVRs support common disk/tape interfaces, TCP/IP, and shared memory data transfers: **Name** — **Options Supported** hpss_mvr_tcp — Standard Disk/Tape Devices hpss_mvr_ipi — IPI-3 hpss_mvr_ssd — SCSI 3490E/3590 hpss_mvr_omi — SCSI Redwood/Timberline hpss_mvr_dd2 — Ampex DST-312 hpss_mvr_dd2_ipi — Ampex DST-312 hpss_mvr_ipi_ssd — IPI-3, SCSI 3490E/3590 hpss_mvr_ipi_omi — IPI-3, SCSI Redwood/Timberline		
Encryption Key	An encryption key used to secure the MVR's IOD/IOR interface.	This value can only be changed by clicking on the **Generate New Key** button.	0
	Advice: A non-zero value must be configured for the security routines to allow any client access to this interface.		

5.6.9 Configure the Log Daemon Specific Information

The Log Daemon specific configuration entry can be created using the Logging Daemon Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 5-3.

To add a new specific configuration, select the Log Daemon entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The Logging Daemon Configuration window will be displayed as shown in Figure 5-28 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Log Daemon entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Logging Daemon Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the Log Daemon entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Logging Daemon Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

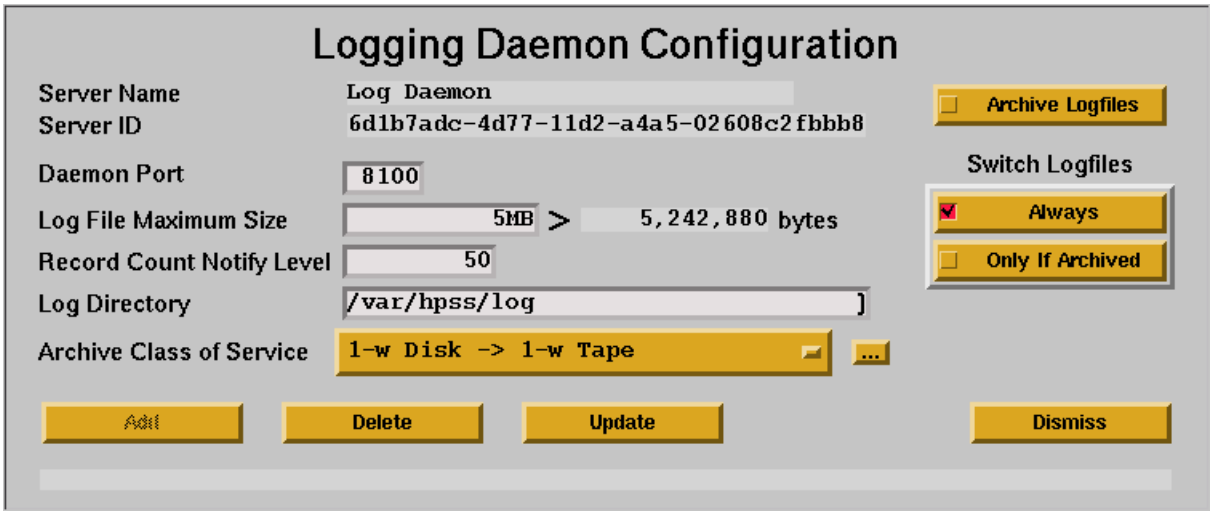


Figure 5-28 Logging Daemon Configuration Window

Log File Archival

The Log Daemon can be configured to automatically archive log files to HPSS. If log files are to be archived, the **/log** directory must be created in the HPSS root directory. The **/log** directory can be created by the root user using **ftp** as follows:

ftp "node" "HPSS Port"
Login as root user
mkdir /log
quote site chown hpss_log /log
quote site chmod 744 /log

Log Daemon Configuration Variables

Table 5-19 lists the fields on the Logging Daemon Configuration window and provides specific recommendations for configuring a Log Daemon for use by HPSS.

Table 5-19 Log Daemon Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the Log Daemon. This name is extracted over from the Log Daemon general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected Log Daemon descriptive name.
Server ID	The UUID of the Log Daemon. This ID is copied over from the Log Daemon general configuration entry.	The UUID of the Log Daemon. This field cannot be modified. It is displayed for reference only.	Extracted from the Log Daemon general configuration entry.
Daemon Port	The port number for communication between the Log Client and the Log Daemon.	Any positive 32-bit integer value not previously assigned.	8100
	Advice: Ensure that the port number assigned does not conflict with any other executing application. The port value must be a different value than the port number in the Log Client configuration entries.		
Log File Maximum Size	The maximum size in bytes of the central log file. Once this size is reached, logging will switch to a second log file. The log file that filled up will then be archived to an HPSS file if the Archive Flag is on.	A positive integer up to the maximum file size allowed by the operating system.	5,242,880
	Note: Since two log files are allocated to accommodate log switching and log file archiving, the amount of space used by logging will be twice the specified value.		
Record Count Notify Level	The difference in the log record count since the previous SSM notification that must be reached to trigger a subsequent notification to SSM when any registered log file attribute changes.	Any positive 32-bit integer value.	50
	Advice: Since log file attributes change each time a log record is written (e.g., current log record), this variable is used to prevent transmission of excessive messages to SSM as a result of logging activity. Setting the variable to a small value will result in increased traffic to SSM.		

Table 5-19 Log Daemon Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Log Directory	The name of the directory in which log files are stored.	Any valid UNIX path name. The string length (in bytes) is limited to a minimum of the operating system maximum allowed file name size, or 1024.	/var/hpss/log
Archive Class of Service	The COS that will determine where HPSS logs are archived. When a log file fills, it will be archived according to this class of service if the Archive Logfiles Flag is set.	Any configured COS name from the pop-up list.	First COS name found in the SFS file.
Archive Logfiles	A flag that indicates whether log files should be automatically archived when they fill.	ON, OFF	OFF
Switch Logfiles	A flag that indicates whether a switch to the second log file should be performed if the archive of the second log file has not yet completed.	Always, Only If Archived	Always
	Advice: Use the default value of **Always**. Although a log may not get archived, the alternative may be a loss of logged messages to the current log from executing servers.		

5.6.10 Configure the Log Client Specific Information

The Log Client specific configuration entry can be created using the Logging Client Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 5-3.

To add a new specific configuration, select the Log Client entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The Logging Client Configuration window will be displayed as shown in Figure 5-29 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Log Client entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Logging Client Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the Log Client entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Logging Client Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

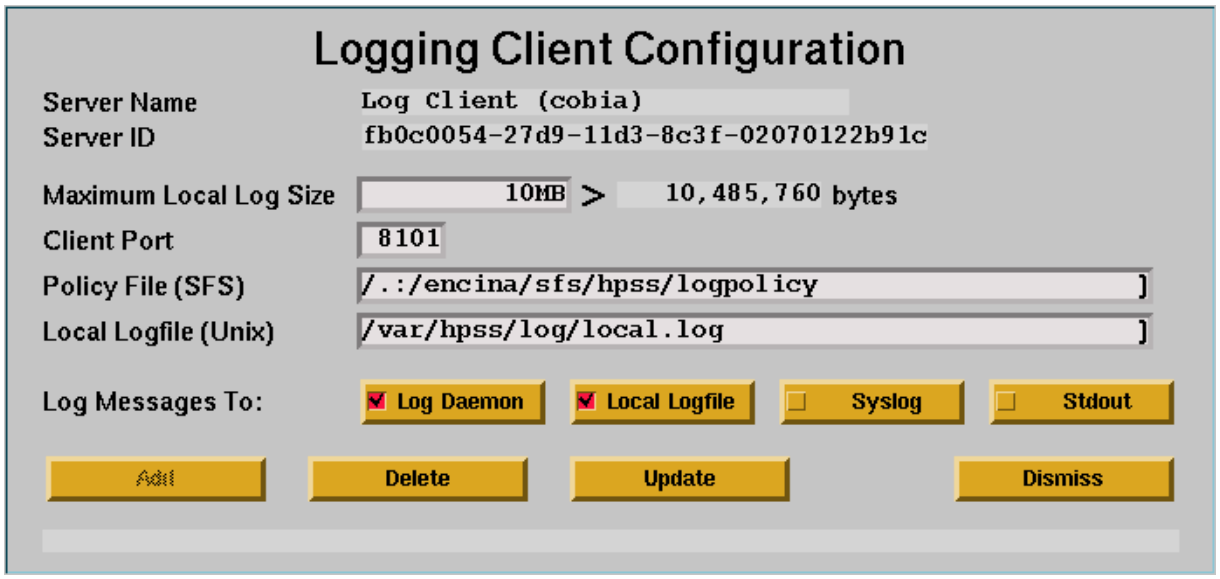


Figure 5-29 Logging Client Configuration Window

Log Client Configuration Variables

Table 5-20 lists the fields on the Logging Client Configuration window and provides specific recommendations for configuring a Log Client for use by HPSS.

Table 5-20 Log Client Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the Log Client. This name is copied over from the Log Client general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected Log Client descriptive name.

Table 5-20 Log Client Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Server ID	The UUID of the Log Client. This ID is copied over from the Log Client general configuration entry.	The UUID of the Log Client. This field cannot be modified. It is displayed for reference only.	Extracted from the Log Client general configuration entry.
Maximum Local Log Size	The maximum size in bytes of the local log file. Once this size is reached, the log will be reused in a wraparound fashion. The local log is not automatically archived.	A positive integer up to the maximum file size allowed by the operating system.	5,242,880
Client Port	The port number for communication between the Log Client and the HPSS Servers.	Any positive 32-bit integer value not previously assigned.	8101
	Advice: Ensure that the specified port is not being used by other executing applications. The port number must be a different number than the one used by the Log Daemon.		
Policy File (SFS)	The name of the Encina SFS file in which logging policy records are stored.	Valid Encina file name.	/.:/encina/sfs/hpss/logpolicy
	Advice: Use a name that is meaningful to the type of metadata being stored.		
Local Logfile (Unix)	The fully qualified path name of the Log Client-formatted log file. If Local LogFile is specified in the **Log Messages To** field, those messages sent to this instance of the Log Client will be formatted and written to the designated file name.	Any valid POSIX path name. The string length (in bytes) is limited to a minimum of the operating system maximum allowed file name size or 1024.	/var/hpss/log/local.log
	Note: The specified file name will contain formatted messages from HPSS applications executing only from the node on which this instance of the Log Client is executing. This option is provided as a convenience feature. All HPSS messages will be written to a central log if Log Daemon is specified in the **Log Messages To** field.		

Table 5-20 Log Client Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Log Messages To:	A mask of options to apply to local logging.	A combination of the following values: Log Daemon—Send log messages to the central log. Local LogFile—format and log messages from servers on the same node as the Log Client to a local file. It is an error if both Local LogFile and Standard Output are both specified. Syslog—format and log messages locally to syslog. Stdout—Send messages to standard output.	Log Daemon, Local Log File
	Advice: If neither Local LogFile nor Syslog is specified, no local logging will occur. If Log Daemon is not specified, messages from HPSS processes executing on the same node as this Log Client will not be written to the central log. The Syslog option should be used with care. The syslog file will grow without bound until it is deleted / truncated, or until the file system runs out of space.		

5.6.11 Configure the Metadata Monitor Specific Information

The Metadata Monitor specific configuration entry can be created using the Metadata Monitor Configuration window. After the configuration entry is created, it can be viewed, updated or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 5-3.

To add a new specific configuration, select the Metadata Monitor entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The Metadata Monitor Configuration window will be displayed as shown in Figure 5-30 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Metadata Monitor entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Metadata Monitor Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the Metadata Monitor entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Metadata

Monitor Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

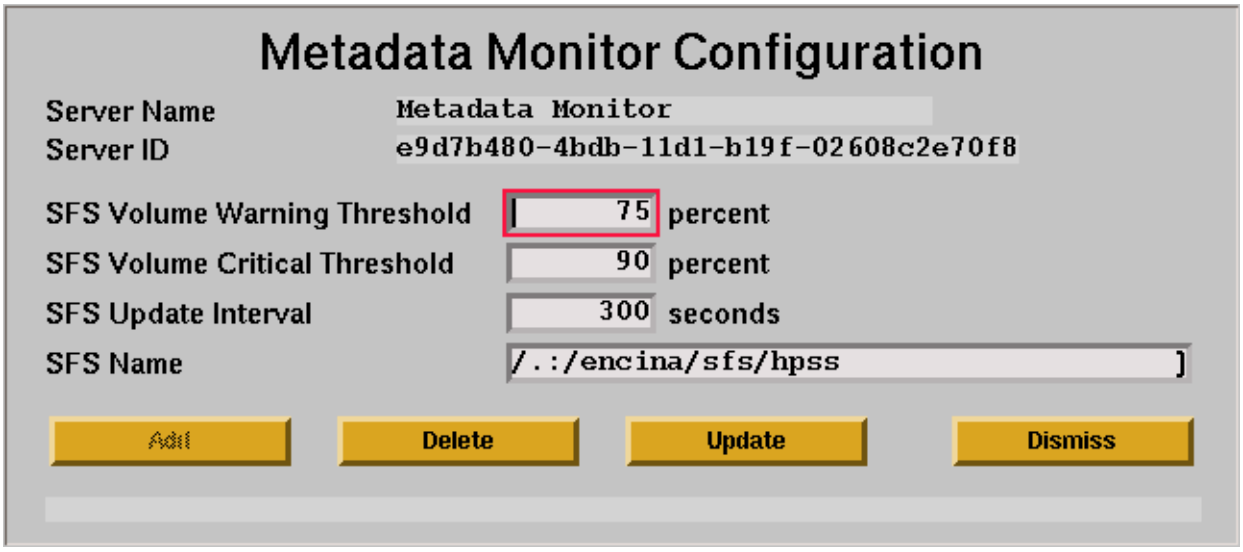


Figure 5-30 Metadata Monitor Configuration window

Metadata Monitor Configuration Variables

Table 5-21 lists the fields on the Metadata Monitor Configuration window and provides specific recommendations for configuring a MMON for use by HPSS.

Table 5-21 Metadata Monitor Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the MMON. This name is copied over from the MMON general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected MMON descriptive name.

Table 5-21 Metadata Monitor Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Server ID	The UUID of the Metadata Monitor. This ID is copied over from the MMON general configuration entry.	The UUID of the MMON. This field cannot be modified. It is displayed for reference only.	Extracted from the MMON general configuration entry.
SFS Volume Warning Threshold	A percentage value that indicates when warning alarms should be issued. The MMON will issue warning alarms to SSM when the space used by any SFS volume reaches this percentage.	Integer value between 1 and 100.	75
SFS Volume Critical Threshold	A percentage value that indicates when critical alarms should be issued. The MMON will issue critical alarms to SSM when the space used by any SFS volume reaches this percentage.	Integer value between 1 and 100.	90
SFS Update Interval	An indication of how often (in seconds) the SFS server should be queried.	Integer value between 60 and 600.	300
SFS Name	The name of the Encina SFS server to monitor.	Valid DCE CDS name that refers to a valid Encina SFS server.	/.:/encina/sfs/hpss

5.6.12 Configure the NFS Daemon Specific Information

The NFS Daemon specific configuration entry can be created using the NFS Daemon Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 5-3.

To add a new specific configuration, select the NFS Daemon entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The NFS Daemon Configuration window will be displayed as shown in Figure 5-31 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the NFS Daemon entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The NFS Daemon Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the NFS Daemon entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The NFS Daemon Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

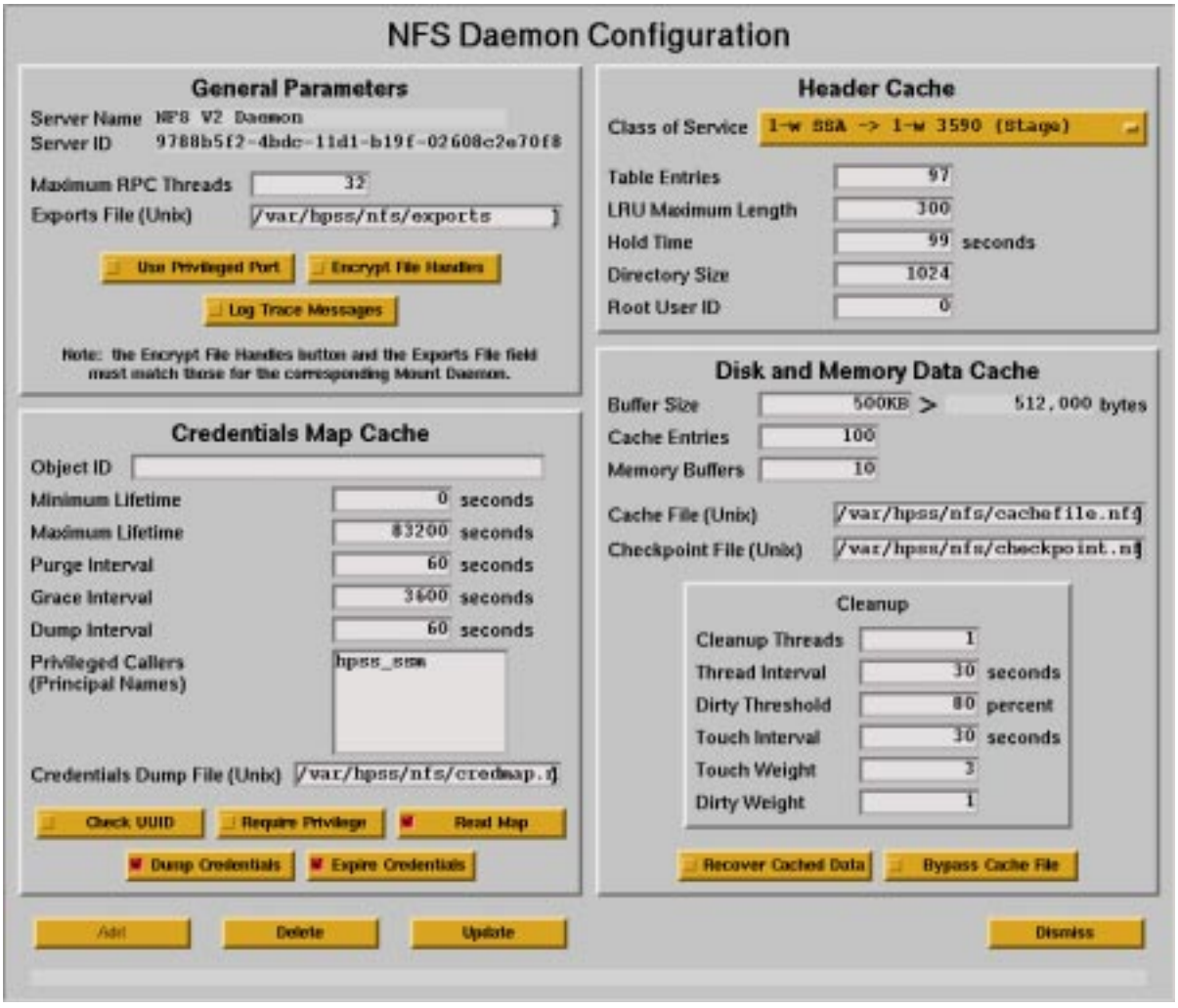


Figure 5-31 NFS Daemon Configuration Window

NFS Daemon Configuration Variables

Table 5-22 lists the fields on the NFS Daemon Configuration window and provides specific recommendations for configuring the NFS Daemon for use by HPSS.

Table 5-22 NFS Daemon Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
General Parameters. The following seven fields define general information for the NFS Daemon.			
Server Name	The descriptive name of the NFS Daemon. This name is copied over from the NFS Daemon general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected NFS Daemon descriptive name
Server ID	The UUID of the NFS Daemon. This ID is copied over from the NFS Daemon general configuration entry.	The UUID of the NFS Daemon. This field cannot be modified. It is displayed for reference only.	Extracted from the NFS Daemon general configuration entry.
Maximum RPC Threads	The number of NFS requests that can be processed concurrently.	Any positive 32-bit integer value	32
	Advice: Set this field based on the maximum number of concurrent requests expected. In general, it should be at least 8.		
Exports File (Unix)	The name of a UNIX file containing the HPSS directory / file exported for NFS access.	Valid and fully qualified UNIX file name.	/var/hpss/nfs/exports
Use Privileged Port	A flag that indicates whether the NFS clients will use privileged port. If set, the NFS clients must use port numbers less than 1024.	ON, OFF	OFF
Encrypt File Handles	A flag that indicates whether incoming and outgoing file handles will contain an encrypted checksum.	ON, OFF	OFF
Log Trace Messages	A flag that indicates whether the trace messages will be sent to the logging services.	ON, OFF	OFF

Table 5-22 NFS Daemon Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Credentials Map Cache. The following 13 fields are used to map the client user credentials to the HPSS user credentials.			
Object ID	The DCE UUID that identifies the credential mapping interface object	Valid DCE UUID.	NULL
Minimum Lifetime	The minimum time, in seconds, for which a credentials mapping is valid.	Any positive 32-bit integer value.	0 seconds
Maximum Lifetime	The maximum time, in seconds, for which a credential mapping is valid.	Any positive 32-bit integer value.	83200 seconds
Purge Interval	The time interval, in seconds, between credentials map purge operations.	Any positive 32-bit integer value.	60 seconds
Grace Interval	An interval, in seconds, to indicate how long after a credential's last use before it will be expired.	Any positive 32-bit integer value.	3600 seconds
Dump Interval	An interval, in seconds, to determine how often the credentials map cache is checkpointed to a UNIX file.	Any positive 32-bit integer value.	60 seconds
	Advice: This value should be based on how often entries are added/removed from the map. If this value is set too high, the user may not be able to access files after an NFS crash. If it is set too low, it may increase the system overhead. This field is used only if the UID option in the Export file is set.		
Privileged Callers (Principal Names)	The list of principal names of those callers permitted to change entries in the NFS credentials map other than their own.	Valid DCE principal name.	hpss_ssm
Credentials Dump File (Unix)	The name of a UNIX file to checkpoint the credentials map cache.	Valid, fully qualified UNIX file name.	/var/hpss/nfs/credmap.nfs2
Check UID	A flag that indicates whether users not on the privileged callers list may only make entries for their own DCE UID.	ON, OFF	OFF

Table 5-22 NFS Daemon Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Require Privilege	A flag that indicates whether only the users on the privileged callers list may change the credentials map.	ON, OFF	OFF
Read Map	A flag that indicates whether the credentials map will be read from a file at startup time.	ON, OFF	ON
	Advice: This flag should be set to OFF when the system is first configured and set to ON when the system is restarted after a crash.		
Dump Credentials	A flag that indicates whether the credentials map will be periodically checkpointed to a UNIX file.	ON, OFF	ON
Expire Credentials	A flag that indicates whether the credentials map entries will be expired and removed based on the grace and purge intervals.	ON, OFF	ON
Header Cache. The following six fields are used to cache names and attributes of the HPSS file objects.			
Class of Service	The name of the COS used when creating the NFS files.	Any configured COS name from the pop-up list.	First configured COS name found in the SFS file.
	Advice: The hierarchy associated with this COS should have disk at its top storage level.		
Table Entries	The number of entries in a hash table used to hold cached attributes for HPSS bitfiles, directories, and symbolic links.	Any positive 32-bit integer value.	97
	Advice: This value should be set to a prime number greater than or equal to the number of HPSS objects used through NFS at any one time.		
LRU Maximum Length	The maximum number of HPSS objects to cache.	Any positive 32-bit integer value.	300
	Advice: This value should be equal to the number of HPSS objects in use through NFS for a period specified by the hold time. Generally, this value should not be greater than three times the number of table entries.		

Table 5-22 NFS Daemon Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Hold Time	The interval, in seconds, for which cached attributes remain valid.	Any positive 32-bit integer value.	99
	Advice: Smaller hold times reduce the effectiveness of the cache; larger hold times result in attributes not being updated as often.		
Directory Size	The size of the cache for directory name entries.	Any positive 32-bit integer value.	1024
	Advice: This value should be based on average file size (fsize) and the average number of files (fnum). The value = 4(44 + fsize) * fnum.		
Root User ID	The HPSS root identifier.	Valid root's UID.	0
	Advice: This field should be equivalent to the root identifier in the NS configuration.		
Disk and Memory Data Cache. The following thirteen fields are used by the NFS Daemon to cache bitfile data.			
Buffer Size	The size of a single data cache entry.	Any positive 32-bit integer value.	500 KB
	Advice: This value is the amount of data read from and written to HPSS by the data cache layer at a time. Is is recommended that this value be a multiple of 8 KB and, if possible, a power of 2. When multiplied by the number of Memory Buffers, this value is the amount of memory that will be used by the data cache layer. When multiplied by the number of Cache Entries, it represents the amount of disk space used by the Cache file.		
Cache Entries	The number of data cache entries stored on local disk.	Any positive 32-bit integer value.	100
	Advice: When multiplied by the Buffer Size, this value represents the amount of disk used to hold the data cache entries locally in the Cache file.		
Memory Buffers	The number of buffers to allocate in memory.	Any positive 32-bit integer value.	10
	Advice: When multiplied by the Buffer Size, this value represents the RAM used to hold cache entries while they are being worked on. This value must be at least 3.		
Cache File (Unix)	The name of a local UNIX disk file that contains the cached entries.	Any valid, fully qualified UNIX file name.	/var/hpss/nfs/cachefile.nfs2
	Advice: The size of this file can be determined by multiplying Buffer Size by Cache Entries		

Table 5-22 NFS Daemon Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
CheckPoint File (Unix)	The name of a local UNIX disk file that contains the status information for the cached entries stored in the Cache file. Its size is related to the number of the cached entries, but it is much smaller than the Cache file.	Any valid, fully qualified UNIX file name.	/var/hpss/nfs/checkpoint.nfs2
Cleanup Threads	The number of threads dedicated to writing dirty cache entries back to HPSS.	Any positive 32-bit integer value.	1
	Advice: This value should be small because extra, temporary cleanup threads are spawned as needed. Each non-temporary cleanup thread wakes up every Thread Interval seconds to look for dirty entries.		
Thread Interval	An interval, in seconds, that specifies how often the cleanup threads wake up to look for dirty entries.	Any positive 32-bit integer value.	30 seconds
	Advice: This value should not be too small because extra cleanup threads are spawned as needed when the cache starts to fill up.		
Dirty Threshold	The percentage of disk cache entries that must be dirty before extra cleanup threads are spawned.	Any integer value between 1 and 100.	80 percent
Touch Interval	The interval, in seconds, that a disk cache entry must not have been accessed in order for it to be considered a candidate to be written back to HPSS by a cleanup thread.	Any positive 32-bit integer value.	30 seconds
	Advice: This value is ignored when the Dirty Threshold is exceeded. It is recommended that this value be greater than the number of seconds the NFS client cache holds on to dirty data.		
Touch Weight	The importance of writing back to HPSS a dirty entry that has not been accessed too recently.	Any positive 32-bit integer value.	3
	Advice: The Touch Weight should be higher than the Dirty Weight to avoid locking out access to cached data that has not been flushed to HPSS. A three to one ratio is recommended (3 for Touch Weight, 1 for Dirty Weight).		

Table 5-22 NFS Daemon Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Dirty Weight	The importance of writing back to HPSS a dirty entry that has not been dirty for a long period of time.	Any positive 32-bit integer value.	1
	Advice: The Dirty Weight should be lower than the Touch Weight to avoid locking out access to cached data that has not been flushed to HPSS. A three to one ratio is recommended (3 for Touch Weight, 1 for Dirty Weight).		
Recover Cached Data	A flag that indicates whether the data cache layer will be forced to look for dirty cache entries in the CheckPoint File. When ON, the data cache layer looks for dirty entries. When OFF, it ignores them and resets the checkpoint file.	ON, OFF	ON
	Advice: This flag should be turned OFF temporarily only when there is a problem with the recovery process that inhibits the NFS Daemon from starting up because any dirty entries will be lost.		
Bypass Cache File	A flag that indicates whether the Cache file will be used. When ON, the Cache file will not be used. All transfers will go directly to HPSS.	ON, OFF	OFF

5.6.13 Configure the Mount Daemon Specific Information

The NFS Mount Daemon specific configuration entry can be created using the Mount Daemon Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 5-3.

To add a new specific configuration, select the Mount Daemon entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The Mount Daemon Configuration window will be displayed as shown in Figure 5-32 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Mount Daemon entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Mount Daemon Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the Mount Daemon entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Mount Daemon Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

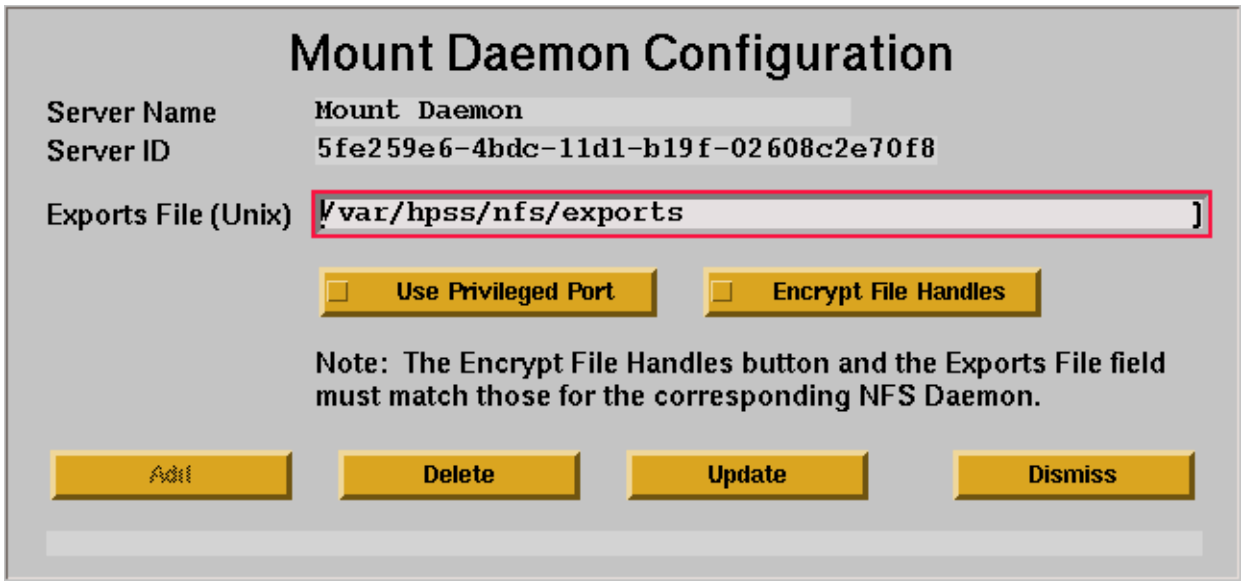


Figure 5-32 Mount Daemon Configuration Window

Mount Daemon Configuration Variables

Table 5-23 lists the fields on the Mount Daemon Configuration window and provides specific recommendations for configuring the Mount Daemon for use by HPSS.

Table 5-23 Mount Daemon Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the Mount Daemon. This name is copied over from the Mount Daemon general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected Mount Daemon descriptive name.
Server ID	The UUID of the Mount Daemon. This ID is copied over from the Mount Daemon general configuration entry.	The UUID of the Mount Daemon. This field cannot be modified. It is displayed for reference only.	Assigned by SSM when the Mount Daemon general configuration entry is created.
Exports File (Unix) .	The name of a UNIX file containing the HPSS directory / file exported for NFS access.	Valid, fully qualified UNIX file name.	/ var / hpss / nfs / exports
	Advice: This field should be set to the same value as the Exports File field for the NFS server.		
Use Privileged Port	A flag that indicates whether the NFS clients will use a privileged port. If set, the NFS clients must use port numbers less than 1024.	ON, OFF	OFF
Encrypt File Handles	A flag that indicates whether incoming and outgoing file handles will contain an encrypted checksum.	ON, OFF	OFF
	Advice: This field should be set to the same value as the Encrypt File Handles field for the NFS server. If it is not, NFS client systems will get ESTALE errors.		

5.6.14 Configure the Non-DCE Client Gateway Specific Information

The Non-DCE Client Gateway specific configuration entry can be created using the Non-DCE Client Gateway Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 5-3.

To add a new specific configuration, select the Non-DCE Client Gateway entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The Non-DCE Client Gateway Configuration window will be displayed as shown in Figure 5-33 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Non-DCE Client Gateway entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Non-DCE Client Gateway Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the Non-DCE Client Gateway entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Non-DCE Client Gateway Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

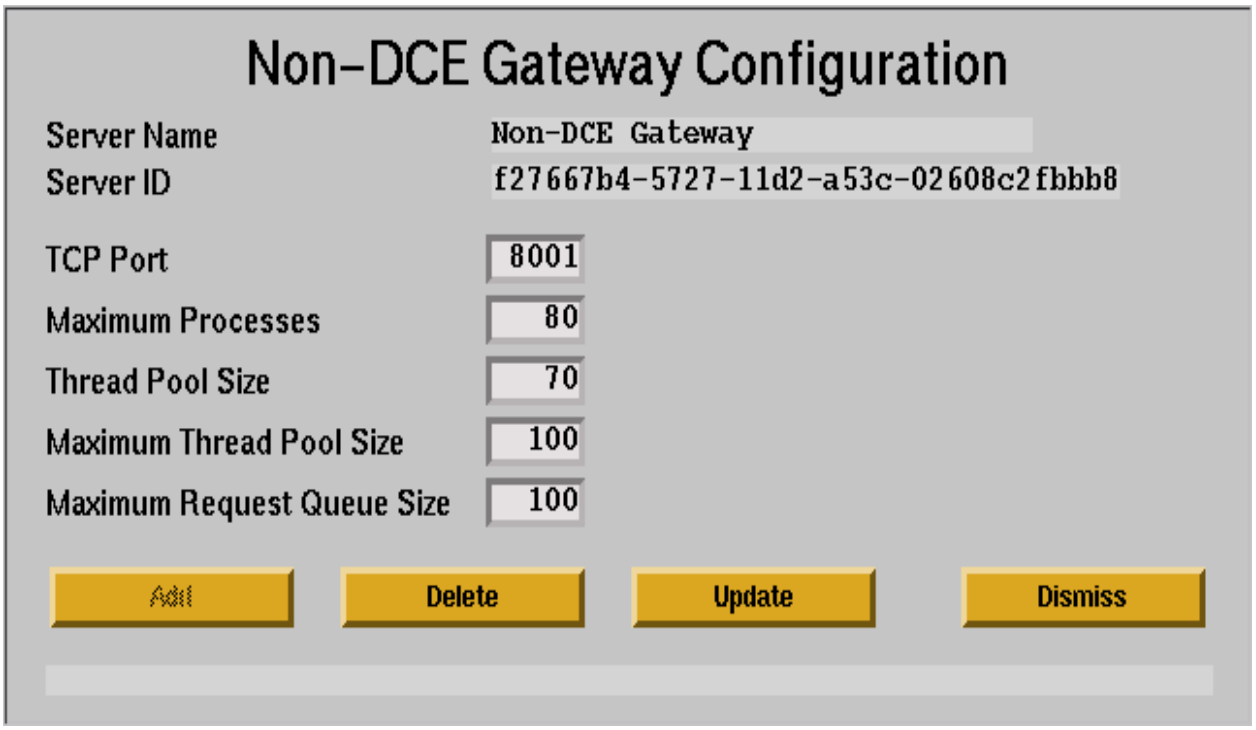


Figure 5-33 Non-DCE Client Gateway Configuration Window

Non-DCE Client Gateway Configuration Variables

Table 5-24 lists the fields on the Non-DCE Client Gateway Configuration window and provides specific recommendations for configuring the Non-DCE Client Gateway for use by HPSS.

Table 5-24 Non-DCE Client Gateway Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the Non-DCE Client Gateway. This name is copied over from the Non-DCE Client Gateway general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected Non-DCE Client Gateway descriptive name.
Server ID	The UUID of the Non-DCE Client Gateway. This ID is copied over from the Non-DCE Client Gateway general configuration entry.	The UUID of the Non-DCE Client Gateway. This field cannot be modified. It is displayed for reference only.	Assigned by SSM when the Non-DCE Client Gateway general configuration entry is created.
TCP Port	Default port number to be used by the Non-DCE Client Gateway's listener process.	This number must be unique among all Non-DCE Client Gateways running on a single host	8001
Maximum Processes	The maximum number of request processes that can be active at any point in time. Effectively the maximum number of clients that can be connected to the Non-DCE Client Gateway at a time.	Any positive 32-bit integer value.	50
Thread Pool Size	The size of the request thread pool for each request process. If the number of simultaneous requests equals this number, the thread pool will be expanded up to its maximum defined value to handle additional requests. When request activity is then reduced, the size of the thread pool with be reduced back to this value.	Any positive 32-bit integer value.	10

Table 5-24 Non-DCE Client Gateway Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Maximum Thread Pool Size	The maximum number of request threads per process active at any one time. If the number of simultaneous requests equals this number, any additional requests will be queued for processing by the next available request thread.	Any positive 32-bit integer value. This value must be at least as large as the value given for Thread Pool Size.	50
Maximum Request Queue Size	The maximum number of requests to queue until a request thread becomes available. If this queue fills, no more requests will be processed for this particular Non-DCE client until there is more room in the queue.	Any positive 32-bit integer value.	20

5.7 Configure MVR Devices and PVL Drives

The MVR Device and PVL Drive objects refer to the same physical drive, but are maintained in two separate SFS files based on what information is required by the PVL for drive management and what information is required by the Mover for device management. The two SFS files are managed by SSM from the same configuration window so that they can be synchronized with one another.

All tape devices that will be used for HPSS data must be set to handle variable block sizes (to allow for the ANSI standard 80-byte volume label and file section headers).

To set the devices to use variable blocks on an AIX platform, use either the **chdev** command (substituting the appropriate device name for **rmt0**):

chdev -l rmt0 -a block_size=0

or **smitty**:

smitty tape
<select “Change / Show Characteristics of a Tape Drive”>
<select the appropriate tape device>
<change “BLOCK size (0=variable length)” to “0”>
<enter>

All locally attached magnetic disk devices (e.g., SCSI, SSA) should be configured using the pathname of the raw device (i.e., character special file).

The configuration of the storage devices (and subsequently the Movers that control them) can have a large impact on the performance of the system because of constraints imposed by a number of factors (e.g., device channel bandwidth, network bandwidth, processor power).

Before proceeding with device and drive configuration associated with StorageTek, or IBM robotics, refer to Appendix K (StorageTek PVR Information), or Appendix J (IBM 3494/3495 Information) as appropriate. These appendices provide additional vendor-specific advice on robot and drive configuration.

A Device and Drive configuration entry can be created, updated and deleted only if the PVL and the associated Mover are not running.

The Device and Drive configuration entry can be created using the Mover Device and PVL Drive Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 5-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Devices and Drives** option. The HPSS Devices and Drives window will be displayed as shown in Figure 5-34.

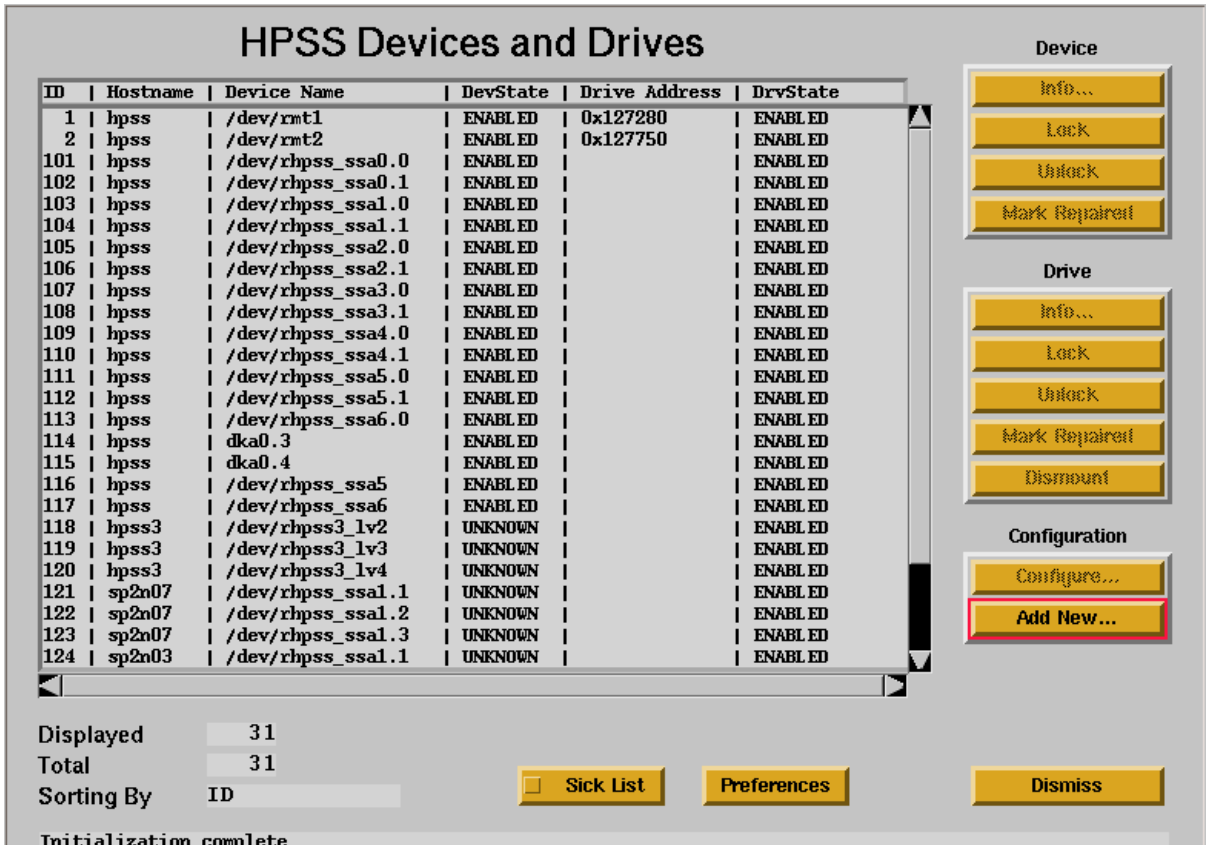


Figure 5-34 HPSS Devices and Drives Window

To configure a new device and drive, click on the **Add New...** button on the HPSS Devices and Drives window. The Mover Device and PVL Drive Configuration window will be displayed as shown in Figure 5-35 with default values for a new tape device / drive. If a disk device / drive is

desired, click the **Disk** button to display the default disk data (as shown in Figure 5-35) before modifying any other fields. If the default data is not desired, change the field with the desired value. Click on the **Add** button to create the configuration entry.

To update an existing device and drive, select the desired device and drive entry on the HPSS Devices and Drives window and click on the **Configure...** button. The Mover Device and PVL Drive Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing device and drive, select the desired device and drive on the HPSS Devices and Drives window and click on the **Configure...** button. The Mover Device and PVL Drive Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

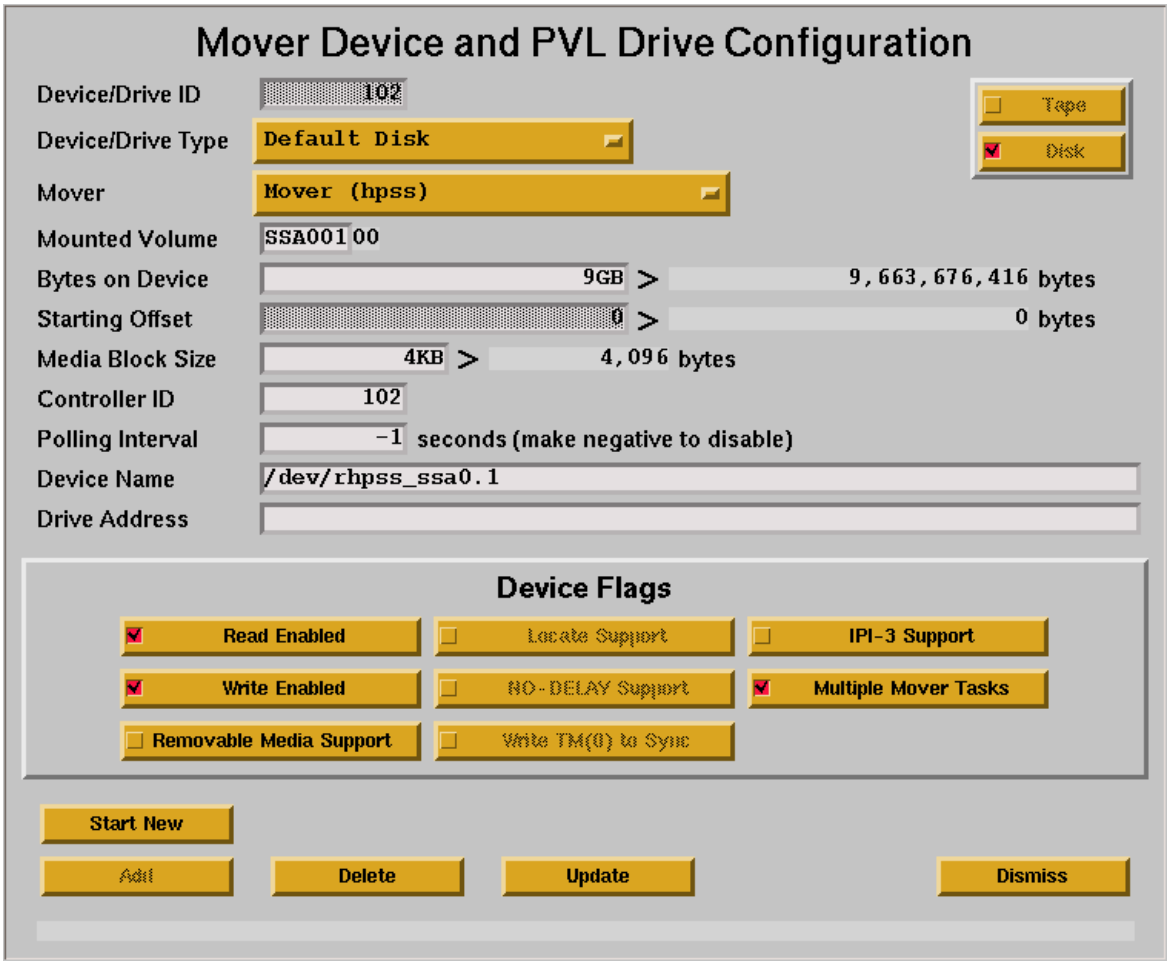


Figure 5-35 Disk Mover Device and PVL Drive Configuration Window

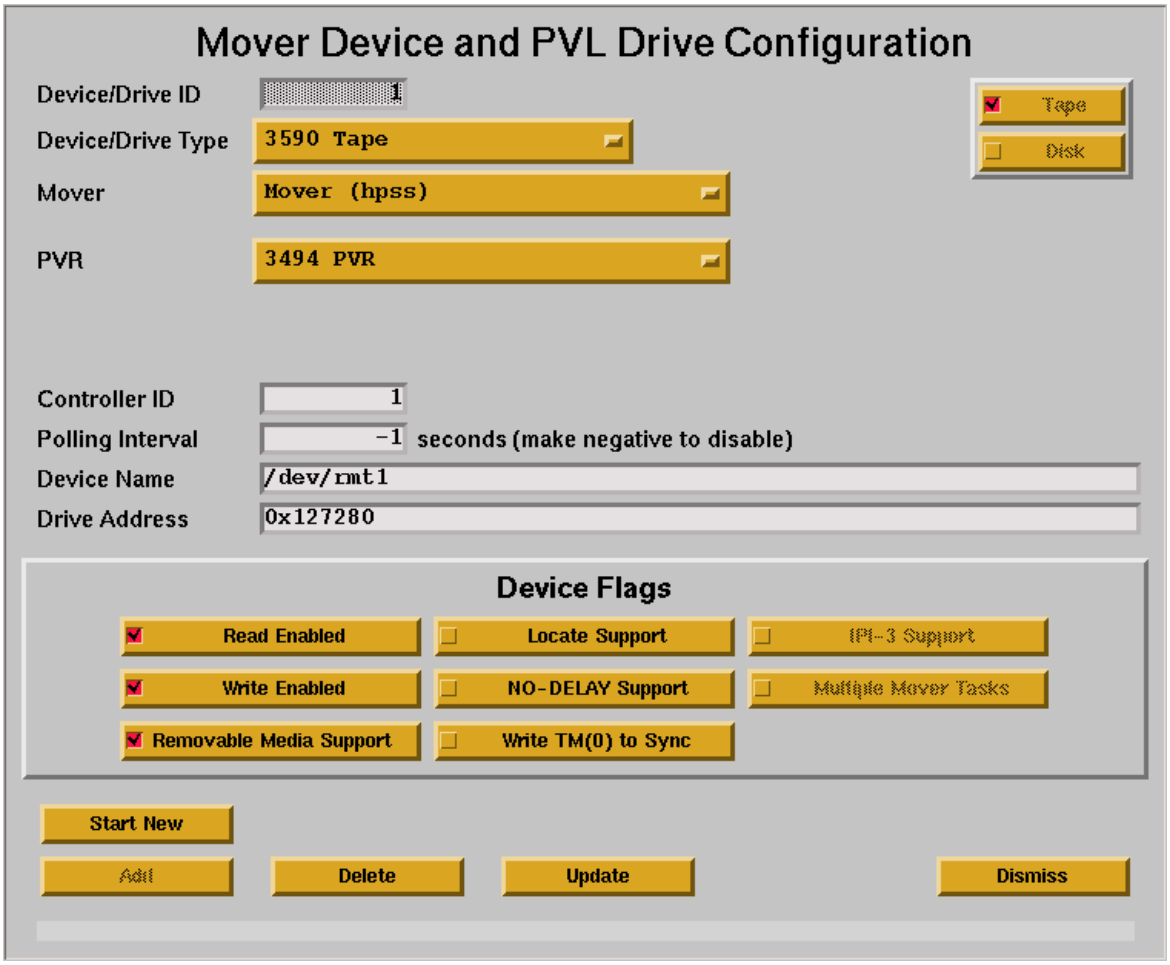


Figure 5-36 Tape Mover Device and PVL Drive Configuration Window

Device and Drive Configuration Variables

Table 5-25 lists the fields on the Mover Device and PVL Drive Configuration window.

Table 5-25 Device/Drive Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Device/Drive ID	The unique, numeric ID associated with this device / drive.	Any non-zero, positive 32-bit integer value.	Highest ID found in the SFS file plus 1.

Table 5-25 Device/Drive Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Device /Drive Type	The type of device over which data will move.	Any valid type from the pop-up list.	Default Tape or Default Disk, based on Tape / Disk toggle buttons.
Mover	The name of the MVR that controls the device.	Any configured MVR name from the pop-up list.	First configured MVR name found in the SFS file.
	Advice: There is a maximum of 64 devices that can be configured for a Mover.		
PVR	The name of the PVR that handles the removable media operations for the drive.	Any configured PVR name from the pop-up list.	First configured PVR name found in the SFS file.
	Advice: This field is meaningful for tape devices only.		
Mounted Volume	The 8-character name of the volume mounted on the disk drive, if any.	Valid 8-character name assigned to the disk volume.	None
	Advice: This value is meaningful for disk devices only.		
Bytes on Device	The size of device in bytes.	Any positive 64-bit integer value up to the system-imposed maximum device size.	None
	Advice: This value is used for disk devices only. If it is modified after the mounted disk volume has been imported into HPSS, the volume must be re-imported into HPSS. Note that the **PV Size** from Section 5.5 for the storage class must be less than or equal to the value of **Bytes on Device**. If the **Starting Offset** is non-zero, then the **Bytes on Device** value cannot be greater than the actual size of the underlying device less the **Starting Offset** value.		
Starting Offset	The offset in bytes from the beginning of the disk logical volume at which the Mover will begin using the volume. The space preceding the offset will not be used by HPSS.	Zero, or a positive integer that is a multiple of the **Media Block Size**.	Zero
	Advice: This value is used for disk devices only.		

Table 5-25 Device/Drive Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Media Block Size	The block size for the device.	Any positive 32-bit integer value.	None
	Advice: This value is used for disk devices only, and must be a multiple of the underlying disk block size.		
Controller ID	An indication of the adapter/bus that is in the data path to the device.	Any positive 32-bit integer value.	Set to Device/Drive ID
	Advice: This field is used to attempt to mount separate volumes of a set of striped media on separate controllers, when possible, to prevent contention on the data path. It is recommended that the drives that are on the same adapter/bus have the same Controller ID. For drives in 3494 and 3495 robots, the Controller ID must be a valid ID. The valid IDs can be found in the Affinity list for any cartridge in the robot. Use the "**mtlib -l <device name> -qV -V<volume name**>" to obtain the Affinity list for a cartridge.		
Polling Interval	The number of seconds to wait between polling requests performed by the PVL to determine if any media is present in the drive.	Any positive 32-bit integer value for polling or any negative 32-bit integer value to disable polling.	60 seconds for tape; -1 for disk, robotic drives
	Advice: This field should be set to negative to disable polling for drives that do not support removable media. The value for drives located within robotic libraries should probably be set higher than for manually mounted drives or set to a negative value (preferred). We recommend that this field be set to 15 seconds for operator-mounted tape drives, -1 for tape robots, and -1 for disk drives.		
Device Name	The name by which the MVR can access the device.	Any valid UNIX path name of a device file.	None
	Advice: This name is usually the path name of a device special file such as **/dev/rmt0/** For locally attached disk devices, the pathname should refer the raw/character special file (e.g., **/dev/rhpss_disk1**). For HiPPI-attached disk arrays, the name should refer to the partition name configured in the IPI-3 configuration file (refer to Section 5.8.5 for details). For AIX systems, SCSI attached tape drives are typically refered to by pathnames of the form /dev/rmtX, where X begins at zero and is incremented for each tape drive detected. For IRIX systems, SCSI attached tape drives are typically refered to by pathnames of the form /dev/rmt/tpsXdYns, where X is the SCSI controller number and Y the SCSI ID of the drive. Note that for Ampex DST drives, the tpsXdYnrns name should be used (indicating that the driver should not attempt to rewind the drive upon close). For Solaris systems, SCSI attached tape drives are typically refered to by pathnames of the form /dev/rmt/Xc, where X begins at zero and is incremented for each tape drive detected (the 'c' indicates that compression is enabled). In particular note that the device that contains a 'b' in the name should NOT be used, as this will change the behavior of the drive which will cause the HPSS Mover to fail.		

Table 5-25 Device/Drive Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Drive Address	The name/address by which the PVR can access the drive.	Valid drive address (see Advice below).	None
	Advice: *For StorageTek robots:* Drive Address configuration entries correspond to the ACS,Unit,Panel,Drive Number used by ACSLS to identify drives. For example, the first drive in a typical configuration has Drive Address **0,0,10,0** (see Appendix K for information about configuration of StorageTek drives). *For IBM robots:* The Drive Address configuration entries correspond to the hexadecimal Library device number of the drive. Determine the Library device number by running the command "**/usr/lpp/hpss/bin/GetESANumbers /dev/rmtX**" for each tape drive in the robot. *For operator mounted drives:* For manually mounted drives, the drive address string is displayed to operators when an explicit drive is selected for mounting. The drive address should therefore be a string that easily communicates to an operator the drive in question (i.e., a name matching the label on the exterior of the drive). *For AML robots:* Leave the Drive Address field blank for AML robots.		
Device Flags. The following fields are the device flags used by the MVR. Refer to Table for more information on the recommended settings for tape devices.			
Read Enabled	An indication of whether the device is available for reading.	ON, OFF	ON
Write Enabled	An indication of whether the device is available for writing.	ON, OFF	ON
Removable Media Support	An indication of whether the device supports removable media.	ON, OFF	ON for tape, OFF for disk
Locate Support	An indication of whether the device supports a high speed (absolute) positioning operation.	ON, OFF	ON
	Advice: This option is supported for 3480, 3490, 3490E, 3590, Timberline, Redwood, 9840, and Ampex DST devices, provided the HPSS supported device driver is used in accessing the device.		

Table 5-25 Device/Drive Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
NO-DELAY Support	An indication of whether the device supports opening the device with no delay flag set, while allowing tape I/O operation after the open.	ON, OFF	ON
	Advice: On some tape devices, this will allow for a quicker polling operation when no tape is presently loaded in the device. This field is meaningful for tape devices only. This option is not supported for the BMUX attached 3480, 3490, 3490E and 3590 devices.		
Write TM(0) to Sync	An indication of whether the device supports issuing a write tapemark request with a zero count to flush data written previously to the tape media.	ON, OFF	OFF
	Advice: On some devices, this may provide a higher performance method to ensure that data has been safely written to the media. This option is not used for the 3480, 3490, 3490E, 3590, Timberline, and Redwood devices, provided the HPSS supported device driver is used in accessing the device. Note that for Ampex DST-312 this field should be set to "ON".		
IPI-3 Support	An indication of whether the device supports the HPSS IPI-3 third-party transfer mechanism.	ON, OFF	OFF
	Advice: This is supported on the Maximum Strategy and IBM 9570 HiPPI attached disk array devices only. This value is meaningful for the disk device only.		
Multiple Mover Tasks	An indication of whether the MVR should allow multiple MVR tasks to simultaneously access the device.	ON, OFF	ON for disk, OFF for tape
	Advice: This value is meaningful for the disk device only.		

Recommended Settings for Tape Devices

Table 5-26 lists the recommended Device Flag settings for tape devices:

Table 5-26 Recommended Settings for Tape Devices

Device Type	Mover TCP Executable	NO-DELAY Support	Locate Support	Write TM(0) to Sync
Redwood	hpss_mvr_omi hpss_mvr_ipi_omi	ON	ON	OFF
Timberline	hpss_mvr_omi hpss_mvr_ipi_omi	ON	ON	OFF
9840	hpss_mvr_omi hpss_mvr_ipi_omi	ON	ON	OFF
3590, 3590E	hpss_mvr_ssd hpss_mvr_ipi_ssd	ON	ON	OFF
3490E (SCSI)	hpss_mvr_ssd hpss_mvr_ipi_ssd	ON	ON	OFF
3490E (BMUX)	hpss_mvr_bmux hpss_mvr_ipi_bmux	OFF	ON	OFF
3480 (BMUX)	hpss_mvr_bmux hpss_mvr_ipi_bmux	OFF	ON	OFF
DST-312, DST-314	hpss_mvr_dd2 hpss_mvr_ipi_dd2	OFF	ON	ON

5.8 Configure UNIX Environments

Additional UNIX configuration must be performed for the following components:

- Client API
- Non-DCE Client API
- FTP Daemon
- NFS Daemon
- MVR (if IPI-3 is supported)
- MVR (if non-DCE mode is utilized)
- MPI-IO API
- AML PVR
- Network

5.8.1 Client API Configuration

The following environment variables can be used to define the Client API configuration:

The **HPSS_LS_NAME** defines the CDS name of the Location Server RPC Group entry for the HPSS system that the Client API will attempt to contact. The default is **/.:/hpss/ls/group**.

The **HPSS_MAX_CONN** defines the number of connections that are supported by the Client API within a single client process. The default is zero which is equal to the default supported by the HPSS connection management software - currently 150.

The **HPSS_KTAB_PATH** defines the name of the file containing the DCE security keys necessary for successfully initializing the Client API. The default is **/krb5/hpssclient.keytab**.

The **HPSS_HOSTNAME** environment variable is used to specify the hostname to be used for TCP/IP listen ports created by the Client API. The default value is the default hostname of the machine on which the Client API is running. This value can have a significant impact on data transfer performance for data transfers that are handled by the Client API (i.e., those that use the **hpss_Read** and **hpss_Write** interfaces).

The **HPSS_TCP_WRITESIZE** environment variable is used to specify the amount of data to be written with each individual request to write data to a network connection during a data transfer. For some networks, writing less than the entire size of the client buffer has resulted in improved throughput. This environment variable may not affect the actual value used, based on the contents of the HPSS network options file - see Section 5.8.10 for further details.

The **HPSS_TRANSFER_TYPE** environment variable is used to specify the data transport mechanism to be used for data transfers handled by the Client API. Valid values are either **TCP** for TCP/IP transfers or **IPI3** for IPI-3 transfers over HiPPI. The default value is **TCP**. Note that the Client API must have been built with IPI-3 support for transfer to be performed via IPI-3 over HiPPI.

The **HPSS_PRINCIPAL** environment variable is used to specify the DCE principal to be used when initializing the HPSS security services. The default value is **hpss_client_api**. This variable is primarily intended for use by HPSS servers that use the Client API.

The **HPSS_SERVER_NAME** environment variable is used to specify the server name to be used when initializing the HPSS security services. The default value is **/.:/hpss/client**. This variable is primarily intended for use by HPSS servers that use the Client API.

The **HPSS_DESC_NAME** environment variable is used to control the descriptive name used in HPSS log messages if the logging feature of the Client API is enabled. The default value is “Client Application”.

The Client API, if compiled with debugging enabled, uses two environment variables to control printing debug information. **HPSS_DEBUG**, if set to a non-zero value, will enable debug messages. By default, these messages will go to the standard output stream. If **HPSS_DEBUGPATH** is set, however, these messages will be directed to the file indicated by this environment variable. Two special cases for the debug path exist: **stdout** and **stderr**, which will use the standard output or standard error I/O streams, respectively.

The **HPSS_NUMRETRIES** environment variable is used to control the number of retries to attempt when an operation fails. Currently this class of operation includes library initialization and

communications failures. A value of zero indicates that no retries are to be performed, and value of "-1" indicates that the operation will be retried until successful. The default value is 4

The **HPSS_BUSY_RETRIES** environment variable is used to control the number of retries to be performed when a request fails because the Bitfile Server does not currently have an available thread to handle that request. A value of zero indicates that no retries are to be performed, and a value of "-1" indicates that retries should be attempted until either the request succeeds or fails for another reason. The default value is 3.

The **HPSS_BUSY_DELAY** environment variable is used to control the number of seconds to delay between retry attempts. Note that this value is used both for retrying initialization operations (see **HPSS_NUMRETRIES**) and Bitfile Server requests (See **HPSS_BUSY_RETRIES**). The default value is 15.

The **HPSS_RETRY_STAGE_INP** environment variables is used to control whether retries are attempted on attempts to open files in a Class of Service that is configured for background staging on open. A non-zero value indicates that opens which would return **-EINPROGRESS** to indicate that the file is being staged will be retried (using the same control mechanisms described in the previous paragraph), while a value of zero indicates that the **-EINPROGRESS** return code will be returned to the client. The default value is non-zero.

The **HPSS_REUSE_CONNECTIONS** environment variable is used to control whether TCP/IP connections are to be left open as long as a file is open or are to be closed after each read or write request. A non-zero value will cause connections to remain open, while a value of zero will cause connections to be close. The default value is zero.

The **HPSS_USE_PORT_RANGE** environment variable is used to control whether the HPSS Mover(s) should use the configured port range when making TCP/IP connections for read and write requests. A non-zero value will cause the Mover(s) to use the port range, while a value of zero will cause the Mover(s) to allow the operating system to select the port number. The default value is zero.

The **HPSS_TOTAL_DELAY** environment variable is used to control the number of seconds to continue retrying requests. A value of zero indicates that no there is no time limit. The default value is 0.

The **HPSS_REGISTRY_SITE_NAME** environment variable is used to specify the name of the security registry used when inserting security information into connection binding handles. This is only needed when the client must support DFS in a cross-cell environment. The default registry is "**/.../dce.clearlake.ibm.com**".

The **HPSS_DMAP_WRITE_UPDATES** environment variable is used control the frequency of cache invalidates that are issued to the DMAPI file system while writing to a file that is mirror in HPSS. The default value is 20.

5.8.2 Non-DCE Client API Configuration

The following environment variables can be used to define the Non-DCE Client API configuration. Do not confuse references to the Client API with references to the Non-DCE Client API. When reference is made to the Client API, the document is referring to the actual Client API calls made by the Non-DCE Client Gateway. When reference is made to the Non-DCE Client API, the document is referring to calls made directly by the Non-DCE client application.

The **HPSS_NDCG_NAME** environment variable is used to specify the server name to be used when initializing the HPSS security services. The default value is **/.:/hpss/client**.

The **HPSS_NDCG_TCP_PORT** defines the default port location for the listener process of the Non-DCE Client Gateway with which the Non-DCE Client API will communicate. This value can be overridden by appropriate entries in the **HPSS_NDCG_SERVERS** environment variable. The default value is **8001**.

The **HPSS_NDCG_SERVERS** defines the name of the server on which the Non-DCE Client Gateway resides. Multiple servers may be separated by a colon (:). If multiple servers are specified, the NDAPI will randomly choose a listed server every time it establishes a new gateway connection. This way, multiple NDCGs may be configured for load-balancing. It is possible to explicitly set the TCP port on a per server basis by following the server name with a forward slash (/) and a port number. For example, a string **"hpss/8002:pluto"** would define two Non-DCE Client Gateways. One (hpss) uses an explicit port number, and the other (pluto) uses the value from **HPSS_NDCG_TCP_PORT**.

The **HPSS_LOGGING_PORT** defines the port number of the Log Client on the host running the connecting Non-DCE Client Gateway. The default value is **8101**.

The **HPSS_LOGGING_TYPE** defines the types of messages to log. It consists of a list of log types, separated by colons (:). For example **"CS_ALARM:CS_STATUS"** would log alarm and status messages. Valid log types are: **CS_ALARM**, **CS_EVENT**, **CS_REQUEST**, **CS_SECURITY**, **CS_ACCOUNTING**, **CS_DEBUG**, **CS_TRACE**, and **CS_STATUS**. The default value is **"CS_ALARM: CS_EVENT:CS_REQUEST:CS_SECURITY"**.

The **HPSS_MAX_CONN** defines the number of connections that are supported by the Client API within a single client process. The default is zero which is equal to the default supported by the HPSS connection management software - currently 150.

The **HPSS_KTAB_PATH** defines the name of the file containing the DCE security keys necessary for successfully initializing the Client API. The default is **/krb5/hpssclient.keytab**.

The **HPSS_HOSTNAME** environment variable is used to specify the hostname to be used for TCP/IP listen ports created by the Client API. The default value is the default hostname of the machine on which the Client API is running. This value can have a significant impact on data transfer performance for data transfers that are handled by the Client API (i.e., those that use the **hpss_Read** and **hpss_Write** interfaces).

The **HPSS_TCP_WRITESIZE** environment variable is used to specify the amount of data to be written with each individual request to write data to a network connection during a data transfer. For some networks, writing less than the entire size of the client buffer has resulted in improved throughput. This environment variable may not affect the actual value used, based on the contents of the HPSS network options file - see Section 5.8.10 for further details.

The **HPSS_TRANSFER_TYPE** environment variable is used to specify the data transport mechanism to be used for data transfers handled by the Client API. Valid values are either **TCP** for TCP/IP transfers or **IPI3** for IPI-3 transfers over HiPPI. The default value is **TCP**. Note that the Client API must have been built with IPI-3 support for transfer to be performed via IPI-3 over HiPPI.

The **HPSS_PRINCIPAL** environment variable is used to specify the DCE principal to be used when initializing the HPSS security services. The default value is **hpss_client_api**.

The **HPSS_DESC_NAME** environment variable is used to control the descriptive name used in HPSS log messages if the logging feature of the Client API is enabled. The default value is "Client Application".

The Client API, if compiled with debugging enabled, uses two environment variables to control printing debug information. **HPSS_DEBUG**, if set to a non-zero value, will enable debug messages. By default, these messages will go to the standard output stream. If **HPSS_DEBUGPATH** is set, however, these messages will be directed to the file indicated by this environment variable. Two special cases for the debug path exist: **stdout** and **stderr**, which will use the standard output or standard error I/O streams, respectively.

The **HPSS_NUMRETRIES** environment variable is used to control the number of retries to attempt when an operation fails. Currently this class of operation includes library initialization and communications failures. A value of zero indicates that no retries are to be performed, and value of "-1" indicates that the operation will be retried until successful. The default value is 4

The **HPSS_BUSY_RETRIES** environment variable is used to control the number of retries to be performed when a request fails because the Bitfile Server does not currently have an available thread to handle that request. A value of zero indicates that no retries are to be performed, and a value of "-1" indicates that retries should be attempted until either the request succeeds or fails for another reason. The default value is 3.

The **HPSS_BUSY_DELAY** environment variable is used to control the number of seconds to delay between retry attempts. Note that this value is used both for retrying initialization operations (see **HPSS_NUMRETRIES**) and Bitfile Server requests (See **HPSS_BUSY_RETRIES**). The default value is 15.

The **HPSS_RETRY_STAGE_INP** environment variables is used to control whether retries are attempted on attempts to open files in a Class of Service that is configured for background staging on open. A non-zero value indicates that opens which would return **-EINPROGRESS** to indicate that the file is being staged will be retried (using the same control mechanisms described in the previous paragraph), while a value of zero indicates that the **-EINPROGRESS** return code will be returned to the client. The default value is non-zero.

The **HPSS_REUSE_CONNECTIONS** environment variable is used to control whether TCP/IP connections are to be left open as long as a file is open or are to be closed after each read or write request. A non-zero value will cause connections to remain open, while a value of zero will cause connections to be close. The default value is zero.

The **HPSS_USE_PORT_RANGE** environment variable is used to control whether the HPSS Mover(s) should use the configured port range when making TCP/IP connections for read and write requests. A non-zero value will cause the Mover(s) to use the port range, while a value of zero will cause the Mover(s) to allow the operating system to select the port number. The default value is zero.

The **HPSS_TOTAL_DELAY** environment variable is used to control the number of seconds to continue retrying requests. A value of zero indicates that no there is no time limit. The default value is 0.

The **HPSS_REGISTRY_SITE_NAME** environment variable is used to specify the name of the security registry used when inserting security information into connection binding handles. This is only needed when the client must support DFS in a cross-cell environment. The default registry is "**/.../dce.clearlake.ibm.com**".

The **HPSS_DMAP_WRITE_UPDATES** environment variable is used control the frequency of cache invalidates that are issued to the DMAPI file system while writing to a file that is mirror in HPSS. The default value is 20.

5.8.3 FTP Daemon Configuration

There are four steps that must be performed prior to using the FTP user interface to transfer HPSS data:

1. Verifying the FTP Initial Configuration
2. Configuring the FTP Daemon Syslog
3. Defining the FTP Access
4. Creating FTP Users

These steps are described in more detail in the paragraphs that follow.

Step 1. Verifying the FTP Configuration

During the HPSS infrastructure configuration phase, the following files were modified by the **mkhpss** script to add the required FTP configuration:

- **/etc/services**—Verify that the **hpssftp** and **hpssftp-data** entries were added correctly. If possible, consider moving the ftp and ftp-data components to some other ports (4020/4021) and assign the hpssftp and **hpssftp-data** to ports 20/21. Reinitialize **inetd** after making these changes: "**refresh –s inetd**" or determine the pid for inetd and "**kill –1 pid**".

- **/etc/inetd.conf—**Verify that the **hpssftp** entry was added correctly. The flags described below can be used in modifying this entry. Note that the flags are preceded by a dash ("**-")** and that single character flags can be grouped in one argument (e.g., -adhvLSUX). If possible, use a "TCP Wrapper" application for initiating the HPSS FTP Daemon. This enhances "on-the-fly" changes to the startup of the HPSS FTP Daemon. Additionally, this provides for enhanced security. Several "TCP Wrapper" applications are available in the Public domain.

HPSS Parallel FTP Daemon options:

The only options which accept additional arguments are the –p, -s, -D, and –F options. The –p requires a port number (decimal value) while the other D and F options expect strings < **MAXPATHLEN** characters in length. The syntax on the –s option is explained below and should be handled carefully or indeterminate errors may occur! No space is allowed between the option and the value. **NOTE: that using long strings MAY cause severe problems for inetd!** Each of these four options MUST be preceded by a space and a "-". In general, it is recommended that these four options NOT be used!

All the other options should be specified by a single "-" at the start and the concatenation of options (e.g., "-abdddhtuvzARSTUX" – Not necessarily in that order)

NOTE the three consecutive "d"s imply to "**syslog"** as much debug information as possible. See the explanation below for the "d" option.

Option	Description
a	Mandate Authentication with DCE/Kerberos Credentials - Disable{Username}/{Password} authentication. This also disables the user command.
b	Read the **hpss_option BUFSZ <value>** option from the {**ftpaccess**} file to obtain the path/name of the size specification to be used. Refer to documentation on the {**ftpaccess**} file for more details.
d	Turn on Debugging - Sets environment variables: **HPSS_DEBUG**=1 and **HPSS_DEBUGPATH**=**FTPBaseDir/adm/hpss_ftpd.log**. See option -D for details on **FTPBaseDir**. This option may be repeated up to 3 times. Each specification increases the detail of debug information sent to the syslog. **NOTE**: the information in the syslog is subject to the specifications in **/etc/syslog.conf**. For the most information, specify ***.debug** in /**etc/syslog.conf**.
h	Search the **ftpaccess** file for the **HOST** specifier - Defines the network interface to be used for data transferred between the PFTPD and the Movers when performing non-parallel transfers. **hpss_option HOST <interface name>** option sets the **HPSS_HOSTNAME** environment variable for the Client API. Refer to documentation on the **ftpaccess** file for details.
p#	Specify a port to be used by the HPSS PFTP Daemon. Used only when initiating the Daemon manually (rather than using inetd.
s string	Specify the syslog facility for the HPSS PFTPD. The syntax on the **-s** option is **-slocal7**. The default syslog facility is **LOG_DAEMON** (reference: **/usr/include/sys/syslog.h**). Alternatives are **local0** - **local7**. Incorrect specification will default back to **LOG_DAEMON**. To make use of the alternates, modify **/etc/syslog.conf** to use the alternate facility. Note, the file specified in the **/etc/syslog.conf** must exist prior to initialization/refresh of the **syslogd**. No space is allowed between the **s** and the facility specifier.
t	Read the **hpss_option DTO <value>** option from the {ftpaccess} file to obtain the default timeout in seconds. Refer to documentation on the {ftpaccess} file for details.
u	Read the **hpss_option UMASK <value>** option from the {ftpaccess} file to obtain the default umask to be applied. Refer to documentation on the {ftpaccess} file for details
v	Increment the Client API Logging Specification. The Client API will perform logging specified by the HPSS_DEBUG environment variable in a file specified by the HPSS_DEBUGPATH environment variable (default name is /var/hpss/ftp/adm/hpss_gtpd.log.) The default value for the "v" option is 1. (No "v" option required.) Future implementation may allow for increasing the value to either 2 or 3 to obtain additional Client API information. This would be accomplished by specifying 1 or 2 "v" options on the hpss_pftp startup line.

z	Sleep for 15 seconds at initialization. Useful when attempting to attach to the HPSS PFTP Daemon with the debugger.
A	Enable Passive Connections. Note: Not supported in the HPSS Parallel FTP Daemon. Client requests to use passive mode will be rejected.
C	Disable the ability for clients to explicitly set the Class of Service for new files (via the "**site setcos**" command).
Dstring	Set the **{FTPBaseDir}** path. Default: **/var/hpss/ftp**. This directory must contain several sub-directories including: **adm**, **bin**, **daemon**, and **etc**. Specific files / sub-directories are located in each of these subdirectories - etc: ftpaccess, [ftpbanner], ftppasswd, ftpusers, and [trusted_hosts]. adm: [daemon.syslog], [hpss_ftpd.log], [xferlog]. daemon: ftpd / ftp.pids-hpss_class. [ ] implies optional others are required. etc / ftppassed is optional if the -S or -X options are specified.
Fstring	Set the {**FTP_FtpAccessFile**}. Default: **ftpaccess**. Located in the directory **{FTPBaseDir}/et**c.
G	Read the **hpss_option AMGR <string>** from the **ftpaccess** file to obtain the path / name of the Authentication Manager to be used. Refer to documentation on the **ftpaccess** file for details.
H	Used to disallow login for users whose home directory does not exist or is not properly configured. The default behavior (without the H option) is to put the user in the " / " directory.
I	Toggle the use of trusted hosts. Default is off. Note: this is not usually recommended.
K	Read the **hpss_option KTAB** from the **ftpaccess** file to obtain the {Path} / {Name} of the HPSS Keytab file containing the hpss_ftp principal. Default: / **krb5/hpss.keytabs**. Refer to documentation on the {ftpaccess} file for details.
L	Read the **hpss_option LS <string>** from the **ftpaccess** file to obtain the CDS name for the RPC group of Location Server. Refer to documentation on the **ftpaccess** file for details. This option is mandatory!
P	Read the **hpss_option PRINC** principal from the **ftpaccess** file to obtain the DCE principal representing **hpss-ftp**.
R	Set the HPSS PFTP Daemon to use a specified port range for connections to the HPSS Movers for Parallel Transfer functions. The actual port range is specified in the Mover specific configuration record. Refer to Section 5.6.8 for more information.
S	Use the DCE Registry for Authentication (bypassing the ftppasswd file). All HPSS FTP Customers MUST be in the DCE Registry since either the **hpss_XLoadThreadState**() or **hpss_LoadThreadState**{} call is invoked to obtain the end users identity.

T	Read the **hpss_option MTO <value>** option from the **ftpaccess** file to obtain the maximum timeout in seconds. Refer to documentation on the **ftpaccess** file for details.
U	Use UDP RPCs Only. Must be set! Sets the environment variable: **RPC_SUPPORTED_PROTSEQS=ncadg_ip_udp**. Unpredictable behavior and/or slow operation has been observed with DCE if this parameter is not set.
X	Use the DCE Registry for authentication (bypassing the ftppasswd file) and use the HPSS.homedir and HPSS.gecos Extended Registry Attributes (ERAs) for the users Home Directory and Accounting fields (if they are filled.)

The AIX (and others) inetd superdaemon will fail (without explanation or indication) if errors are encountered in the /etc/inetd.conf file. In particular, lines in excess of ~130 characters will cause this behavior to occur. It is recommended that HPSS Administrators utilize some form of "TCP wrapper" programs in conjunction with the inetd superdaemon (not provided with HPSS.) Most of these applications, do not exhibit the line length limitation observed by the inetd superdaemon and they also allow "on the fly" modification of initialization parameters for network services; e.g., PFTP, telnet, etc., without having to refresh (kill -HUP) the inetd superdaemon.

The "-D string" option which will allow specification of the {**FTPBaseDir**} (Default: **/var/hpss/ftp**) and the "-F string" option which will specify the name of the **ftpaccess** file (Default: **ftpaccess** in the **{FTPBaseDir}/etc** directory.) The **ftpaccess** file will be in the directory specified by **{FTPBaseDir}/etc**. Default: **/var/hpss/ftp/etc**.

Two or more Parallel FTP Daemons (with different options) may be utilized by specifying a different {ftpaccess} file for each. This can be done by specifying different ports and using the /etc/inetd.conf file with multiple unique FTP entries or by using the same port but invoking a different option set based on the address(es) of the client (if the "TCP Wrapper" facilities are used.)

Step 2. Configuring the FTP Daemon Syslog

The FTP Daemon attempts to write to the system log (using **syslog()**). To enable this output, follow local system procedures for capturing this information. T he default syslog is **LOG_DAEMON**. This is modifiable to other options using the –s option on the startup line. NOTE the precautions above!

Step 3. Defining the FTP Access

The **{FTPBaseDir}/etc/{ftpaccess}** file defines: access, options, and restrictions for the server. An ftpaccess.template file is provided in the **$HPSS_DIR/config/template** directory for the HPSS Administrator to customize appropriately. The pound sign (#) at the beginning of a line is used to indicate a comment. Options are invoked by removing the comment symbol (#) and conversely disabled by introducing this symbol. Some options are specific to individual machines; therefore, each HPSS Parallel FTP Daemon should read its own {ftpaccess} file. It is important not to use NFS for the {**ftpaccess**} file if you use the "-h" option **(hpss_option HOST <interface name>**) in the HPSS PFTPD start options! If the HPSS PFTP Daemon specifies any of the "-L, -K, -P, or -h" options, the {**ftpaccess**} file must have the appropriate and correct **hpss_option** record or unpredictable failures may occur. Note that the -L option is mandatory.

The access options can be defined as follows:

Define users as being guests (which means they cannot
cd out of their own subdirectory).

guestgroup <group> [<group> ...]

Set up user automatically into specified group.
NOT CURRENTLY SUPPORTED.

autogroup <group> <class> [<class> ...]

Set the maximum number of login attempts before closing the
connection:

loginfails <login_failure_threshold>

Setup a private group file
NOT CURRENTLY SUPPORTED.

private <private_group_pathname>

Enable / Disable compression filter
NOT CURRENTLY SUPPORTED.

compress [yes | no] <class> [<class> ...]

Enable / Disable tar filter
NOT CURRENTLY SUPPORTED.

tar [yes | no] <class> [<class> ...]

Control for logging (sent to syslog()).

log [commands] [anonymous] [{ inbound }]
[transfers] [guest] [{ outbound }] [real] [debug]

Define a class of users, <user> is a full network address,
including wildcarding, e.g., *.nsl.nersc.gov.

class <class> { anonymous | guest | real } <user>
[<user> ...]

Define a limit on the number of users from a certain class
that can be using the FTP Daemon at specific times.

limit <class> <n> <times> [<msg_file>]

Deny access to a set of users, <addrglob> is full network address,
including wildcarding, e.g., *.nsl.nersc.gov.

deny <addrglob> [<msg_file>]

Send shutdown message to current FTP users.

shutdown <shutdown_info_pathname>

Send message to users, possible on access to a directory.

message <pathname> [<when>]

Display prompt to read a file, possible on access to a directory.

readme <pathname> [<when>]

Determine the appropriate behavior when needed data must be staged.
<stage-code> can have several values:
-1 wait forever
0 do not wait (default) - inform user that data needs staging and exit
>0 wait for n seconds - if staging takes more than n seconds, inform the user and exit
note: This option requires that the relevant storage class be configured in a compatible
way, or behavior may not be as expected. For example, setting wait_on_stage to -1
(wait forever) for a SC with no-stage set would make little sense.

wait_on_stage <stage-code>

Display a banner prior to the user prompt (before log in.) Very useful for informing the user that
he/she is dealing with the HPSS file systems rather than the local file system of the host executing
the FTP Daemon. This is highly recommended!

banner <pathname/filename>

The following "**hpss_options**" are read only if the corresponding flag (See the FTP Daemon Flags above) appears on the inetd.conf initialization line for the HPSS PFTP Daemon and may be left "active" (not commented out with the # symbol) even if the default value is desired.

Define the Authentication Manager

hpss_option AMGR </usr/lpp/hpss/bin/auth_type>

Define the default Blocksize in Bytes

hpss_option BUFSZ <value>

Define the default timeout in Seconds

hpss_option DTO <value>

Disallow HINTS processing

hpss_option HINTS off **/*** The default with this option is on */

Define the Location Server RPC Group Entry (Mandatory)

hpss_option LS </.:/hpss/ls/group>

Define the HPSS FTP Principal

hpss_option PRINC <hpss-ftp>

Define the Maximum timeout in Seconds

hpss_option MTO <value>

Define the default umask

hpss_option UMASK <value>

Define the Keytab table name.

hpss_option KTAB <Path/Name of HPSS Keytab file containing the hpss_ftp principal>

Define the FTPD - Mover Network Interface to be used for non-parallel FTP transfers. Defaults to
the interface associated with the "hostname" of the machine running the HPSS PFTP Daemon.

hpss_option HOST <myhost-fddi>

Note 1. **Message/readme/banner/shutdown** (message lines) are text files, with the following keywords (all one character in length) recognized, identified by a preceding **%**:

Keyword	Description
M	Class limit
T	Current time on FTP Daemon
C	Current working directory
R	Remote hostname
L	Local hostname
U	User name
s	Shutdown time
d	User disconnect time
r	Connection deny time
%	%

Note 2. The format of the **<shutdown_info_pathname>** file is:

<year> <mon> <day> <hour> <min> <deny> <disc>

Message lines contain keywords mentioned above. For example:

1994 1 28 16 0 120 30
System shutdown at %s (current time is %T)
New FTP sessions denied at %r
FTP users disconnected at %d

indicates that shutdown will be 1/28/94 at 16:00, with users disconnected at 15:30 and sessions denied at 14:40 (the 120 indicates 1 hour, 20 minutes).

Note 3. The **<when>** clauses may be either **login** (in which case the file is displayed when the user logs in) or **cwd=[<directory_name> | *]** (in which case the file is displayed when the user changes to the named directory, with * wildcarding all directories).

Note 4. The **<times>** clause has the following format:

<day><times>-<time> | <day><time>-<time> | . . .

where **<day>** is one of **Su, Mo, Tu, We, Th, Fr, Sa, Su, Wk**, or **Any** and **<time>** is the time of day on a 24-hour clock. For example, to specify Tuesday between 8:00 am and 4:00 pm:

Tu0800-1600

The **Wk** entry for the **<day>** clause indicates a week day (Monday-Friday), and **Any** indicates all days of the week.

Step 4. Creating FTP Users

In order for an HPSS user to use FTP, a DCE **userid** and **password** must be created. Refer to Section 6.12.1 for information on how to use the **hpssuser** utility to create the DCE **userid** and **password** and set up the necessary configuration for the user to use FTP. Note that this step should not be done until the NS and the BFS are running so that the **hpssuser** utility can create the home directory for the FTP user.

If desired (this is not recommended), the /bin/passwd_import and /bin/passwd_export utilities can be used to import/export the /etc/passwd file into/from the DCE Security Registry. However, caution should be used so that the /etc/passwd file is not overlaid. Also, note that the /bin/passwd_import and /bin/password_export utilities do not transfer the actual passwords in/out of DCE!

The **/usr/lpp/hpss/bin/hpss_ftppw** utility can be used to change the encrypted passwords in the /**var/hpss/ftp/etc/ftppasswd** file. The syntax for this utility is as follow:

hpss_ftppw <userid> [<password file pathname>]

The utility will prompt the user for the old and new passwords. The password file pathname argument can be used to specify a password file other than the default file, **/var/hpss/ftp/etc/ftppasswd**.

If the HPSS PFTP Daemon utilizes the DCE Registry for authentication (-S or -X options), the {**ftpaccess**} file is superfluous and the rest of this section may be skipped! The HPSS home directory for the user must still be established and configured.

To enable anonymous FTP, the “hpss_ftp” user must be defined in either the HPSS FTP password file or in the DCE registry (depending on which authentication mechanism is enabled). In addition, the entry for the “hpss_ftp” user must contain a home directory defined to be a non-NULL value.

T*he home directory defined for the “hpss_ftp” user will be the root directory for the anonymous ftp session. The user will not be able to change out of the file tree with that directory as its root. Care must be taken if symlinks are created within this directory tree however - as it is possible that symlink will point out of this tree (and therefore allow an anonymous ftp user access outside of the directory tree).*

To disable anonymous FTP, either:

1. Define the "**hpss_ftp**" user entry (in either the HPSS FTP password file or the DCE registry depending on which authentication mechanism is enabled) with a NULL home directory name,

2. If the HPSS FTP password file is used for user authentication, do not define an entry for the "**hpss_ftp**" user,

or

3. Add "**hpss_ftp**", "**anonymous**" or "**guest**" to the HPSS FTP user file (normally "**/var/hpss/ftp/etc/ftpusers**").

5.8.4 NFS Daemon Configuration

Additional HPSS NFS configuration information is specified through the HPSS **exports** file and through environment variables. If the default server name for the HPSS LS is not used, the environment variable for the LS, **HPSS_LS_NAME** should be changed in the **hpss_env** file. Similarly, if the default for the NFS Server descriptive name is not used, the environment variable **HPSS_NFS_DESC_NAME** should also be changed in the **hpss_env** file.

HPSS NFS does not support yellow pages (Sun MicroSystems' Network Information Services) to validate hosts. HPSS NFS does provide an option to validate the network address of hosts attempting to mount HPSS directories. The default configuration disables this check. To enable this check, define the variable **HPSS_MOUNTD_IPCHECK** in the **hpss_env** file.

Note: For users who have several NFS clients concurrently updating and reading the HPSS namespace and depend on having their namespace changes immediately *reflected in all clients, it is necessary to perform NFS mounts with the noac option. Here is an example:*

```
mount -orw,intr,timeo=30,noac hpssserver:/hpss /hpss
```

Using the noac option will degrade performance since it prohibits caching. Only use it if NFS clients rely on immediate HPSS namespace updates.

The HPSS Exports File

The HPSS exports file contains a list of HPSS directories that can be exported to the NFS clients. It also contains an entry for each directory that can be exported to the NFS clients. If this file is changed, the NFS Daemon must be restarted before the changes can affect the way the NFS server operates.

Entries in the file are formatted as follows:

Directory -Option [, Option] ...

These entries are defined as follows:

- **Directory.** Specifies the complete HPSS directory pathname.

- **Option.** Specifies optional characteristics for the directory being exported. More than one variable can be entered by separating them with commas. Choose from the following options:

id=Export-id	Integer between 1 and 255 that specifies a unique identifier for this export entry. This option is required. If duplicate identifiers are found in the hpss exports file, an error message is logged and only the first entry will be included in the exports list.
ro	Exports the HPSS directory with read-only permission. Otherwise, if not specified, the directory is exported with read-write permission.
rw=Client[:Client]	Exports the HPSS directory with read-write permission to the machines/networks specified by the Client parameter and read-only to all others. The Client parameter can be either the host name or the network name. Network names are specified in the local system /etc/networks file. If a Client is not specified, the directory is exported with read-write permission to all.
anon=UID	If a request comes from a client user with the HPSS root identity, use the UID value as the effective user ID. The default value for this option is -2.
root=HostName[:HostName,...]	Gives HPSS root access only to the HPSS root users from the specified host name. The default is for no hosts to be granted root access. Network names are not allowed for this option.
access=Client[:Client,...]	Gives mount access to each client listed. A client can be either a hostname or a network name. Each client in the list is first checked in the host's database and then in the /etc/networks file. The default value allows any machine to mount the given directory.
NOSUID	Causes the NFS server to mask off **setuid mode** bits. The default is for **setuid mode** bits to be allowed.
NOSGID	Causes the NFS server to mask off **setgid mode** bits. The default is for **setgid mode** bits to be allowed.
UIDMAP	Causes the NFS server to require an entry in the NFS credentials map cache for all non-mount related requests. The default is to not require UID mapping.

key=Key	Specifies an alpha numeric string up to 8 characters that will be used as an encryption key when the mount and NFS servers are configured with Encrypt File Handles enabled. This option is required if Encrypt File Handles is enabled. The default is for no key.

A # (pound sign) anywhere in the HPSS exports file indicates a comment that extends to the end of the line.

Examples

1. To export the HPSS directory **/usr** to network sandia.gov, enter:
   /usr -id=1,access=sandia.gov

2. To export the HPSS **/usr/local** directory to the world, enter:
   /usr/local -id =4

3. To export the HPSS directory **/usr2** only to these systems, enter:
   /usr2 -id=5,access=hermes:zip:tutorial

4. To give root access only to these systems, enter:
   /usr/tps -id=20,root=hermes:zip

5. To convert client root users to guest UID=100, enter:
   /usr/new -id=10,anon=100

6. To export read-only to everyone, enter:
   /usr/bin -id=15,ro

7. To allow several options on one line, enter:
   /usr/stuff -id=255,access=zip,anon=-3,ro

Files

/var/hpss/nfs/rmtab	Lists all currently mounted HPSS directories.
/etc/hosts	Contains an entry for each host on the network.
/etc/networks	Contains information about subnetworks on the network.

5.8.5 MVR Configuration to Support IPI-3

This section describes the configuration for the MVR to support IPI-3 data transfers over HiPPI, which is currently only supported on AIX platforms. If this feature is not being used, skip the entire section. The following information assumes the reader has some familiarity with HiPPI and IPI-3.

HPSS supports both source routing and logical routing over HiPPI networks. When using source routing, the i-field contains the set of HiPPI ports that must be traversed to get from the source of a packet to the destination, and usually requires no special configuration of the HiPPI switch(es).

Source routing is usually sufficient for an HPSS configuration; however, if a disk array and the MVR controlling that disk array are not physically attached to the same HiPPI switch, source routing cannot be used. This is because the i-field to communicate to the client from the disk array will be different from that used from the Mover machine. In this case, logical routing must be used, which requires that the HiPPI switch(es) be configured to give each node on the HiPPI network a logical address (a 12-bit value). This value can be used in the i-field from any point on the HiPPI network to communicate with the client.

For supporting IPI-3 data transfers over HiPPI, two configuration files are required: the **ipi3en.conf** file and the **ipi3data.conf** file. These files are located in the directory named by the **HPSS_PATH_ETC** environment variable. If the environment variable is not set, HPSS will look for the files in the **/usr/etc** directory.

The ipi3en.conf File

The ipi3en.conf configuration file contains four types of records used by the HPSS Movers. The Device and Partition records are used to define the HiPPI attached disk devices. The Slave and Client records are used for MVR communications with IPI-3 clients.

Note that the numeric fields discussed in this section can be entered as decimal (default), hexadecimal (0x prefix) or octal (0 prefix).

- **Device Record**. The format of the **Device** record is:

  DEVICE <name> <type> <nstk> <slaveid> <port> <netaddr>
  <h-port> <log-addr> <64-bit>

  where:

  DEVICE is the keyword identifying the record type.

  <name> is the name used to identify the device.

  <type> is the type of device. Currently, only **DISK** is supported.

  <nstk> is the number of maximum number of commands that will be stacked (queued) on this device, and is in the range 1 to 128.

  <slaveid> is the slave identifier used by the disk array. This value must match the value configured in the array, and is in the range of 0 to 255. The default value for the array is usually 0.

  <port> is the command socket port number used by the array, in the range of 0 to 32767. This value must match the value configured in the array. The default value for the array is usually 1024.

  <netaddr> is the IP address or IP hostname of the array's Ethernet connection.

  <h-port> is the HiPPI data port number. Usually, this value will be 1, 2, 3 or 4, depending on the number of HiPPI ports available on the disk array.

<log-addr> is the logical address of the array if logical routing is to be used over the HiPPI network. If only source routing is to be used, this value is unused and can be set to 0x000. Note that this field may not exceed 12 bits (3 hexadecimal digits).

<64-bit> indicates whether the device supports 32-bit transfers (value equal to zero) or 64-bit transfers (value equal to one).

- **Partition Record**. The format of the **Partition** record is:

  PARTITION <name> <facility> <partition> <block-size>
  <start-block> <num-blocks> <max-connections>

  where

  PARTITION is the keyword identifying the record type.

  <name> is the partition name. This name must match the device name configured in the MVR device metadata. There is currently no relationship required between the name of a Device and the name of a Partition; however, for administrative convenience, the Partition name usually contains the Device name (e.g., if a Device name is **dka0**, Partition names frequently take the form **dka0.0**, **dka0.1**, etc.).

  <**facility**> is the facility number on the disk array, in the range of 0 to 255.

  <**partition**> is the partition number on the disk array in the range of 0 to 255 or **N/A**. Note that the block numbers used for the <**start-block**> field start at 0 for each physical partition within the Maximum Strategy device. Also, the partition numbers for the <**partition**> field are the hexadecimal numbers shown on the SMC console's Main.Facility.Partition display. Note that "N / A" or "0" are not valid input for this field if the Maximum Strategy device has multiple physical partitions.

  <**block-size**> is the size, in bytes, of the disk array blocks. This value must match the format of the disk array. This value is usually 32768 or 65536 bytes.

  <**start-block**> is the starting block number of this partition.

  <**num-blocks**> is the number of blocks in this partition.

  <**max-connections**> is the maximum number of connections that should be attempted for this partition. The disk array currently supports a maximum of 10 concurrent connections. This is an optional field that will default to 4 if no value is present.

Note that the Partition record refers to the last Device record preceding the Partition record in the file, and that it is invalid to have a Partition record preceding the occurrence of the first Device record in the file.There is currently a limit of 32 partition records within the ipi3en.conf file.

Example:

```
#
# device record
#DEVICE name type nstk slaveid port# netaddr h-port log-addr 64-bit
#
```

```
DEVICE dka0 DISK 16 0 1024 ibm9570 2 0x00B 0
#
# partition records
#PARTITION name facility partition# blksz strtblk nblks max-conn
#
PARTITION dka0.test    2   N/A   65536   0       5000    2
PARTITION dka0.0       2   N/A   65536   5000    30000   2
PARTITION dka0.1       2   N/A   65536   35000   60000   2
PARTITION dka0.2       2   N/A   65536   95000   81920   4
PARTITION dka0.3       2   N/A   65536   176920  163840  4
```

- **Slave Record**. The format of the **Slave** record is:

  SLAVE <slave-name> <slave-device> <log-addr> <slave-identifier>

  where

  SLAVE is the keyword identifying the record type.

  <**slave-name**> is the name used to identify the slave, and must be of the form **hpss_mvr.<mover TCP/IP port>.<mover task number>,** where **<mover TCP/IP port>** is the port number configured in the MVR specific configuration file, and **<mover task number>** begins at zero, and is incremented for each possible mover task.

  <**slave-device**> is the name of the AIX IPI-3 slave device, usually **/dev/ipihipS0**, if a single HiPPI card is installed (the precise name can be determined from the system configuration information).

  <**log_addr**> is the logical address of the local machine if logical routing is to be used on the HiPPI network. If only source routing is to be used, this field is unused and can be set to 0x000. Note that the value of this field must not exceed 12 bits (3 hexadecimal digits).

  <**slave-identifier**> is a unique non-negative integer in the range of 0 to 255, (each slave record must have a unique slave identifier).

For Movers that will control a HiPPI attached disk array, this value should be unique amongst Movers controlling the disk array, even if those Movers run across separate nodes (but need not be unique for Movers controlling different disk arrays).

Example set of slave records:

```
SLAVE   hpss_mvr.5001.0   /dev/ipihipS0   0x000   0
SLAVE   hpss_mvr.5001.1   /dev/ipihipS0   0x000   1
SLAVE   hpss_mvr.5001.2   /dev/ipihipS0   0x000   2
SLAVE   hpss_mvr.5001.3   /dev/ipihipS0   0x000   3
```

- **Client Record**. The format of the **CLIENT** record is:

  CLIENT <client> <dev> <interface> <addr> <max-length> <max-stack>

  where

  CLIENT is the keyword identifying the record type.

<**client**> is the client name (see the description of the **ipi3data.conf** file below).

<**dev**> is used for cascading, with a value of ***** meaning **all**.

<**interface**> is the interface type, and must be set to **HiPPI**.

<**addr**> is the HiPPI i-field address, and is based on the client's HiPPI switch connectivity. Note that if source routing is used for communication with this client, the i-field is the set of HiPPI ports that must be traversed to reach the client from either the local machine or the disk arrays. If logical routing is to be used to communicate with the client, the i-field contains the logical address of the client in the low-order 12 bits and must have the bit set to indicate logical addressing, which is 0x02000000 (see the client entry for cyclone in the example, below). If clients are configured to use logical routing, the logical address fields must be correctly filled in for the Device and Slave records, as described above.

<**max-length**> is the maximum length in bytes of a single data packet that will be sent to the client.

<**max-stack**> is the maximum number of commands that may be stacked (queued) on behalf of the client.

Example set of client records:

```
CLIENT   demofb    *    HiPPI    0x41000007    1048M    15
CLIENT   tempest   *    HiPPI    0x0100000F    64M      15
CLIENT   cyclone   *    HiPPI    0x0300001C    64 M     15
```

The ipi3data.conf File

The ipi3data.conf configuration file contains a single record, which describes the machine's IPI-3 client interface. The format of the record is:

<**interface**> <**client-name**> <**master-device**>

where

<**interface**> is the type of interface and must be set to **HiPPI**.

<**client-name**> is the client's name that is passed in the client's I/O request and must match the name in the client record of the ipi3en.conf file, as described above.

<**master-device**> is the name of the IPI-3 master device, which is typically **/dev/ipihipM0** for an AIX system with a single HiPPI card installed. (The precise name can be determined from the system configuration information.)

5.8.6 MVR Configuration to Support Non-DCE Execution Mode

This section describes the configuration for the MVR to support running the Mover in non-DCE mode - which involves part of the Mover running on a DCE/Encina platform (currently AIX or Solaris), while the part of the Mover that manages storage devices and performs data transfers runs on a non-DCE/Encina platform (currently AIX, Solaris or IRIX).

The Mover can be run in a non-DCE mode, in which case the part of the Mover that handles accessing configuration metadata and servicing the Mover administrative interface (still built on top of DCE and Encina) runs on a DCE/Encina platform (currently AIX or Solaris), while the part of the Mover (to which the PVL and Storage Servers communicate) that manages the storage devices and performs data transfers runs on a remote node that does not require DCE or Encina services. To support this mode of operations, there is some additional configuration which must be done on the remote (non-DCE/Encina) node.

The /etc/services *and* /etc/inetd.conf *Files*

To support starting the non-DCE/Encina part of the mover, the remote nodes' *inetd* is utilized to start the parent process when a connection is made to a port based on the Mover's type specific configuration (see section 5.6.8).

An entry must be added to the */etc/services* file so that the *inetd* will be listening on the port to which the Mover parent process (running on a DCE/Encina node) will connect. The port number is one greater than that configured for the "TCP Listen Port" in the Mover's type specific configuration. For example, if the configure listen port is 5001, the following line must be added to */etc/services*:

hpss_mvr1 5002/tcp

A corresponding entry must be added to the */etc/inetd.conf* file, which will define which executable to be run and the arguments to be used when a connection is detected. The sole argument (other than the program name) is the pathname to a file that will contain an ASCII representation of the configured encryption key for the Mover (see the next section for further details). For example:

hpss_mvr1 stream tcp nowait hpss /var/hpss/bin/hpss_mvr_dd2 hpss_mvr_dd2 /var/hpss/bin/ek.mvr1

Which will cause the executable */var/hpss/bin/hpss_mvr_dd2* to be run under the *hpss* user id when a connection is detected on port 5002. The Mover process will use the */var/hpss/bin/ek.mvr1* file to read the encryption key that will be used to authenticate all connections made to this Mover.

The Mover Encryption Key Files

To authenticate access made to the non-DCE Mover processes, the encryption key configured in this Mover's type specific configuration (see section 5.6.8) is read from a file accessible from the local file system. This file contains an ASCII representation of the encryption key (the pathname of the file is passed to the Mover executable as specified in the */etc/inetd.conf* file). For example, if the encryption key in the Mover's type specific configuration is **1234567890ABCDEF**, then the encryption key file should contain:

0x12345678 0x90ABCDEF

System Configuration Parameters for IRIX Platforms

When running the non-DCE Mover process on an IRIX platform, there are a number of system configuration parameters which may need to be modified before the Mover can be successfully run. The values can be modified with the *systune* utility (and will likely require the system to

be rebooted before they take effect). The following table defines the parameter names and minimum required values.

Note that the **semmns** and **semmnu** values should be increased if running more than one Mover on the IRIX machine(multiply the minimum value by the number of Movers to be run on that machine).

Table 5-27 IRIX System Parameters

Parameter Name	Minimum Value	Parameter Description
semmns	1024	Maximum number of semaphores in system
semmnu	512	Maximum number of undo structures in system
semmsl	512	Maximum number of semaphores per set
maxdmasz	513	Maximum DMA size (required for Ampex DST support)

System Configuration Parameters for Solaris Platforms

When running the Mover or non-DCE Mover process on a Solaris platform, there are a number of system configuration parameters which may need to be modified before the Mover can be successfully run. The values can be modified by editing the */etc/system* configuration file and rebooting the system. The following table defines the parameter names and minimum required values.

Note that the **semmns** and **semmnu** values should be increased if running more than one Mover on the Solaris machine(multiply the minimum value by the number of Movers to be run on that machine).

Table 5-28 Solaris System Parameters

Parameter Name	Minimum Value	Parameter Description
semsys:seminfo_semmns	1024	Maximum number of semaphores in system
semsys:seminfo_semmnu	512	Maximum number of undo structures in system
semsys:seminfo_semmsl	512	Maximum number of semaphores per set
shmsys:shminfo_shmmax	8388608	Maximum shared memory segment size (will allow up to a 2MB Mover buffer size)

5.8.7 MVR Configuration to Support Local File Movement

The Mover data transfer protocol allows the Mover to transfer data directly between a local Unix file system and HPSS. This transfer can only occur if the Mover has direct access to the Unix file system on which the file resides.

To enable this feature, a configuration file must be created on each node that is running a Mover. The configuration file resides at **/var/hpss/ect/hpss_mvr_localfilepath.conf**. The file contains a list of Unix paths. These paths specify which files are allowed to be moved using this special data transfer protocol. Any Unix file on the Mover node whose path starts with any of the paths listed in the configuration file is allowed to be transferred using the local file movement protocol.

If the configuration file does not exist, SSM will issue a minor alarm with the Mover is started, and the local file movement protocol will be disabled.

Here is an example configuration file:

```
# This file contains Unix paths that can be accessed by the local Movers
# on this node. If the client wishes to make a local file transfer and the
# start of the client path matches any of the paths in this list then the
# transfer will proceed, otherwise the Mover will not transfer the file.
#
# The format of this file is simply a list of paths, one per line.
/gpfs
/local/globalfilesystem
```

In the above sample configuration, any file whose pathname begins with either‘/gpfs’ or ‘/local/globalfilesystem’ may be transferred using the special data protocol.

Care should be taken when using this feature. The Mover will move data from any file its list allows. If the file system is not global, then the Mover will use the local Unix file system. If the HPSS data is then striped across multiple Mover nodes, then the data will land is multiple local Unix files.

Also, for both reads and writes, the Unix file must already be created before the data transfer starts. It is up to the client to create the file. The Mover will not create it automatically (for security reasons).

5.8.8 MPI-IO API Configuration

The following environment variables can be used to define the MPI-IO API configuration:

The **MPIO_LOGIN_NAME** defines the DCE login name of the principal who will be executing the application. If specified along with **MPIO_KEYTAB_PATH**, which specifies the path name to a file containing the principal’s security keys, each process in a distributed application will be able to create the credentials necessary for DCE authentication to HPSS. If either is not specified, the default is to use the DCE credentials for the current login session, if any.

The **MPIO_DEBUG** can be toggled to direct MPI-IO to give messages when errors are detected. A value of 0 (zero) will disable message reporting, and any nonzero value will enable reporting.

In addition to the MPI-IO-specific variables, any of the HPSS Client API variables can be used (see Section 5.8.1).

5.8.9 AML PVR Configuration

Users must configure the Insert/Eject ports for the AML PVR using the configuration files **/var/hpss/etc/AML_EjectPort.conf** and **/var/hpss/etc/AML_InsertPort.conf**. The AML robot can have multiple insert and eject ports, which have the capability to handle different media types. These two configuration files in conjunction with the AMU AMS configuration files specify which Insert/Eject areas or bins the tapes should be placed in for insertion into the archive and where they are ejected to when the HPSS export command is used.

For more information on configuring the AML PVR, see Appendix L and Section 5.6.7

5.8.10 Network Configuration

HPSS makes extensive use of a system's networking capabilities. Therefore, the setting of the tunable networking parameters for the systems on which the various HPSS servers and clients will run can have a significant impact on overall system performance.

Under AIX, a utility is provided to display and modify a number of networking parameters. The utility is **_no_** (Network Options). Refer to the *AIX Versions 3.2 and 4.1 Performance Tuning Guide*, SC23-2365-03, for details of each option and its impact on networking and system performance.

Some options that typically impact performance within an HPSS system environment are:

thewall	Controls the maximum amount of system memory that can be used by the system networking code. A value that is too low can cause networking requests to be delayed or denied. The recommendation from AIX development is to set this value to at least two times the maximum number of concurrent connections times the size of the socket send/receive buffers. Note that currently the maximum amount of memory that the system will allocate for networking is 64 MB.
sb_max	Controls the maximum size allowed for send and receive buffers for a socket.
udp_recvspace	Controls the default size of the receive buffer for UPD/IP sockets. A value that is too small can cause server RPC sockets to be overrun.
tcp_recvspace, tcp_sendspace	Controls the default size for the receive and send buffers for TCP/IP sockets. Internally, HPSS servers and clients attempt to set these buffers sizes explicitly, but other utilities may not.
rfc1323	Controls whether large TCP window sizes are used. Usually set to ON for higher throughput networks (e.g., HiPPI, SP switch) and set to OFF for lower throughput networks (e.g., ethernet, FDDI).

AIX also provides a configuration attribute that controls the maximum amount of memory that can be allocated to **mbufs**. It can be viewed or modified via **smit** (select "**Process Environments**", then "**Change / Show Characteristics of Operating System**") or via the command line ("**lsattr -E -l sys0**", "**chdev -e sys0 -a maxmbuf = <new value>**").

It is recommended that the available combination of options be tested as part of the initial HPSS system testing. In particular, poor network performance has been experienced where options on one system do not match options on other remote systems.

There are also attributes that are specific to the individual network interface that may affect network performance. For example, the network interface for the IBM SP TB3 switch provides settings for the size of the send and receive pool buffer size, which have had an effect on throughput. It is recommended that the available interface specific documentation be referenced for more detailed information.

The anticipated load should also be taken into account when determining the appropriate network option settings. Options that provide optimal performance for one or a small number of transfers may not be the best settings for the final multi-user workload.

The hpss_netopt.conf File

The **hpss_netopt.conf** configuration file contains network options to be used by the HPSS Client API and Mover. The file allows different options to be specified based on IP address - when the Client API or Mover establish connections, they will search the contents of this file for a matching IP address and use those options specified in the matching entry.

The configuration file entries contain values to be used for the socket send and receive space, an indication of whether **RFC 1323** support (which allows for increased performance over some networks, but may have a negative performance impact on others) should be enabled, the size of individual write commands to the network and an indication of whether the delay to coalesce small packet option should be enabled or disabled. Note that currently the ability to enable or disable **RFC 1323** support in this manner is specific to AIX platforms. Which entry to use is based on the specified IP address and netmask value. The Client API or Mover will apply the netmask specified in the configuration file entry to the current address to see if it matches the IP address specified in that same entry. The first matching entry found in the file will be used to determine the network options used for that connection.

The network write size allows the size of the individual write requests to the TCP/IP connections to be configured. The default behavior (if no entry in the file matches a connection or if zero is entered for the value of this field) is that the size of the write request is the size of the data buffer. On some networks (e.g., HiPPI and the SP/x switch), improved performance has been measured by using a smaller value (e.g., 32KB) for the size of the individual writes to the network. If no entry is found that matches a network connection or the value specified is zero, HPSS will query an environment variable, **HPSS_TCP_WRITESIZE**, and use that value, if set and non-zero, for the write size.

The TCP Nodelay option determines whether HPSS will enable or disable the algorithm that tries to improve performance from small network writes. This algorithm attempts to coalesce small writes to a TCP/IP connection so they can be sent in a single packet by delaying physical writes to the network. HPSS typically disables this algorithm so that delays are not experienced while sending Mover Protocol and parallel data transfer headers. However, if this causes a performance degredation on a specific network (e.g., causes smaller than optimal packet sizes for large transfers), this can be disabled for data transfer connections

The network options configuration file is located in the directory named by the **HPSS_PATH_ETC** environment variable. If the environment variable is not set, the default directory is **/usr/etc.** If the file is not present or no matching entry is found, the Client API and Mover will default to trying to set the largest send and receive buffers possible (that are powers of two), up to a maximum of 1MB.

The format of the file entries is:

<IP Address> <Net Mask> <RFC1323> <SendSpace> <RecvSpace> <Net WriteSize> <TCP nodelay>

where:

<IP Address>	Contains the dotted decimal IP address of the host / network
<Net Mask>	Contains the dotted decimal net mask to apply to the address to determine whether the entry applies
<RFC1323>	Indicates whether the RFC1323 option should be disabled ('0') or enabled (any other value)
<SendSpace>	Contains the value to be used for the socket send buffer space
<RecvSpace>	Contains the value to be used for the socket receive buffer space
<Net WriteSize>	Size to be used for each individual write request to the network
<TCP nodelay>	Indicates whether the TCP algorithm to coalesce small writes should be disabled ('0') or enabled ('1')

NOTE: To specify an entry to be used if no other entries match, specify an IP address of 0. Also note that this entry should be the last in the file, as the file entries are checked in order with the first matching entry being selected. The default value could be used so that the system default values (as specified via the "**no**" command under AIX) can be set differently to allow those values to be used by non-HPSS applications.

Example:

```
192.94.47       255.255.255.0   0   16384   16384   8192    1
192.225.22      255.255.255.0   0   65536   65536   65536   1
192.225.23      255.255.255.0   1   524288  524288  262144  1
0               0               0   262144  262144  32768   1
```

This example might correspond to the first entry representing an Ethernet network, the second an FDDI network, the third an ATM or Fibre Channel network (note the increasing send / receive space values and the setting of RFC 1323 support for the higher speed network), with the last entry indicating defaults to be used by the Client API and Mover if no other matching entries are found.

Valid entries are supported that do not include the last two fields (Network Write Size and TCP Nodelay indication) to maintain compatibility with previous releases. In this case, the Network Write Size will be taken to be zero and the value for the TCP Nodelay indication taken to be one.

5.8.11 SP/x Switch Device Driver Buffer Pools

IBM SP/x systems provide the capability to tune the buffer pool allocation in the switch device driver. Two variables can be changed: **rpoolsize**, which is the size of the buffer pool for incoming data, and **spoolsize** which is the buffer pool size for outgoing data. If these values are too small, then buffer overruns may occur.

The current values of these variables can be interrogated with the **lsattr** command (e.g., "**lsattr -E -l css0**"), and can be changed with the **chgcss** command (e.g., "**chgcss -l css0 -a spoolsize=<new size> -a rpoolsize=<new size>**"). Refer to the *IBM Parallel System Support Programs for AIX Administration Guide, GC23-3897-02* for more details.

5.8.12 HiPPI Device Parameter Settings

The various HiPPI devices that are configured into an AIX system (dependent on whether TCP/IP and/or IPI-3 support is installed) provide a number of configuration parameters to control resources available to the HiPPI subsystem. On nodes that run both TCP/IP and IPI-3 over HiPPI, problems have been seen running with the default system settings. The following settings are recommended both for large transfer sizes over IPI-3 and when running both TCP/IP and IPI-3 over HiPPI on the same node.

Note that all values may be set via SMIT, and will not take effect until the system has been rebooted (the following output should reflect the corresponding smit screens). Refer to the *High-Performance Parallel Interface User's Guide and Programmer's Reference (SC23-2780-00)* for further details.

Note that depending on the system configurations, the "Bus DMA Region Width" may need to be increased (e.g., to 0xC000000). If, during system startup after changing these options, the HiPPI subsystem logs a message the AIX error log, indicating that bus space could not be allocated, then this value should be increased.

HiPPI Adapter

HiPPI Adapter	HiPPI0
Description	N/A
Status	Available
Location	00-11
Outbound SCB Pipe Size	[4096]
Inbound SCB Pipe Size	[4096]
Size of Small Buffers for Unsolicited Data	[256]
Number of Small Buffers	[16]

Size of Large Buffers for Unsolicited Data	[4096]
Number of Large Buffers	[16]
Bus DMA Region Width	[0xA000000]
Apply change to DATABASE only	yes

HiPPI Common Functions

Name	hcom0
Status	Available
Description	N / A
Parent adapter	HiPPI0
Translate space (Kbytes)	[134000]
Translate segments	[256]
Buffer pool space (Kbytes)	[128]
Buffer pool segments	[32]
Maximum transfer (Kbytes)	[64000]
Configure at startup	[yes]
Change applicability	[All]

IPI-3/HiPPI Master Transport

Name	ipihipM0
Status	Available
Description	N / A
Parent device	hcom0
Protocol ID	06
Buffer pool size	[6]
Connection timeout (milliseconds)	[5000]
Transmission timeout (milliseconds)	[5000]
Minimum byte count to establish connection	[0]

Starting CRN [1]

Ending CRN [16382]

Configure at startup [yes]

Change applicability [All]

IPI-3/HiPPI Slave Transport

Name ipihipS0

Status Available

Description N / A

Parent device hcom0

Protocol ID 07

Buffer pool size [6]

Connection timeout (milliseconds) [5000]

Transmission timeout (milliseconds) [5000]

Minimum byte count to establish connection [0]

Configure at startup [yes]

Change applicability [All]

IP Interface (Use smit -C chifhp)

Network Interface hp0

Maximum IP PACKET SIZE for THIS DEVICE [65280]

Connection Timeout in Milliseconds [50]

Transmission Timeout in Milliseconds [50]

Minimum Byte Count to establish connection [0]

Should the adapter wait for receive buffers? [no]

Should the adapter secure pad characters? [no]

---Network Interface Reserved Buffer Pool---

Number of Pages(4K) per Type-A Recv Buffer [1]

Total Number of Type-A Recv Buffers [30]

Low-Water Mark, Recv Type-A Buffer Pool [20]

Number of Pages(4K) per Type-A Xmit Buffer [1]

Total Number of Type-A Xmit Buffers [8]

Low-Water Mark, Xmit Type-A Buffer Pool [4]

Number of Pages(4K) per Type-B Recv Buffer [2]

Total Number of Type-B Recv Buffers [10]

Low-Water Mark, Recv Type-B Buffer Pool [6]

Number of Pages(4K) per Type-B Xmit Buffer [2]

Total Number of Type-B Xmit Buffers [8]

Low-Water Mark, Xmit Type-B Buffer Pool [3]

Number of Pages(4K) per Type-C Recv Buffer [8]

Total Number of Type-C Recv Buffers [20]

Low-Water Mark, Recv Type-C Buffer Pool [14]

Number of Pages(4K) per Type-C Xmit Buffer [8]

Total Number of Type-C Xmit Buffers [6]

Low-Water Mark, Xmit Type-C Buffer Pool [3]

Number of Pages(4K) per Type-D Recv Buffer [16]

Total Number of Type-D Recv Buffers [30]

Low-Water Mark, Recv Type-D Buffer Pool [20]

Number of Pages(4K) per Type-D Xmit Buffer [16]

Total Number of Type-D Xmit Buffers [4]

Low-Water Mark, Xmit Type-D Buffer Pool [2]

5.9 Making HPSS Operational

5.9.1 Starting the HPSS Servers

After the HPSS servers and their associated data are configured, they can be started up using SSM. When they are started for the first time after HPSS configuration, it is recommended that the servers be brought up one at a time, even though it is possible to start up all the configured servers at once. Doing so will simplify the diagnostics of any server startup problems due to configuration errors.

SSM can be used to start up the following types of HPSS server:

- Name Server
- Bitfile Server
- Storage Server
- Migration / Purge Server
- Location Server
- DMAP Gateway
- Physical Volume Library
- Physical Volume Repository
- Mover
- Log Daemon
- Log Client
- NFS Daemon
- NFS Mount Daemon
- Metadata Monitor
- Non-DCE Client Gateway

Before starting up the HPSS servers, ensure that all configured HPSS Startup Daemons are up and running. As part of the HPSS initial configuration, the Startup Daemon was configured to be invoked at the operating system startup. However, after configuration, it will need to be manually started. Invoke the **/etc/rc.hpss** script as **root** to start the HPSS Startup Daemon. As a default, the **rc.hpss** script will redirect all servers' output to **/dev/console**. The **rc.hpss** script can also be invoked with the **-o** option if redirection of the output to **/dev/console** is not desired.

If the servers' output is to be redirected to /dev/console, ensure that the HPSSLOG variable in the hpss_env file is not set to stdout since this combination may severely degrade the HPSS performance.

It is recommended that the Log Daemon and the Log Client(s) be started up before the other HPSS servers so that the HPSS servers' startup messages are logged to aid in the startup problem determinations.

Ensure that any required PVR controllers are up and running before bringing up the PVRs.

Refer to Section 6.3.2 for more information on server startup.

5.9.2 Unlocking the PVL Drives

As a default, all newly configured drives are locked. They must be unlocked before the PVL can use them. Refer to Section 6.6.5.1 for more information on unlocking a PVL Drive.

5.9.3 Creating HPSS Storage Space

Adding storage space in HPSS is done in two distinct phases: import and create. The first phase, import, involves physically introducing the tape cartridges and the disk volumes into the PVL and labeling them with HPSS volume labels. The second phase, create, defines the Storage Server data structures that will describe the data stored on the imported volumes.

Refer to Section 6.4.1 for more information on creating HPSS storage space.

5.9.4 Creating File Families

Refer to Section 5.5.4 for more information on how to create file families.

5.9.5 Creating Filesets and Junctions

SSM can be used to create DFS / HPSS and HPSS-only filesets. SSM can also be used to create the junctions that link the filesets to the HPSS name space.

Before creating the DFS/HPSS filesets, DFS must already be configured and the procedures to set up the DFS filesets must already be performed. Refer Chapter 7 for more information on the DFS and DFS filesets configuration.

Ensure that the Name Server and the DMAP Gateway are up and running before creating the filesets and junctions. Also ensure that SSM is connected to both of the servers.

5.9.6 Creating HPSS directories

Using an HPSS namespace tool such as **scrub** or **pftp**, the **/log** directory must be created. This directory must be owned by **hpss_log** and have permissions **rwxr-xr-x**.

5.10 Verifying HPSS Configuration

After HPSS is up and running, the administrator should use the following checklist to verify that HPSS was configured correctly:

5.10.1 Servers

- Verify that all required HPSS servers are configured properly and the servers are up and running.
- Verify that a log policy is configured for each server. The HPSS log should be reviewed periodically to verify that the desired level of information is logged. In addition, verify that all Mover log policies have the **DEBUG** flag turned on to aid in the diagnostics of future data transfer problems.

5.10.2 Devices and Drives

- Verify that all devices / drives are configured and each is assigned to an appropriate PVR / Mover.
- For tape devices, verify that the "**Locate Support**" option is enabled (unless there are unusual circumstances why this functionally is not or cannot be supported).
- For tape devices, verify that the "**NO-DELAY** " option is enabled (unless there are unusual circumstances why this functionally is not or cannot be supported).
- For disk devices, verify that the "**Multiple Mover Tasks**" flag is enabled.
- For disk devices, verify that the "**Bytes on Device**" and "**Starting Offset**" values are correct, and the sum of the two values does not exceed the actual size of the underlying device.
- Verify that all configured drives are unlocked.

5.10.3 Storage classes

- Verify that all storage classes are defined and each has sufficient storage space created to store HPSS files.
- Each storage class that will support migration and purge are configured with the appropriate migration and purge policy.
- A storage class at the lowest level in a hierarchy must not be configured with a migration or purge policy.
- A tape storage class must not have a purge policy.
- Each storage class must have an associated MPS.
- To support repack and recover of tape volumes, the stripe width of a each tape storage class must not be more than half of the number of available drives of the same drive type.

5.10.4 Storage Hierarchies

- Verify that all storage hierarchies are defined properly.

5.10.5 Classes of Service

- All classes of service are defined properly.
- Each COS is associated with the appropriate storage hierarchy
- Verify that the COS is intended to use the characteristics of the hierarchy and the underlying storage classes. In addition, verify that the classes of service have the correct **Minimum File Size** and **Maximum File Size** values. If these sizes overlap, the file placement may be indeterminate when the user creates a file using the size hints. For classes of services which are not to be used as part of standard file placement, set their **Force Selection** flag to ON so that they will only be chosen if specified by their COS ID.

5.10.6 File Family, Filesets and Junctions

- File families and filesets are created according to the site's requirement.
- Each fileset is associated with the appropriate file family and/or COS.
- Each fileset has an associated junction.

5.10.7 User Interfaces

- Verify that the desired HPSS user interfaces (FTP, NFS, DFS etc.) are properly configured.

5.10.8 Operational Checklist

Use each configured user interface

- Create files of various sizes on each defined COS.
- Verify that the files are created on the expected storage class with acceptable transfer rates.
- If necessary, redefine the associated storage class definition to enhance the throughput performance.
- The characteristics fields (**Transfer Rate**, **Latency**, etc.) in the storage class and class of service definition should be updated to reflect actual performance results.
- After the files are created on the correct storage class, verify that the files are created with correct file ownerships and permissions.
- Verify that other file operations (delete, chmod, copy, etc.) work properly.
- If accounting is configured, verify that the files are created with the correct accounting indices.
- If file families, filesets and junctions are configured, verify that they work as intended.

Verify Storage Management

- Verify that migration and purge operations work as intended.
- Force Migration and Force Purge operations to verify that files are migrated and purged correctly.
- Verify that files can be accessed after being migrated / purged.
- Monitor free space from the top level storage class in each hierarchy to verify that the migration and purge policy are sufficient in maintaining adequate free space.

5.10.9 Performance

Measure data transfer rates in each COS for:

- Client writes to disk
- Migration from disk to tape
- Staging from tape to disk
- Client reads from disk

Transfer rates should be as fast as the underlying hardware. The actual hardware speeds can be obtained from their specification and by testing directly from the operating system. For example, using **dd** to read and write to each device. Keep in mind that performance testing can often be limited by other factors external to HPSS. For example, a client reading a file from HPSS may be limited by the performance of the UNIX file system while writing the file rather than while reading the file from HPSS.

Chapter 6

HPSS Management

6.1 Overview

This chapter provides instructions and supporting information for most day-to-day or regularly-performed administrative and operational procedures for HPSS. Refer to Chapter 8 (HPSS Special Intervention Procedures) for information on special intervention or exception-handling procedures that do not fall under day-to-day or regular management.

To successfully manage HPSS, the administrator or the operator responsible for day-to-day management should be familiar with the material in this entire document. The procedures described in this chapter assume that the HPSS installation, HPSS infrastructure configuration, and HPSS configuration have been completed.

All HPSS management procedures can be performed through the SSM windows. In addition, a number of the HPSS utilities are also available to aid the administrator in managing HPSS. The primary management procedures described in this chapter include:

- Monitoring HPSS Health and Status (Section 6.2)
- Managing HPSS servers (Section 6.3)
- Managing Filesets and Junctions (Section 6.5)
- Managing HPSS storage space (Section 6.4)
- Managing HPSS devices and drives (Section 6.6)
- Monitoring HPSS managed objects (Section 6.7)
- Generating an accounting report (Section 6.8)
- Managing HPSS logging services (Section 6.9)
- Backing up HPSS metadata (Section 6.10)
- Managing HPSS users (Section 6.12)
- Using HPSS utilities (Section 6.13)

HPSS Managed Objects

HPSS uses the concept of *managed objects* and attributes for much of the administrative and operational management of the system and its components. Use of managed objects is also common in other kinds of management applications, notably systems and network management. The managed objects in HPSS represent a generalized management view of servers and resources in the storage system. Obtaining information about HPSS objects and controlling the behavior of these objects can be accommodated.

Viewing HPSS objects and attributes to monitor the HPSS system is, in general, a safe procedure. Modifying attributes, particularly those related to the state of the object, may have serious consequences. Whenever you modify an attribute, be certain that you understand what the results are likely to be in terms of system behavior.

SSM can register to receive data change notifications from a server. Thus, whenever a particular attribute of an object maintained by the server changes its value, SSM can be sent a notification message containing information about the change. SSM uses this ability to keep any object attribute values that are displayed in a window up-to-date during system operation.

When an SSM window is opened to display the data of a particular HPSS object, the registration operations for data change notifications are performed automatically. Corresponding deregistration operations are also performed when the window is closed. In addition, SSM will automatically register for the operational state of all servers, media, and devices / drives to keep the statuses on the HPSS Health and Status window up to date. When viewing an HPSS managed object window, the **Freeze Display** toggle button can be turned ON to keep it from being updated with data changes and can be turned OFF so that it can receive data changes.

A typical HPSS server maintains a basic server managed object and a server specific managed object. In addition, some servers maintain other managed objects essential to their services. Refer to Section 6.7 for more information on other HPSS managed objects.

6.2 Monitoring HPSS Health and Status

The HPSS Health and Status window is the most essential of the SSM windows for monitoring and controlling the HPSS system. In addition to providing the overall HPSS system data and status, this window also provides a convenient navigational facility to all other SSM windows.

The HPSS Health and Status window allows the SSM users to monitor the overall health and status of the entire HPSS system. Based on the information reported on this window, the user can use the HPSS Alarms and Events window, the HPSS Servers window, the Delogged Messages window, and other SSM windows to further investigate any problems.

Using the HPSS Health and Status Window

The HPSS Health and Status window will be displayed automatically after the SSM logon as shown in Figure 6-1.

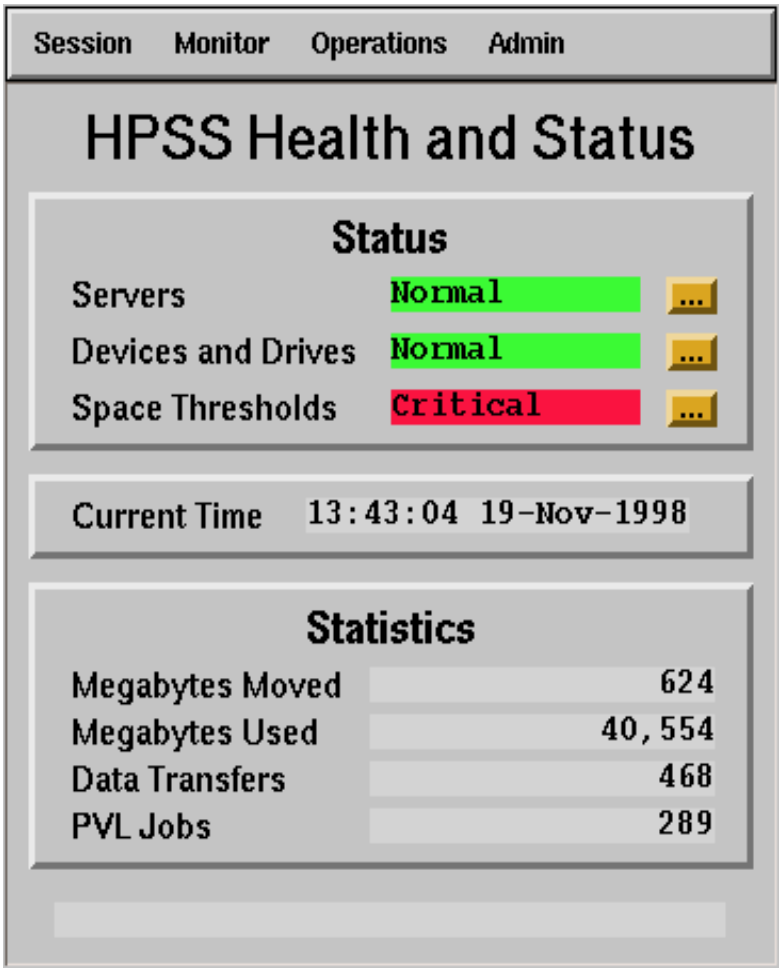


Figure 6-1 HPSS Health and Status Window

Across the top of the HPSS Health and Status window are several options that allow the user to access other SSM windows and management tasks. SSM procedures can be performed by clicking on the appropriate option, which will display a pull-down menu for the option selected. Some procedures may require a higher authorization level than others. Refer to Appendix H for more information on the SSM window access authority.

The options across the top of the HPSS Health and Status window, and their pull-down menu options are:

Session Menu

The following options can be found under the **Session** menu:

- **Preferences**
  - **Alarms and Events**
  - **Servers**
  - **Devices/Drives**
- **Set Keyboard**
  - **DEC Alpha (OSF/1)**
  - **HP 9000 (HP-UX)**
  - **Nighthawk (CX/UX)**
  - **IBM RS/6000 (AIX)**
  - **SCO (Open Desktop)**
  - **SGI**
  - **Sun (Solaris)**
  - **Sun (SunOS)**
- **Show Sammi Environment**
- **View Sammi Errorlog**
- **About Sammi**
- **Log Off Session**
- **Quit Sammi**
- **Broadcast Message**
- **Print SSM Window**

Monitor Menu

The following options can be found under the **Monitor** menu:

- **Alarms and Events**
- **Servers**
- **Storage Classes**
- **PVL Jobs**
- **Devices and Drives**
- **Tape Mount Requests**
- **Tape Check-In Requests**

- **HPSS Objects**
  - **Bitfiles**
  - **Cartridges and Volumes**
  - **Filesets**
  - **Logfile 1**
  - **Logfile 2**
  - **Security**
  - **Specify by SOID**
- **HPSS Message Log**
- **SSM Data Server Statistics**
- **SSM Consoles**

Operations Menu

The following options can be found under the **Operations** menu:

- **Import Volumes**
- **Create SS Resources**
- **Export Volumes**
- **Delete SS Resources**
- **Move Cartridge**
- **Create DFS/HPSS Fileset**
- **Create HPSS-only Fileset**
- **Delete Fileset**
- **Create Junction**
- **Delete Junction**
- **Dismount Drive**
- **Cancel PVL Jobs**
- **Start Migrate/Purge**
- **Start Repack/Reclaim**
- **Start Accounting**

Admin Menu

The following options can be found under the **Admin** menu:

- **Configure HPSS**
  - **Servers**
  - **Logging Policies**
  - **Accounting Policy**
  - **Location Policy**
  - **Migration Policies**
  - **Purge Policies**
  - **Storage Classes**
  - **Storage Hierarchies**
  - **Classes of Service**
  - **File Families**
  - **Devices and Drives**
- **Shut Down**
  - **All Non-SSM Servers**
  - **All of SSM**
  - **SSM Data Server Only**
- **Ping SSM System Manager**

On the upper half of the HPSS Health and Status window are three status fields that represent the aggregate status of the HPSS system. These fields are:

1. **Servers**. This field displays the aggregate statuses reported to SSM by the HPSS servers.
2. **Devices/Drives**. This field displays the aggregate statuses reported to SSM for all configured devices and drives.
3. **Space Thresholds**. This field displays the aggregate statuses reported to SSM for storage space thresholds. The types of space thresholds reported are:
   - Storage Class space used thresholds
   - PVR cartridge capacity thresholds
   - Name Server name space thresholds
   - SFS data volume space thresholds

For the **Servers** and **Devices and Drives** fields, possible status values in increasing order of severity are:

- Normal: No problem reported.
- Unknown: SSM cannot determine the true status due to communication problems or other difficulties.
- Suspect: There may or may not be a problem.

- Minor: A problem is encountered by the server, but it does not significantly affect the services.
- Major: A problem is encountered by the server that may degrade the services.
- Critical: A problem is encountered by the server that may disrupt services unless the problem is resolved.

For the **Space Thresholds** field, the possible status values are:

- OK: No problem reported.
- Warning: A threshold has been reported as being over its warning limit.
- Critical: A threshold has been reported as being over its critical limit.

The statuses displayed in these fields reflect the highest severity encountered by the entities. As problems are resolved or returned to normal, the status fields will automatically reflect the changes.

In addition to the descriptive text values describing the statuses, these fields are displayed with the following background colors to reflect the status of the resources:

- Red: Major and Critical problems
- Yellow: Unknown, Suspect, Minor and Warning problems
- Green: Normal, no problem

When there are errors associated with these resources, press the **...** button to the right of the field to obtain a pop-up selection list of the resources with errors. You can select one of these entities to pursue any possible problems that are indicated by the status values. Note that if the controlling server is down or SSM does not have a connection with the server, the resource object will not be available for viewing.

The lower half of the HPSS Health and Status window displays several data statistics concerning the number of bytes moved, number of bytes used, number of data transfers, and current jobs in the system. The number of bytes moved and number of data transfers indicate the data accumulated by the Movers since startup or data reset. At a glance, these numbers provide an indication of the overall health of the HPSS system.

Normal SSM procedure is to automatically have the HPSS Health and Status window come up immediately after the user logs on. If the HPSS Health and Status window does not appear, an installation or an SSM user configuration problem may exist.

The HPSS Health and Status window is meant to be present during an SSM session. The user will not be able to dismiss it; however, it can be changed to an icon.

6.3 Managing HPSS Servers

The configured HPSS servers can be managed and controlled through the HPSS Servers window (Figure 6-2). From this window, an HPSS server can be started up, shut down, halted, reinitialized, and notified of repair. Once the server is up and running, SSM monitors and reports the server states and statuses in the HPSS Servers window and the HPSS Health and Status window. In

addition, the server information can be viewed and updated through the HPSS Servers window and other SSM information windows.

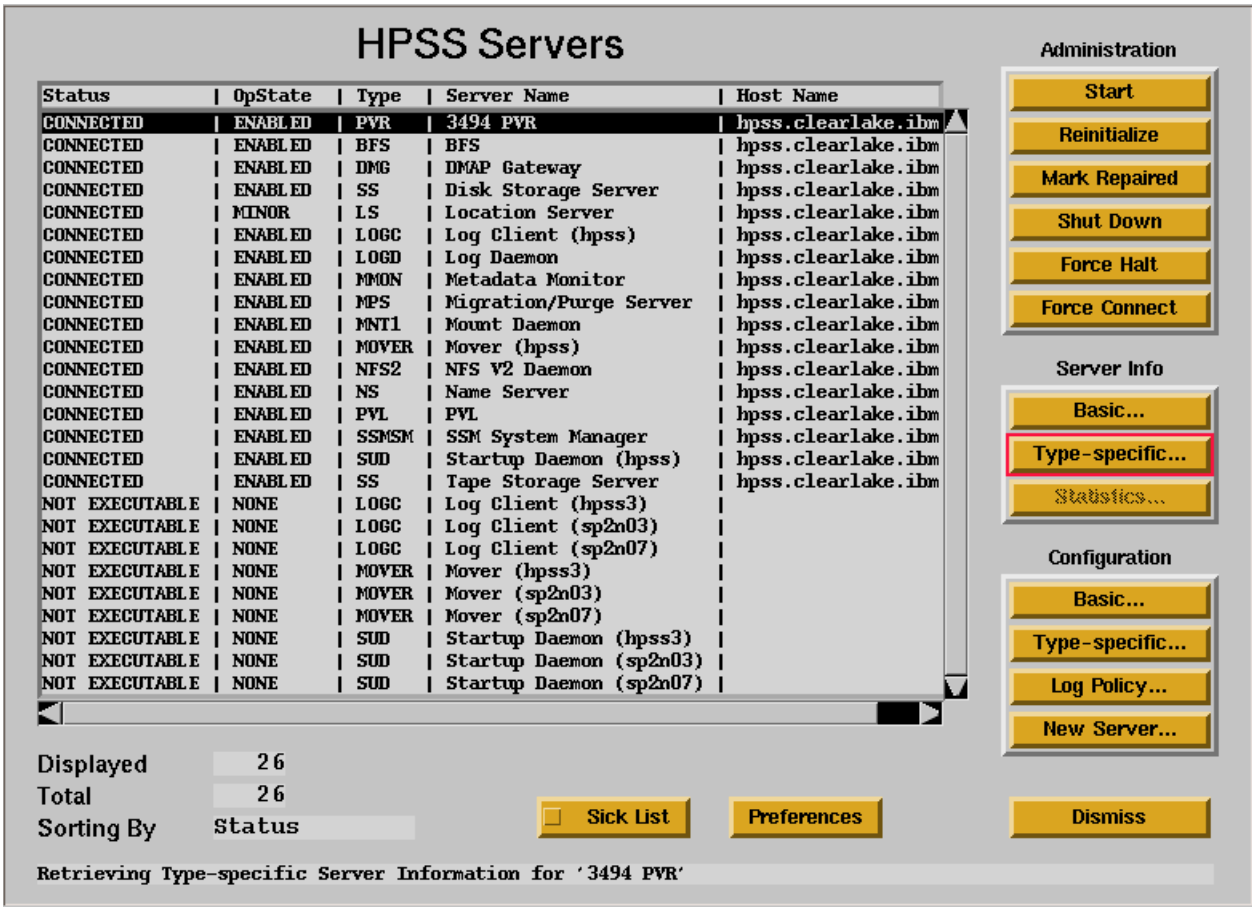


Figure 6-2 HPSS Servers Window

Using the HPSS Servers Window

From the HPSS Health and Status window (Figure 6-1), click on the **Monitor** menu and select the **Servers** option.

The HPSS Servers window will be displayed, appearing initially as shown in Figure 6-2. The server entries displayed on the window can be sorted by each column category. To sort the server entries by status, for example, click on the **Status** column title. The actual display can vary greatly by the setting of window "preferences". Preferences can be used to select which columns of the table and which servers are visible. Preferences can also be saved and automatically reloaded by SSM when new sessions are started (see Appendix H).

To set preferences for the HPSS Servers window, click the **Preferences** button on the window. Or, from the HPSS Health and Status window, click on the **Session** menu, select the **Preferences** option, and then select **Servers**. The HPSS Server List Preferences window is displayed as shown in Figure

6-3. Refer to the window's help file for more information on setting, saving, and reloading the preferences.

For some operations that can be initiated from this window, one or more servers can be selected. Other operations allow only one server to be selected at a time.

Clicking the left mouse button on a server will select that server and deselect any previously-selected servers. Holding down the **Control** key while clicking will add the server to the set of selected servers (previous selections will not be deselected). Holding down the **Shift** key while clicking will select all servers between the current server (the one being clicked) and the last previously-selected server.

Refer to the window's help file for more information on the individual fields as well as the supported operations available from the window.

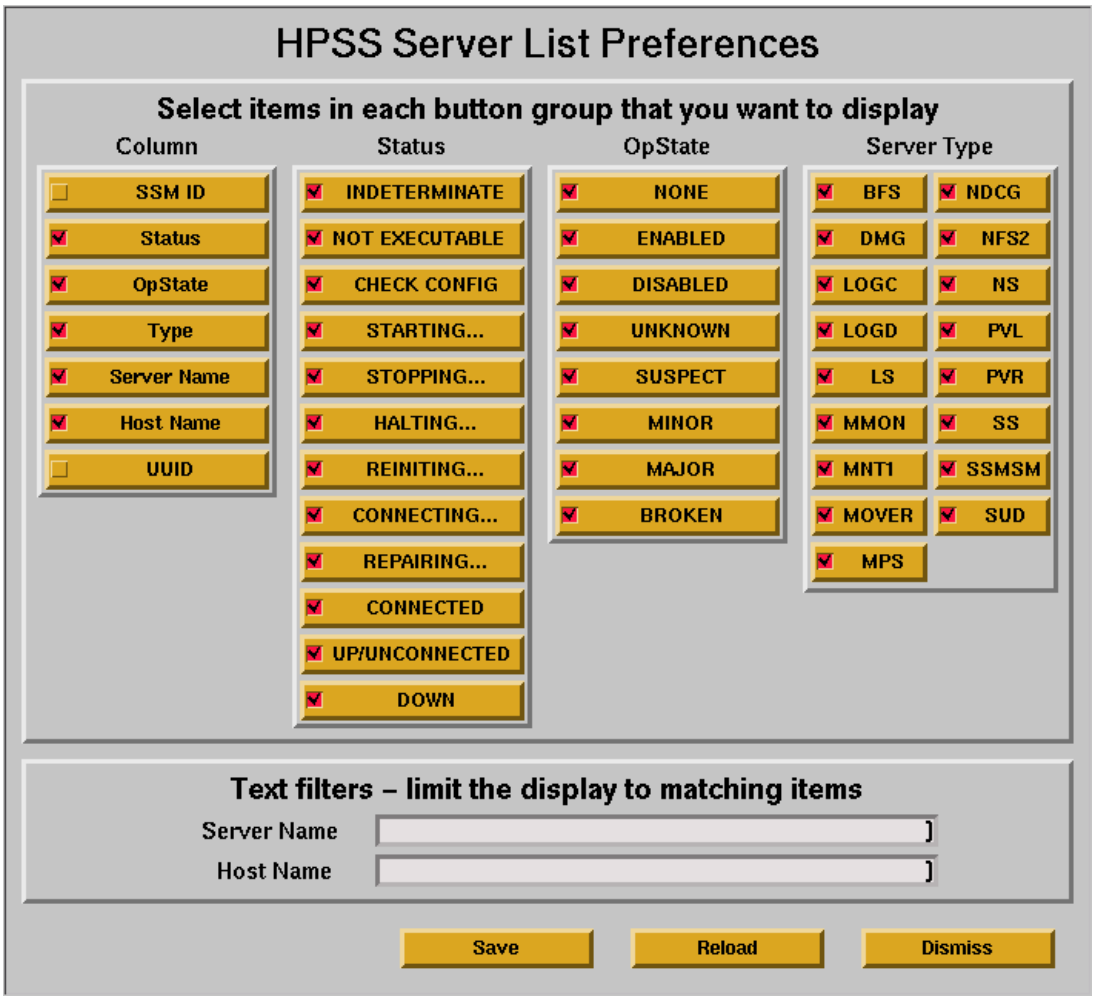


Figure 6-3 HPSS Server List Preferences Window

6.3.1 Monitoring HPSS Servers

An executing HPSS server should be monitored to ensure that it remains running to provide services to HPSS. SSM monitors each executable server and informs the SSM user of the server's abnormal termination or error conditions. The subsections in this section describe how to monitor a server's state and status and how to monitor a server's managed objects to diagnose and resolve problems.

6.3.1.1 Monitor Server Status

As soon as a server is configured, SSM starts to monitor its execution status. If the server is up and running, SSM establishes a connection with the server. SSM reports the server execution and connection status in the Status field of the HPSS Servers window as follows:

- CONNECTED. Server is up and running and communicating normally with SSM.
- UP/UNCONNECTED. Server is up and running (according to the Startup Daemon) but SSM cannon connected to it. Server cannot be completely controlled and monitored through SSM.
- DOWN. Server is down. SSM can be used to start up the server.
- INDETERMINATE. The server's state cannot be determined by SSM and the Startup Daemon is either not running or not connected to SSM.
- CHECK CONFIG. SSM detected an incomplete or inconsistent configuration for the server. The server's configuration should be carefully reviewed to see that it is correct and complete.
- NOT EXECUTABLE. The server is configured as non-executable. SSM will not monitor the server's status.

In addition to the above status values, the Status field also reports the transient status for the server as the result of the user request on the server as follows:

- STARTING...The server is being started.
- STOPPING...The server is being shut down.
- HALTING...The server is being halted.
- REINITING...The server is reinitializing.
- CONNECTING...SSM is trying to establish a connection to the server.
- REPAIRING...The server is repairing its states and statuses.

A server that is configured to execute and is running should have a CONNECTED status. If its status is anything other than CONNECTED (excluding the transient status values), the following actions should be taken:

1. **The server status is UP/UNCONNECTED**. Monitor the server status closely for a few minutes. SSM will periodically try to establish a connection with the server. The Force Connect feature described in Section 6.3.10 can be used to speed up the process.

2. **The server status is DOWN**. Use the server start up feature described in Section 6.3.2 to start up the server. The Startup Daemon will ensure that only one instance of the server is executing.

3. **The server status is INDETERMINATE**. Verify whether the server is running. If the server is not running, start it up. If the server is running, ensure that the Startup Daemon configured for the same node is running and has a connection to SSM. If the Startup Daemon is not running, start it up using the **/etc/rc.hpss** script. Otherwise, use the **Force Connect** operation described in Section 6.3.10 to establish the connections for the server and the Startup Daemon. If this does not correct the server's status, review the HPSS Alarms and Events window to search for problems that the server and SSM may have reported. In addition, delog the HPSS logs for the server's and SSM's log messages to help determine the problems.

SSM will continuously monitor the servers' execution and connection status and update the Status fields when there are any changes.

If a server is configured to execute and is not running, SSM will report it as a an error. Therefore, if a server is not intended to run for an extended period, its Executable flag should be set to OFF. SSM will stop monitoring the server and will not report the server-not-running condition as a critical error. This will also help reduce the work load for the SSM servers.

6.3.1.2 Monitor Server Operational State

An HPSS server that is running and connected to SSM will report any error conditions that it is experiencing by setting its operational state to an error value. SSM reports this information in the **OpState** field on the HPSS Servers window as follows:

- ENABLED. The server operates normally.
- DISABLED. The server is not operational, usually due to a shutdown / halt request.
- SUSPECT. The server may have a problem.
- MINOR. The server encountered a problem that does not seriously affect the HPSS operation.
- MAJOR. The server encountered a problem that may seriously impact the overall HPSS operation.
- BROKEN. The server encountered a critical error and shut itself down.
- UNKNOWN. The server's state is unknown to SSM because SSM cannot communicate with it, and the server has not set its state to DISABLED or BROKEN. The server may or may not be running.

- NONE. The server state is not normally obtainable (usually applicable to a non-executable server).

SSM will continuously monitor the server's operational state and update the OpState field in the HPSS Servers window when there are any changes. When this field indicates that there is an error, the user should select the entry of the server with the error and click on the **Server Info...** button to view the server's managed object window for a possible indication of the error if SSM can communicate with the server.

6.3.1.3 Monitor the Server Information

A server that is running and connected to SSM will allow the SSM user to view and update its information. This section describes the server execution statuses and configuration information. Other information maintained by the servers are described in Section 6.7.

A typical HPSS server allows the SSM users to control its execution and monitor its server-related data through the Basic Server Information window and the server specific information window. These windows are described in the following subsections.

Using the Basic Server Information Window

Most HPSS servers allow the SSM user to view the server's states and statuses as well as changing its Administrative State to lock its service, shut it down, and repair its states and statuses.

On a *normal* condition, the server states and statuses reported on the Server Information window are as follows:

- Administrative State: Unlocked
- Operational State: Enabled
- Usage State: Active
- Execution State: Active
- Service Status: Normal
- Software Status: Normal
- Hardware Status: Normal
- Communication Status: Normal
- Security Status: Normal

However, when the server is experiencing errors or encountering abnormal conditions, it will change the appropriate states and statuses to error values, notify SSM of the changes, and issue an alarm to SSM. Refer to Section 6.9.2 for more information on how to track a problem using the HPSS alarms and the HPSS logging services.

From the HPSS Servers window (Figure 6-2) select the appropriate server entry and then click on the **Basic...** button from the **Server Info** button group. The Basic Server Information window is displayed as shown in Figure 6-4. Refer to the window's help file for more information on the individual fields as well as the supported operations available from the window.

The Startup Daemon and the SSM servers do not support the Basic Server Information window.

If the server's operational state is BROKEN or UNKNOWN, the Basic Server Information window will not be available for viewing.

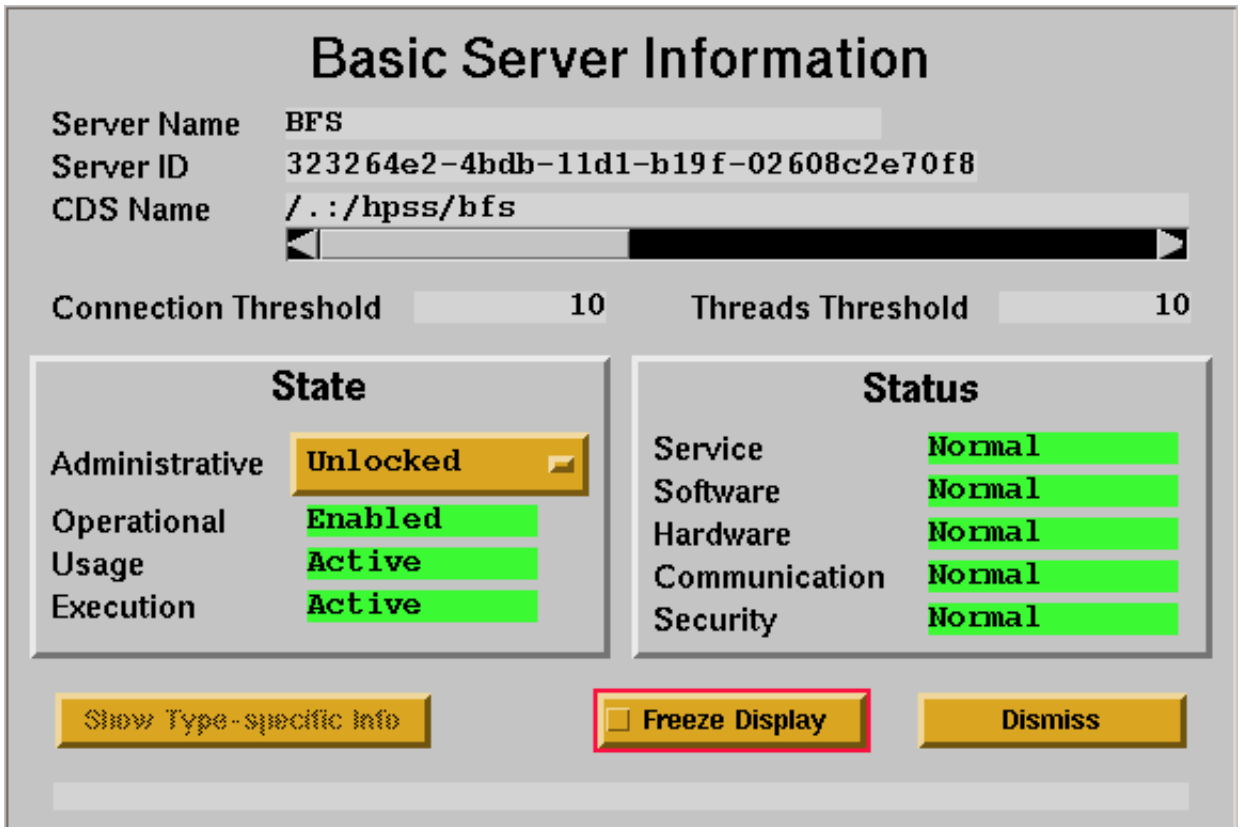


Figure 6-4 Server Information Window

Using the Server Specific Information Windows

The majority of the HPSS servers also allow the SSM user to view and change their server specific data through the SSM windows. A server specific information window displays all the essential data maintained by the server. A typical server allows authorized users to change the value of the fields and use them immediately in the current execution.

There are server specific information windows for the following servers:

- Name Server
- Storage Server
- Migration / Purge Server
- DMAP Gateway
- Physical Volume Library
- Physical Volume Repository

- Mover
- Metadata Monitor

To display a server specific information window, from the Basic Server Information window (Figure 6-4), click on the **Show Type-specific Info** button. The server specific information window will be displayed. This windows can also be brought up from the HPSS Servers window by selecting the appropriate server entry and then clicking on the **Type-specific...** button in the **Server Info** button group. An example of one of these windows is shown in Figure 6-5. A unique window is used to display the data for each type of HPSS server; Figure 6-5 shows the Name Server Information window. Refer to the window's help file for more information on the individual fields as well as the supported operations available from the window.

Any changes made on a server's information window will be effective only during the server's current execution. To make the changes permanent, make the identical changes on the server's configuration window (see Chapter 5).

If the server's operational state is BROKEN or *UNKNOWN, the Specific Server Information window will not be available for viewing.*

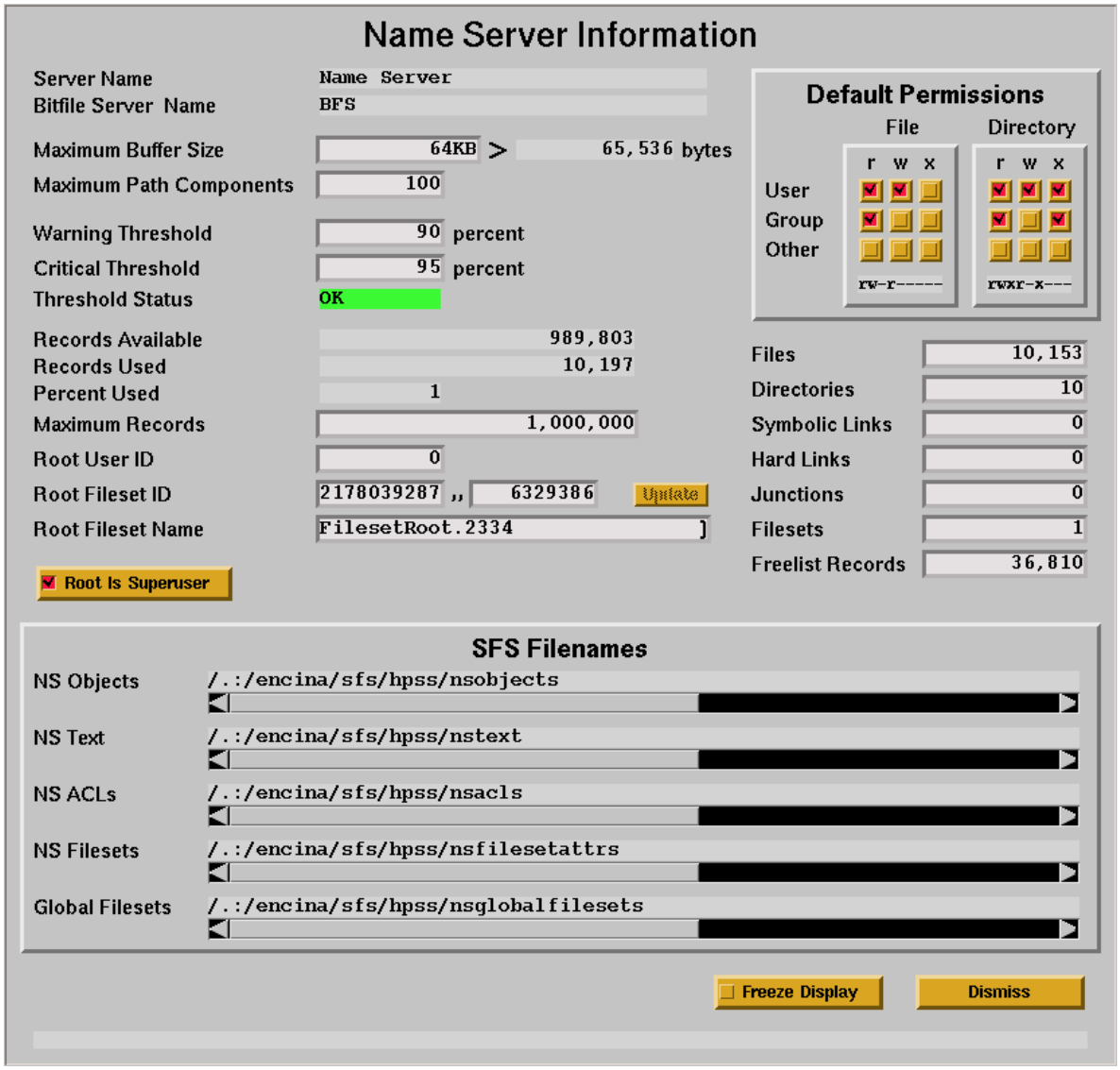


Figure 6-5 Name Server Information Window

6.3.2 *Starting HPSS Servers*

One or more HPSS servers can be started by the same request. SSM forwards the server's configuration data to the appropriate HPSS Startup Daemon. The Startup Daemon invokes the server with the appropriate input arguments. After the server is started, SSM attempts to establish a connection with the server and reports it in the Status field on the HPSS Servers window.

Before starting a server, ensure that DCE and Encina are up and running, the Startup Daemon on the server's node is up and running and has a connection with SSM, and the server is already configured and its **Executable** flag is set.

To initiate the request, select the desired server(s) displayed on the HPSS Servers window and click on the **Start** button. SSM will confirm the request through a pop-up window. Verify the result of the request in the message area on the HPSS Servers window. In addition, monitor the HPSS Alarms and Events window for the "Server Initialized" event.

The Startup Daemon allows only one instance of a configured server to be brought up at a time. If a server is already running, the subsequent startup request for that server will fail.

To be able to receive alarms and events associated with the server start up, it is recommended that the Log Daemon and all Log Clients are brought up first.

The HPSS Startup Daemon(s) can not be started from this window. Refer to Section 5.9 for more information on how to invoke a Startup Daemon.

The SSM System Manager cannot be started from this window. Refer to Section 5.2.1 for more information on starting up the System Manager.

Administrators of sites using IBM robotics should review Appendix J for vendor-specific information on any IBM robotics startup procedures.

Administrators of sites using StorageTek libraries should review Appendix K for vendor-specific information on any StorageTek startup procedures.

6.3.3 Automatic Server Restart

The HPSS servers may be configured to automatically restart. A restart count is contained in each basic server configuration file. Table 5-1 defines the basic server configuration variables, including the **Auto Restart Count**. This count may be used to define the number of times a failed server will be automatically restarted without any administrative action. A failure is defined as a server terminating abnormally. An abnormal termination is the result of the server exiting with a nonzero return code, or exiting as a result of a signal other than **SIGTERM** or **SIGKILL**. Termination due to an administrative **HALT** or **SHUTDOWN** is not considered to be an abnormal termination and a server will not be automatically restarted under these conditions.

The following **Auto Restart Count** values may be set:

0: Do not restart

-1: Infinite restart attempts

n: Restart a failed server n time (where n is a positive integer > 0)

After the restart count is exhausted, the Startup Daemon will not attempt to restart the server. If the server executes for more than 1 hour without failing, the number of failures will be set back to 0.

6.3.4 Shutting Down an HPSS Server

One or more HPSS servers can be shut down by the same request. SSM requests each server to shut down gracefully. The server will attempt to complete the current tasks and inform SSM of its termination prior to exiting. After the server is shut down, SSM reports the server's status in the Status field on the HPSS Servers window.

To initiate the request, select the desired server(s) displayed on the HPSS Servers window and click on the **Shut Down** button. SSM will confirm the request through a pop-up window. The SSM user may wish to verify the result of the request in the message area on the HPSS Servers window as well as to monitor the HPSS Alarms and Events window for the "Server Terminated" event.

The HPSS Startup Daemon(s) and the SSM System Manager cannot be shut down from the HPSS Servers window. Use the UNIX kill -9 command to shut down the Startup Daemon. The System Manager can either be killed with the kill command or by selecting the Shut Down -> All of SSM option from the Admin menu on the HPSS Health and Status window. Note that the Shutdown SSM menu option will also terminate the SSM Data Server and any logged on SSM sessions.

Since the shut down request is never forced, the server may not terminate immediately. Monitor the HPSS Alarms and Events window for possible notifications of delay from the server. In addition, monitor the HPSS Servers window for the server's shut down status. If an immediate shut down is desired, use the Halt Server operation described in Section 6.3.5.

SSM must have a connection to the server or the shut down request will fail.

6.3.5 Halting an HPSS Server

One or more HPSS servers can be halted by the same request. SSM informs each server that it is requested to halt immediately. The server will attempt to inform SSM of its termination and exit immediately. SSM will also request the Startup Daemon to issue a **kill -9** command to a server if it does not halt immediately. After the server is halted, SSM reports the server's status in the Status field on the HPSS Servers window.

To initiate the request, select the desired server(s) displayed on the HPSS Servers window and click on the **Force Halt** button. SSM will confirm the request through a pop-up window. Verify the result of the request in the message area on the HPSS Servers window. In addition, monitor the HPSS Alarms and Events window for the "Server Halted" alarm.

The HPSS Startup Daemon(s) and the SSM System Manager cannot be halted from the HPSS Servers window. If attempted, a pop-up error window will be displayed to inform the user of the invalid request.

SSM must have a connection either to the server or the appropriate Startup Daemon or the halt request will fail.

6.3.6 Reinitializing an HPSS Server

One or more servers can be reinitialized by each request. SSM will inform the server that it is requested to reinitialize with new configuration data. The server will reread its configuration and reinitialize.

To initiate the request, select the desired server displayed on the HPSS Servers window and click on the **Reinitialize** button. SSM will confirm the request through a pop-up window. Verify the result of the request in the message area on the HPSS Servers window. In addition, monitor the HPSS Alarms and Events window for the "Server Reinitialized" event.

Only the Log Client, Log Daemon, Location Server, and the Metadata Monitor support reinitialization. If reinitialization is requested for other servers, a pop-up error window will be displayed to inform the user of

the invalid request. Servers that do not support reinitialization must be shut down and restarted to use the modified configuration data.

6.3.7 Repairing an HPSS Server

One or more server can be repaired by each request. SSM will inform the server that it is requested to clear its error states and statuses. The server will reset its states and statuses to "normal."

To initiate the request, select the desired server displayed on the HPSS Servers window and click on the **Repair** button. SSM will confirm the request through a pop-up window. Verify the result of the request in the message area on the HPSS Servers window. In addition, monitor the HPSS Alarms and Events window for the "Server Repaired" event.

Clicking on the Repair button issues an instruction to the server to reset its state and status values. It does not correct any underlying problems that might still exist. Rather, it is a means by which the administrator can notify the server that the underlying problem has been addressed. It is used for problems that the server cannot fix by itself or for problems that have been fixed unbeknownst to the server, such as a drive being offline or a tape getting stuck in a drive.

6.3.8 Shutting Down All HPSS Servers

All HPSS servers (except the SSM servers and the Startup Daemons) can be shut down gracefully by selecting the **Shut Down -> Shut Down All Non-SSM Servers** option from the **Admin** menu on the HPSS Health and Status window. SSM will process this request similarly to the Shutdown Server request issued from the HPSS Servers window.

This feature is useful when the entire HPSS system must be shut down, leaving SSM up to monitor the server shut down process. When the servers are down, the SSM servers and sessions can be shut down by selecting the **Shut Down -> All of SSM** option from the **Admin** menu on the HPSS Health and Status window.

6.3.9 Shutting Down the SSM Servers

The SSM servers can be shut down by selecting the **Shut Down** option from the **Admin** menu on the HPSS Health and Status window. Selecting the **All of SSM** option under **Shut Down** will shut down the SSM System Manager, the SSM Data Server, and all SSM sessions. Selecting the **SSM Data Server Only** option will shut down the Data Server and the SSM sessions, but leave the System Manager running.

Choosing either option will pop up a confirmation window which displays the number of other SSM consoles attached to the SSM servers. If the number is greater than zero, then proceeding with the shutdown will also kill other SSM sessions. If this is not desirable, the shutdown request can be cancelled.

As the Data Server exits, it pops up a notification window on each logged on SSM console. The console's user can click the button on this window to finish shutting down the SSM session.

6.3.10 Forcing an SSM Connection

When the status of the server's connection with SSM is UP / UNCONNECTED, select the server and then click on the **Force Connect** button on the HPSS Servers window. SSM will attempt to establish the connection with the server and reflect the latest data in the Status field on the HPSS Servers window.

The SSM System Manager periodically retries the connection with all the servers that are configured to execute. This request may be used only to speed up the connection request.

6.3.11 Ping the SSM System Manager

For various reasons, the Data Server may be unable to connect with the System Manager. As a result, the currently displayed SSM windows may look stale and may behave sluggishly. Select the **Ping SSM System Manager** option from the **Admin** menu on the HPSS Health and Status window to request the Data Server to reconnect after verifying that the System Manager is up and running.

This option should be done when the SSM Data Server notifies the SSM user that it cannot establish a connection with the System Manager or when the System Manager is being recycled.

The SSM Data Server automatically tries to ping the System Manger at intervals when it thinks there is a problem, but like the Force Connect request, this can force it to take place immediately.

6.3.12 Configuring a New Server

A new HPSS server can be configured at any time. To add a new server, perform the following steps:

1. **Create a new basic server configuration entry**. Refer to Section 5.3 for more information on creating a new basic server configuration entry.

2. **Create a server specific configuration entry**. Refer to Section 5.6 for more information on creating a new server specific configuration entry.

3. **Create a Log Policy for the server**. Refer to Section 5.4.4 for more information on creating a new log policy.

After the server is configured, it can be started up through the SSM HPSS Servers window. Refer to Section 6.3.2 for more information on starting up an HPSS server.

6.3.13 Deleting a Server Configuration

A server's configuration can be removed when the server is no longer needed. However, care should be taken to ensure that all objects created by the server are deleted properly. In addition, all references to the server must be removed from other HPSS configuration entries.

A server's configuration should only be removed when it is no longer needed. Otherwise, the configuration should be updated instead of deleted to reduce the possibility of introducing configuration errors into HPSS.

There must be at least one Name Server, Bitfile Server, Tape and/or Disk Storage Server, PVL, and Mover configuration entry in order for HPSS to be operational.

It is very important that the server's configuration entries be deleted in the order shown below. If the general configuration entry is deleted before the specific configuration entry, SSM will no longer be able to find and delete the specific configuration entry.

To delete a server, perform the following steps in the order shown:

1. **Delete all objects created by the server**.

    - If deleting a Name Server configuration, ensure that all objects created by the Name Server (e.g., directories, files, hard links, and symbolic links) have been deleted. Since a directory can only be deleted if it is empty, all objects within a given directory must be deleted first. Note that in a system configured with a single Name Server, deleting all the objects created by the Name Server will result in an empty name space.

    - If deleting a Bitfile Server configuration, ensure that all the files in the HPSS system are deleted. This implies that all of the BFS metadata files associated with the mapping of bitfiles are empty. Effectively, the BFS configuration should not be deleted unless the HPSS system is to be disassembled.

    - If deleting a Storage Server configuration, ensure that all created objects (e.g., physical volumes, virtual volumes, storage segments, and storage maps have been deleted. This implies that all bitfile segments stored on any of these objects have been deleted.

    - If deleting a PVL configuration, all imported cartridges must first be exported.

    - If deleting a PVR configuration, all injected cartridges must either be exported or moved to another PVR.

2. **Remove references to the server from other configurations**.

    - If deleting a Name Server configuration, update the **Name Server** field in the appropriate Bitfile Server Configuration window (Figure 5-16).

    - If deleting a Bitfile Server configuration, update the **Bitfile Server** field in the Name Server Configuration window (Figure 5-15) and the HPSS Migration / Purge Server Configuration window (Figure 5-19).

    - If deleting a Storage Server configuration, update the **Storage Servers** field in the appropriate Bitfile Server Configuration window (Figure 5-16) and the HPSS Migration / Purge Server Configuration window (Figure 5-19).

    - If deleting the PVL configuration, update the **PVL Server** field in the appropriate PVR Configuration window (Figure 5-22) and the Storage Server Configuration window (Figure 5-17).

    - If deleting a PVR configuration, update the **PVR Servers** field in the PVL Configuration window (Figure 5-21) and the Configure Mover Device and PVL Drive Configuration window (Figure 5-36).

    - If deleting a Migration / Purge Server configuration, update the **Migration/Purge Server** field in the Bitfile Server Configuration window (Figure 5-16) and the appropriate HPSS Storage Class Configuration window (Figure 5-10).

- If deleting a Mover configuration, update the appropriate Device and Drive configuration using the Configure Mover Device and PVL Drive window (Figure 5-35) with a new Mover name.

- If deleting the SSM configuration, update the **SSM Server ID** field in the general Server Configuration window (Figure 5-4) for other servers. Ensure that the modified configuration entries contain the newly configured SSM Server ID.

After the affected configurations are modified, reinitialize (if supported) or recycle the appropriate HPSS servers.

3. **Ensure that the server is not running**.

4. **Delete the server's logging policy, if exists**.

5. **Delete UNIX files created/used by the server, if no longer needed**.

   - If deleting the MPS, delete the MPS reports.

   - If deleting the NFS Daemons, delete the UNIX files configured for these daemons. They are usually created in the **/var/hpss/nfs** directory.

   - If deleting the Startup Daemon, delete all server lock files in the **/var/hpss/tmp** directory. The name of these files always begin with **hpssd.**

   - If deleting the Mover, delete Mover's socket files in **/var/hpss/tmp** directory.

   - If deleting DFS HDM server, delete files in **/var/hpss/hdm/<hdm_id>** directory.

6. **Delete the server specific configuration**.

Before deleting the specific configuration, obtain the name of the SFS file(s) used by the server. This information will later be used to delete the SFS file(s) that are no longer used. When deleting these SFS files, ensure that they are not being shared by other servers.

From the HPSS Servers window, select the entry for the server being deleted and click on the **Type-specific** button in the **Configuration** button group. The server specific configuration window will be displayed. Click on the **Delete** button in this window to delete the configuration entry.

7. **Delete the basic server configuration**.

Before deleting the basic configuration entry, obtain the name of the SFS specific configuration file used by the server. This information will later be used to delete the specific configuration file, if it is empty.

From the HPSS Servers window, select the entry for the server being deleted and click on the **Basic...** button from the **Configuration** button group. The Basic Server Configuration window will be displayed. Click on the **Delete** button in this window to delete the configuration entry.

8. **Delete SFS files**. After a server's configuration has been deleted, the no longer used SFS files configured for the server must also be deleted using the **managesfs** utility. Also, there may be an empty server specific configuration file if it is not shared with other servers. The administrator needs to determine if the specific configuration file is empty before deleting

it. Refer to Appendix I for more information on how to invoke the **managesfs** utility to delete the SFS files.

6.3.14 Changing a Server Configuration

A server configuration can be modified whenever necessary. Bring up the server's configuration windows from the HPSS Servers window, modify the desired configuration fields, and then click on the **Update** button.

For the modified configuration data to be used, the server must be reinitialized, if supported, or restarted.

6.4 Managing HPSS Storage Space

Managing the HPSS storage space consists of monitoring the existing storage space, creating additional storage space when necessary, and deleting the existing storage space if needed. These storage management procedures are described in the following subsections.

6.4.1 Creating HPSS Storage Space

Adding storage space in HPSS is done in two distinct phases: import and create. The first phase, import, involves physically introducing the tape cartridges and the disk volumes into the PVL and labeling them with HPSS volume labels. The second phase, create, defines the Storage Server data structures that will describe the data stored on the imported volumes.

6.4.1.1 Importing Volumes into HPSS

To import a tape cartridge into HPSS, the administrator must be familiar with the operation of the PVR to which the cartridge will be added because the physical cartridge injection process differs among the three PVRs supported by HPSS.

- STK 4400 robotic devices: In the case of imports into the STK 4400 robotic devices, the cartridges to be imported need to be physically injected into the robots before the import is attempted. Failure to do this will result in the failure to import those cartridges that have not been pre-injected into the robots.

- IBM 349x robotic device: When importing cartridges into an IBM 349x robotic device, the cartridges must be entered into the robot before any HPSS import operations are performed. Refer to Appendix J for more information.

- Storage Tek: Administrators and operators with StorageTek PVR at their sites should review Appendix K (StorageTek) before attempting cartridge imports. This appendix discusses the STK library and its interaction with HPSS.

- AML: The cartridges to be imported should be placed in the E/I/F ports, which are setup in the **AML_InsertPort.conf** file. From the Import HPSS Media window, specify the cartridges to import. It is unnecessary to issue the "**dasadmin insert**" command to physically insert the cartridges into the library prior to importing.

The procedure for injecting cartridges into a PVR depends on the PVR in question. In the case of operator (hand) mounted PVRs, no injection step is necessary or possible. In the case of the STK 4400 robots, the cartridge(s) to be imported can be entered by:

- Using the enter command of ACSLS (StorageTek's robot software) and putting the cartridge(s) into the Cartridge Access Port (CAP).
- Loading a cartridge into a CAP operating in ACSLS's "hot enter" mode.
- Bulk loading a robot and then performing an ACSLS audit of the slots that have been filled.

We strongly recommend that tape cartridges be imported into a robot that verifies external labels before mounting. This will ensure that a blank tape being labeled (imported) has the appropriate external label. Once labeled, a cartridge can be moved to another PVR, such as an operator (hand) mounted PVR.

If a site generates an internal label on tape volumes prior to importing those volumes into HPSS, then care should be taken that that label has a format that HPSS can process. In particular, if an attempt is made to set the **OwnerID** *field, the first four characters should be set to* "HPSS"*, and the last two characters (the* **OwnerID** *field is fourteen characters in length) to* "00" *- i.e., two ASCII characters each representing the digit zero (ASCII value 48). HPSS uses the last two characters to indicate the side number of the cartridge (in anticipation of supporting cartridges that have multiple physical sides - e.g., optical disk cartridges that contain an* A *and a* B *side). If these two characters are not set correctly, then the import will fail because the side information contained in the internal label will not match the side information (currently always* "00"*) passed in the Import request. If the start of the* **OwnerID** *field is NOT* "HPSS"*, then the Mover will interpret the label as indicating side zero, and the Import will succeed (the volume will be marked as having a* Foreign *label).*

When importing a non-removable disk volume into HPSS, the raw disk must already be defined in the native system. For disk volumes, the import simply involves labeling the volumes with the HPSS labels.

The volumes can be imported into HPSS using the Import HPSS Media window. Once the volumes are imported, the HPSS storage space can be created. Refer to Section 6.4.1.2 for more information on how to create and manage the HPSS storage space.

The import step must be done for both disk and tape volumes. Ensure that the PVL, PVR, and appropriate Movers are up and running before initiating a requesting for the import operation. Also ensure that SSM is connected to the PVL.

From the HPSS Health and Status window (Figure 6-1), select the **Operations** menu and click on the **Import Volumes** option. The Import HPSS Media window will be displayed as shown in Figure 6-6 for disk import. Figure 6-7 shown the window to be used for tape import. First, select the type of media to be imported by clicking on the **Tape** or **Disk** button, as desired. Then enter the appropriate information and then click on the **Import** button to start the import process.

During the import phase, requests for cartridge import(s) will be issued to the PVL. Multiple import requests can be issued concurrently to the PVL by setting the **Maximum Drives** field on the Import HPSS Media screen. This field determines how many separate threads are created to handle the import. The total number of cartridges are divided as evenly as possible among the threads for processing. Depending on the total number of drives in the system and the workload from other requests, as far as possible, all the threads are assigned a drive and allowed to process their cartridges concurrently.

When imports to an operator PVR are performed, each request to mount a cartridge is serialized by the Operator PVR. This serialization of import requests is necessary to ensure proper labeling of previously unlabeled tapes. Any unlabeled cartridge mounted to a drive under control of the operator PVR will be assumed to be the cartridge in question. Because any blank tape that is inserted will be labeled as if it was the one being requested, this operation is not without risk. It is strongly recommended that labeling of cartridges be performed by a robot that can verify the external label of a cartridge before mounting.

Once SSM completes the Import request, it reports the number of volumes imported on the window status field. Failure at any time during the import process will cause SSM to stop processing the request. If the reported numbers are less than the number of volumes to be imported, retry the import request after first addressing the reason for the initial failure. For the retry, the original list of the to-be-imported volumes can be used because the PVL will skip over the imported volumes.

If the tape cartridge labeling occurred within an operator (hand) mounted drive, ensure that the internal and external cartridge labels match. There is no utility in the current release of HPSS to do this. An administrator could create a program to do this using a drive that is not on-line to HPSS.

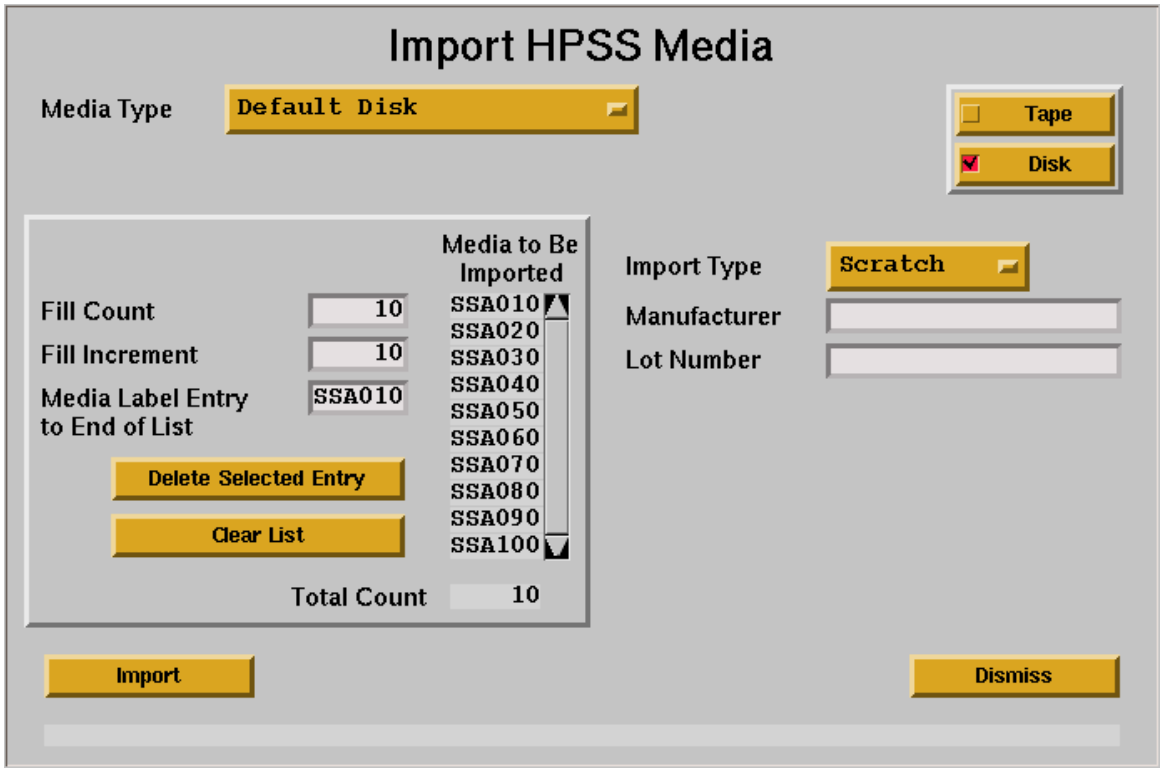


Figure 6-6 Import HPSS Disk Media Window

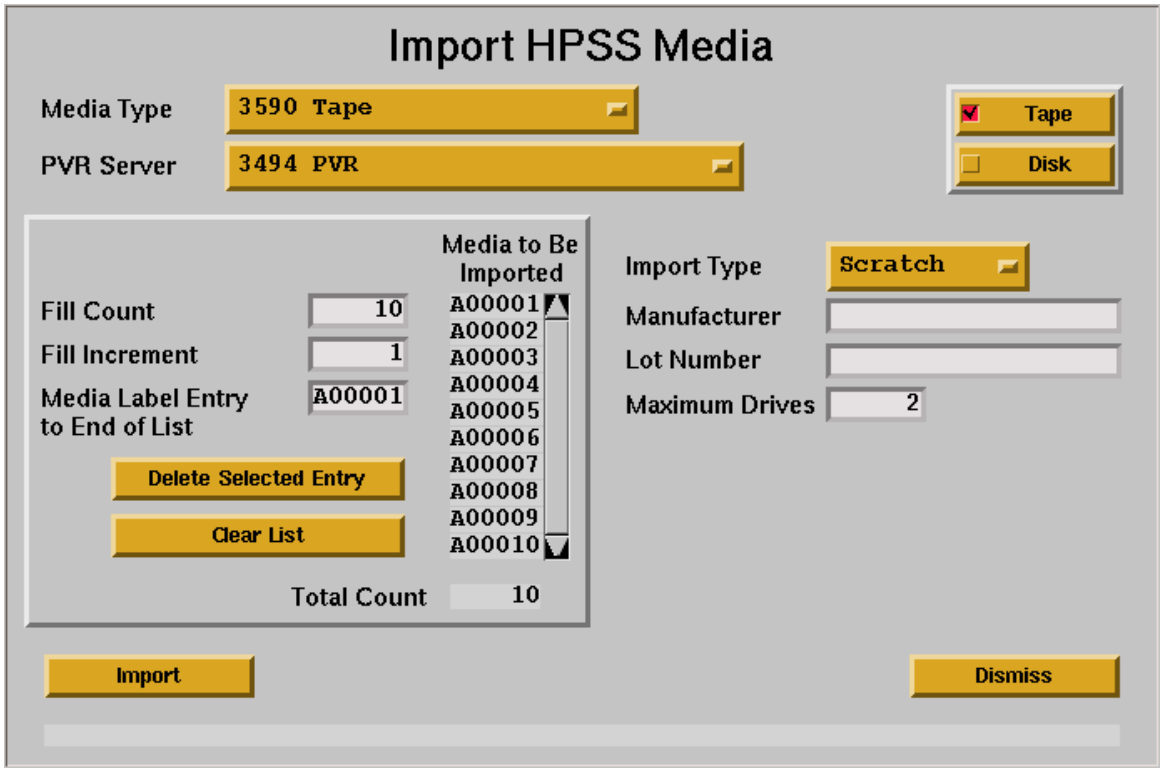


Figure 6-7 Import HPSS Tape Media Window

Import HPSS Media Variables

Table 6-1 lists the fields on the Import HPSS Media window.

Table 6-1 Import HPSS Media Variables

Display Field Name	Description	Acceptable Values	Default Value
Media Type [P]	The type of media to be imported.	Any valid media type from the pop-up list.	Default Tape or Default Disk, based on **Tape** / **Disk** toggle buttons.
PVR Server [P]	The descriptive name of the PVR to import the tape cartridges into.	Any configured PVR name from the pop-up list.	None
	Advice: This field is only used for tape import.		

Table 6-1 Import HPSS Media Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Import Type [P]	The type of import to be performed.	Default, Oversrite Scratch	Default
	Advice: Specifying **Scratch** Import Type, will usually cause an HPSS label to be written onto the media, potentially causing any data already on the media to be lost. Specifying **Overwrite** Import Type will cause the HPSS label to be re-written, if an existing HPSS or Foreign label already exists (provided it is for the same Volume ID) - this allows tapes to be re-written in newer drives (e.g., tapes originally created on a 3590 drive to be reused on a 3590E drive). Based on the Import Type, the import request will be processed depending on how the media is currently labeled. Refer to Tables Table 6-2 and Table 6-3 for more information on selecting the appropriate Import Type.		
Manufacturer	The maker of the media to be imported.	Any valid character string.	None
Lot Number	The manufacturer's lot number of the media.	Any valid character string.	None
Maximum Drives	The maximum number of tape drives to use when running the tape imports.	Any positive integer value.	1
	Advice: This field is only used for tape import.		
Fill Count	The number of media labels to be generated in the **Media to Be Imported** list when a value is typed into the **Media Label Entry to End of List** field.	Any positive integer.	1
Fill Increment	The number by which each automatically generated media label will differ from the previous one, when a value is typed into the **Media Label Entry to End of List** field, and the **Fill Count** is greater than 1.	Any positive integer.	1
Media Label Entry to End of List	The 6-character label of a media volume to be imported. The label is added to the end of the **Media to be Imported** list. If **Fill Count** is greater than 1, multiple media labels are generated using the entered label as a starting point.	Any valid media label.	None

Table 6-1 Import HPSS Media Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Media to be Imported	The list of media labels to be imported.	List of to-be-imported media labels. Generated by SSM.	None
Total Count	The total number of media to be imported.	Count of media labels generated in the **Media to be Imported** list. Calculated by SSM.	None

Selecting Import Type for Tape Cartridges

Table 6-2 lists information for selecting tape import types.

Table 6-2 Tape Import Types

Current Tape Label	Default Import	Overwrite Import	Scratch Import
An ANSI or HPSS label with a correct Volume ID (the Volume ID on the label is as expected by HPSS)	Tape Imported	Label Written, Tape Imported	Tape Imported
An ANSI or HPSS label with an incorrect Volume ID (the Volume ID on the label is different from the Volume ID expected by HPSS)	Tape Not Imported	Tape Not Imported	Tape Not Imported
Random data (e.g., a tar file)	Tape Not Imported	Tape Not Imported	Label Written, Tape Imported
No data (two tapemarks at the start of tape)	Label Written, Tape Imported	Tape Not Imported	Label Written, Tape Imported
Unreadable (e.g., some brand new tapes or a degaussed tape; also possibly a tape written at a different density)	Tape Not Imported	Tape Not Imported	Label Written, Tape Imported

Any existing data on a tape imported into HPSS will be overwritten.

HPSS always attempts to read a label at the beginning of a tape when performing an import. Some tape drives and device drivers will report errors when asked to read unreadable tapes (as described

above). If this happens, manually write two tape marks at the start of the tape and retry the import. Most UNIX systems provide the **mt** command, which can be used to write tape marks.

An HPSS label is basically just an ANSI label. A few characters are changed to identify the tape as having been labeled by HPSS. Both label types are supported by HPSS; tapes already beginning with an ANSI label are not relabeled.

Selecting Import Type for Disk Volumes

Table 6-3 lists information for selecting disk import types.

Table 6-3 Disk Import Types

Current Disk Label	Default Import	Overwrite Import	Scratch Import
An ANSI or HPSS label with a correct Volume ID (the Volume ID on the label is as expected by HPSS)	Label Written, Disk Imported	Label Written, Disk Imported	Label Written, Disk Imported
An ANSI or HPSS label with an incorrect Volume ID (the Volume ID on the label is different from the Volume ID expected by HPSS)	Disk Not Imported	Disk Not Imported	Label Written, Disk Imported
No label	Disk Not Imported	Disk Not Imported	Label Written, Disk Imported

6.4.1.2 Creating the Storage Server Resources

After the media have been imported into HPSS, the Storage Server resources that describe the new storage space must be created using the Create Storage Server Resources window. Once the SS resources are created, HPSS files can be stored on the new media. If a problem develops while creating the SS resources, they can be deleted and created again without having to export the media from HPSS. Refer to Chapter 6 for more information on how to manage the HPSS storage space.

Ensure that the PVL and the appropriate Storage Server(s) are up and running before initiating a request for the create operation. Also ensure that SSM is connected to the appropriate Storage Server.

The storage server data structure creation step must be done for both disk and tape volumes.

From the Health and Status window (Figure 6-1), select the **Operations** menu and click on the **Create SS Resources** option. The Create Storage Server Resources window will be displayed as shown in Figure 6-8. Enter the appropriate information and click on the **Create** button. This action will start the create process.

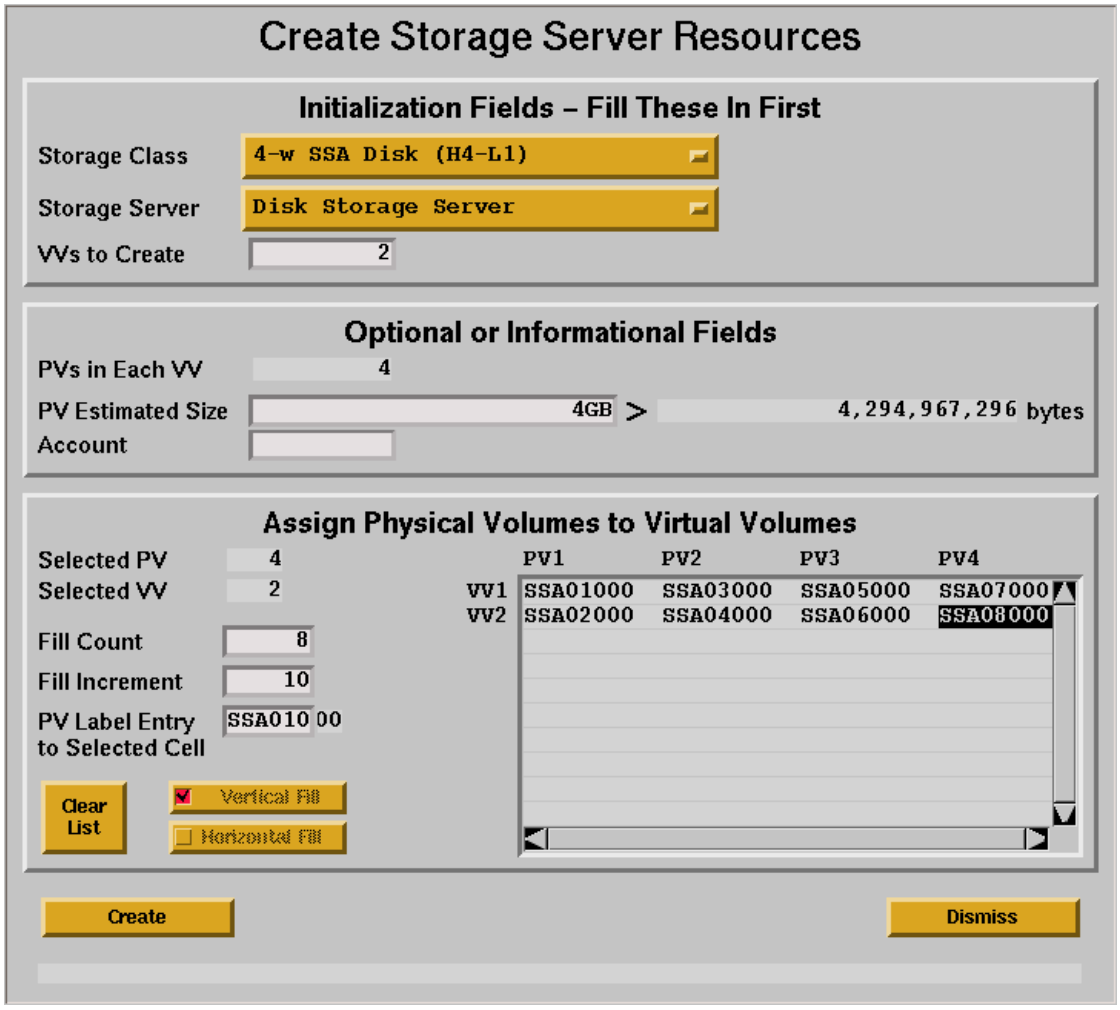


Figure 6-8 Create Storage Server Resources Window

Create Storage Server Resources Variables

Table 6-4 describes the create storage server resources variables.

Table 6-4 Create Storage Server Resources Variables

Display Field Name	Description	Acceptable Values	Default Value
Storage Class [P]	The name of the Storage Class whose characteristics will be used to create the requested virtual volume(s).	Any valid storage class from the popup list.	None.
Storage Server [P]	The name of the Storage Server that will process the create request and in which the space resources will be created.	Any Storage Server names selectable from the pop-up list.	None
	Advice: The **Storage Class** field must be filled in before selecting a Storage Server.		
VVs to Create	The number of virtual volumes to be created.	Any positive integer value between 1 and 1000.	None
PVs per VV	The number of consecutive physical volumes that are to be grouped together in a stripe group to make up a single virtual volume.	This number (the **Stripe Width**) is taken from the characteristics of the selected Storage Class. It is displayed for reference only.	The Stripe Width defined in the selected storage class.
PV Estimated Size	The estimated amount of data that can be written on the physical volume.	Any positive 64-bit integer value. It must be an integer multiple of the **VV Block Size** for the selected storage class, and, for disks, it cannot be greater than 16,384* **VV Block Size** / **Stripe Width**.	The PV Estimated Size defined in the selected storage class.
	Advice: For tape storage classes, this field is displayed for reference only, and cannot be changed. For disk storage classes, a size different than that specified in the storage class can be entered for use in resource creation.		
Account	The accounting identifier associated with the new volumes.	Any valid account ID.	None
Selected PV	The column number of the currently selected cell in the PV-VV table.	Calculated by SSM.	1
Selected VV	The row number of the currently selected cell in the PV-VV table.	Calculated by SSM.	1
Fill Count	The number of physical volumes to be generated in the PV-VV table when a value is typed into the **PV Label Entry to Selected Cell** field.	Any positive integer value between 1 and 256,000.	1

Table 6-4 Create Storage Server Resources Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Fill Increment	The number by which each automatically generated PV label will differ from the previous one, when a value is typed into the **PV Label Entry to Selected Cell** field, and **Fill Count** is greater than 1. The increment is added only to the cartridge label portion of the label (the first six characters), not to the side number portion (the last two characters).	Any positive 32-bit integer value.	1
PV Label Entry to Selected Cell	The label of a physical volume to be inserted into the PV-VV table at the currently selected cell. If **Fill Count** is greater than 1, multiple media labels are generated using the entered label as a starting point. If the **Vertical Fill** button is ON, labels will be filled in column-wise (top to bottom, then left to right). If the **Horizontal Fill** button is ON, labels will be filled in row-wise (left to right, then top to bottom).	Any valid volume name. SSM converts lowercase characters to uppercase.	None
PV-VV Table	A table of virtual volumes to be created, and the physical volumes that will be assigned to each virtual volume. Each entry in the table is a "cell", and any cell can be selected by clicking it with the mouse. The selected cell is highlighted, and is also shown by the Selected PV and Selected VV fields. The Clear List button can be used to empty the table if desired.	List of to-be-created PV names. Generated by SSM.	None

The Storage Server that manages space in the selected PVL will create a descriptive record for each of the new physical volumes (cartridges). All physical volumes created will have identical characteristics (type, estimated size, stripe size, stripe width, block size, blocks between tape marks,

etc.). For disk devices, however, the estimated size for the storage class may be overridden. That is, the disk devices within a storage class do not need to be of the same size.

Virtual volumes are created by grouping the physical volumes together in groups of N, where N is the number of physical volumes per virtual volume (the Stripe Width). The physical volumes are arranged into virtual volumes according to the table on the Create Storage Server Resources window, which the administrator should modify as desired before clicking the Create button. Every physical volume must be placed into a virtual volume before it is usable by HPSS, even if there is only one physical volume per virtual volume. The Storage Server creates a descriptive record for each virtual volume that includes the storage class specified on the Create Storage Server Resources window and the virtual volume block size defined for that storage class. All virtual volumes created during one operation (i.e., as a result of clicking once on the **Create** button from this screen) will have identical characteristics.

The Storage Server creates a storage map for each of the virtual volumes. All information needed to create each map is taken from the physical volume and virtual volume information. By default, each new storage map describes storage space that is *immediately available* for use. Therefore, the storage space created with the Create Storage Server Resources window may be selected by the Storage Server for use immediately after the create operation completes.

Once SSM completes the Create request, it reports the number of physical volumes and virtual volumes created in the window status field. Failure at any time during the create process will cause SSM to stop processing the create request. If the reported numbers are less than the number of volumes to be created, retry the create request after first addressing the reason for the initial failure.

Note that new tape resources are not assigned to a File Family. They are actually assigned to family zero, which the system interprets as no family assignment. Tapes are assigned to file families as they are needed to satisfy requests to write to tape in a given family. You cannot pre-assign tapes to families in Release 4.1.1.

6.4.2 Monitoring the HPSS Storage Space

After the HPSS system is operational, one of the most important tasks for an administrator is to monitor the space usage on the HPSS system. The goal is early detection of any potential space shortages in each active storage class so that corrective action can be taken in a timely manner. Each active storage class is configured with a space-used warning threshold and a critical threshold to inform the SSM user when the free space in a storage class is running low. Warning and major alarms are sent to the HPSS Alarms and Events window periodically when the warning and critical thresholds are exceeded, respectively. As of Release 4.1, the migration policy can be configured to automatically kicked off migration when either threshold is exceeded.

When a storage class experiences a threshold-exceeded condition, the administrator may need to review the associated migration and purge policies for the storage class, and the total storage space available to the storage class. The migration and purge policies may need to be modified to free up more space and to free up the space more frequently. In addition, the total storage space for the storage class may need to be reviewed to determine whether it is sufficient to accommodate the actual usage of the storage class.

The HPSS storage space can be monitored through the Active HPSS Storage Classes window and the Storage Class Information window.

6.4.2.1 Using the Active HPSS Storage Classes Window

From the HPSS Health and Status window (Figure 6-1), click on the Monitor menu and select the Storage Classes option.

The Active HPSS Storage Classes window will be displayed as shown in Figure 6-9. This window works similarly to the HPSS Servers window (Figure 6-2); however, in this window, only one storage class can be selected at a time. Also as with the HPSS Servers window, the actual display can vary greatly by the setting of window preferences.

To set preferences for the Active HPSS Storage Classes window, click the "Preferences" button on the window. Or, from the HPSS Health and Status window, click on the Session menu, select the Preferences option, and then select Storage Classes. The HPSS Storage Class List Preferences window is displayed as shown in Figure 6-10. Refer to the window's help file for more information on setting, saving, and reloading the preferences.

The Active HPSS Storage Classes window allows the SSM user to monitor the data for all storage classes that actually have storage space created in them. Refer to the window's help file for more information on the individual fields as well as the supported operations available from the window.

The Migration/Purge Server must be running and connected to SSM, otherwise the Active HPSS Storage Classes window will be empty. All Disk and tape Storage Servers must be running or the storages classes managed by them will not be reported in this window.

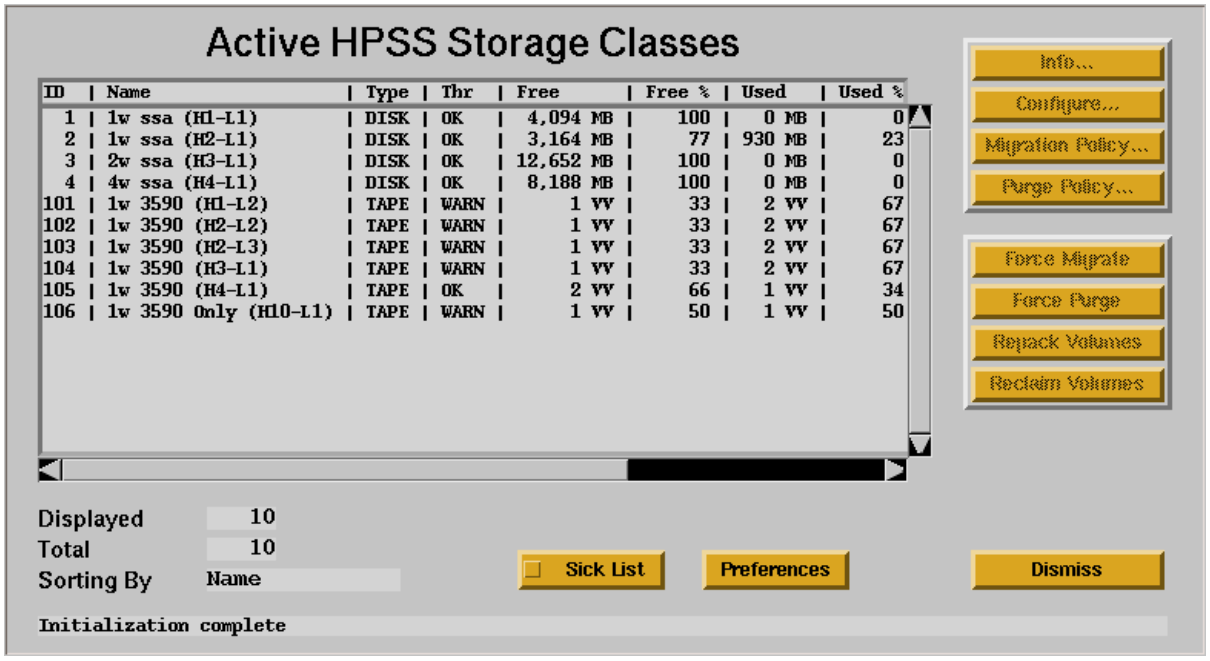


Figure 6-9 Active HPSS Storage Classes Window

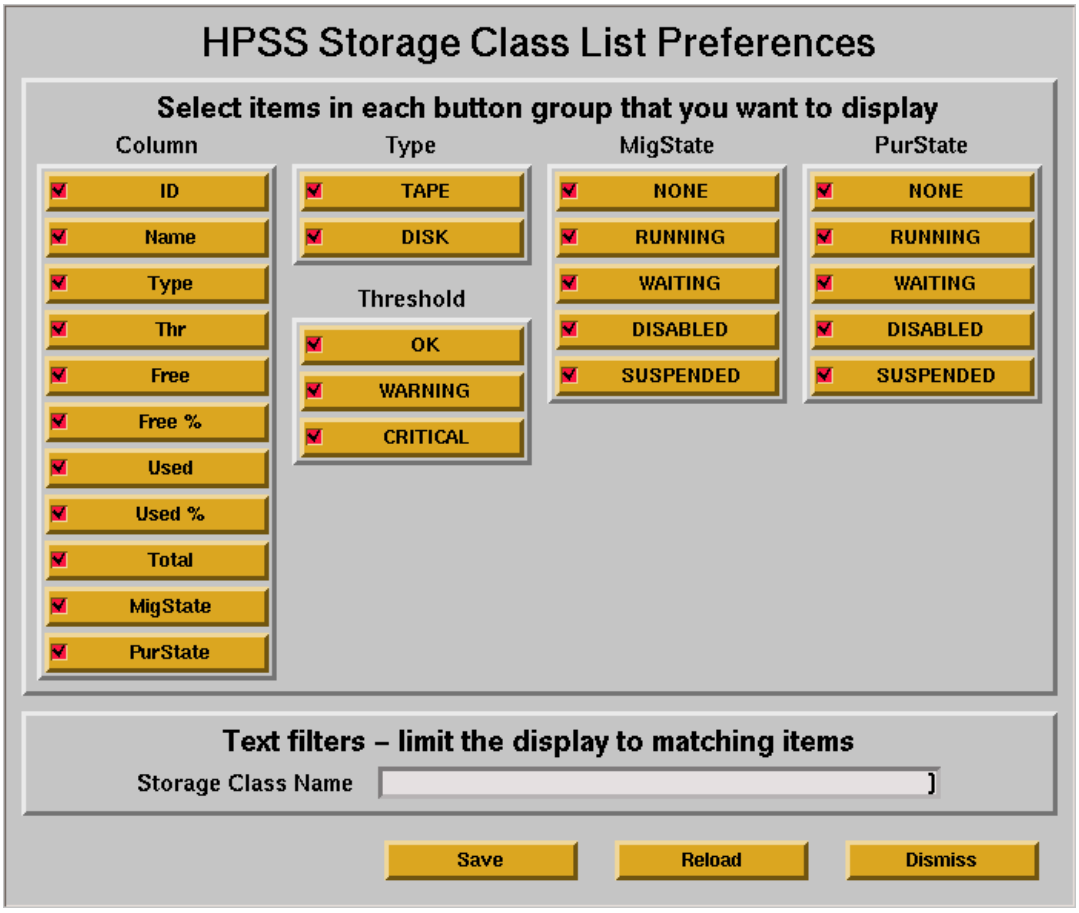


Figure 6-10 HPSS Storage Class List Preferences Window

6.4.2.2 Using the MPS Storage Class Information Window

From the HPSS Active Storage Classes window, select the appropriate storage class and click on the **Info...** button.

The MPS Storage Class Information window will be displayed as shown in Figure 6-11. This window reports the storage space data as well as any threshold exceeded conditions. The window also reports detailed information on the migration and purge statuses. In addition, this window allows the SSM user to further control the migration and purge process to override the associated migration and purge policies. Refer to the window's help file for more information on the individual fields as well as the supported operations available from the window.

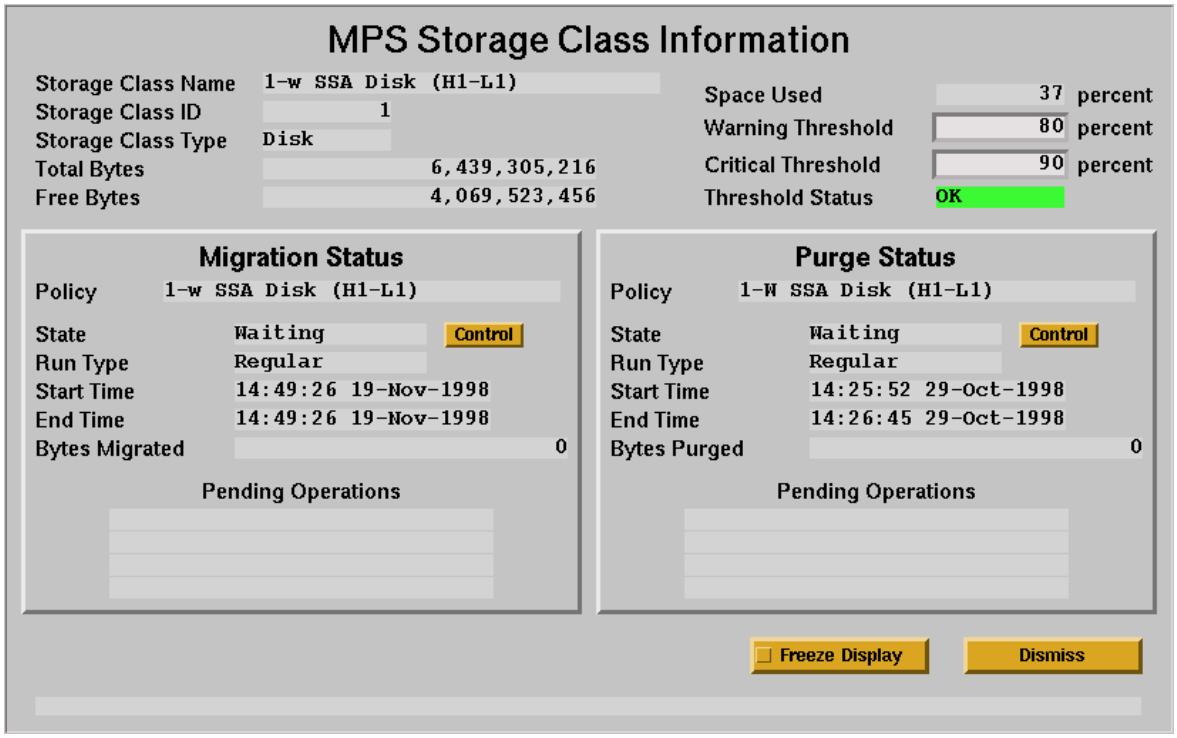


Figure 6-11 MPS Storage Class Information Window

6.4.3 *Creating Additional Storage Space*

Before the available storage space in a tape storage class drops to a level where the storage class may no longer be able to satisfy storage requests, efforts must be made to create additional free space. This can be accomplished by two methods. The first method is to tune the migration and purge policy to cause the migration and purge operations to occur more frequently, if necessary, and free up more storage space. The second method is to import and create additional storage volume into the storage class. Depending on the amount of additional space needed, one or both methods may be needed to achieve the goal. To fine tune the migration and purge policies, refer to Sections 5.4.1 and 5.4.2 for more information. To create additional storage space, refer to Section 6.4.1 for more information.

In addition, free tape storage space can be increased by reclaiming empty volumes and repacking sparse volumes. The repack process, which moves information from sparsely filled volumes to other volumes in the same storage class, can be started via SSM. When repack is finished, a number of volumes will be cleared of data and in an empty state. Before these volumes can be made available to HPSS for reuse, the metadata entries that describe them need to be regenerated. Reclaim, which can be started by SSM, is the process that locates these empty volumes and regenerates their metadata.

Before the available storage space in a disk storage class drops to a level where the storage class may no longer be able to satisfy storage requests, efforts must be made to create additional free space. Changing the migration and purge policies to cause migration and purge to occur more

frequently is the usual means of making more disk space available. This is the only way to create more free disk space, other than adding more disks to the system, which is usually not an option. To fine tune the migration and purge policies, refer to Sections 5.4.1 and 5.4.2 for more information.

6.4.4 Deleting HPSS Storage Space

At times, due to various reasons, the created storage space may need to be deleted so that the underlying volumes can be recreated with different storage characteristics or be exported from HPSS.

Deleting HPSS storage space can be separated into two distinct phases. The first phase, deleting the storage space, involves deleting Storage Server volumes that no longer contains HPSS files. The second phase, export, involves removing the volumes that are no longer allocated to a Storage Server from HPSS.

The existing storage space can be deleted and its freed-up volumes can be recreated without having to export the media out of the HPSS system. It is recommended that the media be exported from HPSS only if it is to be removed permanently.

6.4.4.1 Deleting Storage Server Space

An HPSS administrator must be able to delete storage media that HPSS no longer uses to store data. The deletion of media involves the deletion of storage server physical volume, virtual volume and storage map metadata information as well as deallocation of any media described by these structures from the PVL. The procedures for deleting storage media are the same for both disk and tape.

Unused volumes must be deleted at the level of the virtual volume (VV) that includes the media. You must delete all metadata structures associated with each virtual volume that is to be deleted. To delete a virtual volume, the volume must not contain any storage segments. Any attempt to delete a virtual volume containing storage segments will fail. This rule applies to both disk and tape.

Deleting Tape Volumes

For tapes, if the state of the virtual volume's storage map is **Empty**, all storage segments have been removed from the tape VV and no further data will be written to the volume. The volume metadata records can be deleted by following the steps described in the following paragraphs. If the storage map state is not **Empty**, that state must be attained.

Begin by setting the storage map state to **EOM** via SSM if it is not already in that state. This can be done by bringing up the SSM Identify Cartridges and Volumes window and enter the label of one of the physical volumes in the virtual volume, then selecting the "**SS Storage Ma**p" button. When the Storage Map Information window is displayed, select the "Map State" menu and select EOM.

Next move all of the active data on the tape VV to other VVs in the same class with the **repack** utility. The option to repack a specific volume is not available from SSM, but the **repack** utility can be invoked from a command line with the appropriate arguments to repack the desired volume. Refer to Appendix I for more information on how to invoke the **repack** utility. Once this is done, the volume's storage map state should be **Empty** and you can proceed with the deletion of the storage server tape space.

Deleting Disk Volumes

For disks, the procedure is somewhat different. There is no **EOM** or **Empty** state for disk storage maps. Before removing the disk virtual and physical volume, the disk must be free of user data. To get to this state, begin by setting the VV's storage map **administrative state** to **Locked** using SSM. This will prevent the Disk Storage Server from selecting the VV when new storage segments are created, but will not interfere with reading, writing, or copying existing segments. Once the map is **Locked**, move all of the storage segments in the VV to other VVs using the **repack** utility. When the **repack** is done, verify that no storage segments remain in the VV by examining the VV storage map (Section 6.7.5.4, Figure 6-35). The number of active segments should be zero and the **Administrative State** should be **Locked**. When this state is achieved, you may proceed with the deletion of the storage server disk space.

Using the Delete Storage Server Resources Windows

From the HPSS Health and Status window (Figure 6-1), select the **Operations** menu and click on **Delete SS Volumes** option.

The Delete SS Resources window will be displayed as shown in Figure 6-12. Refer to the window's help file for more information on the individual fields as well as the supported operations available from the window.

To delete SS storage space, generate the list of physical volumes to be deleted and click on the **Delete** button. After the SS volumes are deleted, the associated media are candidates for export. The media are also candidates for reuse by a Storage Server in need of more physical storage space.

Although the Delete SS Resources request is essentially a request to delete SS virtual volumes, the resources to be deleted are specified by physical volume names. Since there may be multiple physical volumes in one virtual volume, specifying one physical volume name may cause other physical volumes (in the same virtual volume) to be deleted. Conversely, specifying multiple physical volumes belonging to the same virtual volume will effectively generate only one delete request; subsequent requests to delete other physical volumes in the same virtual volume will be ignored.

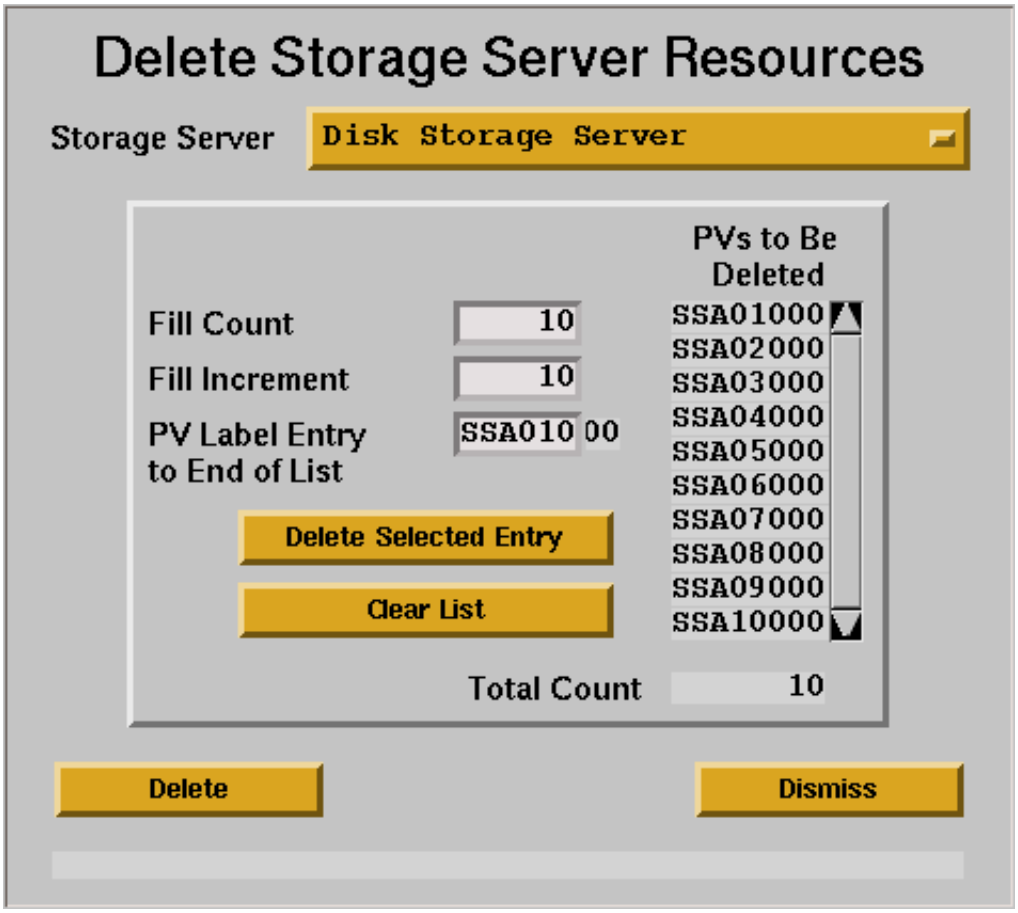


Figure 6-12 Delete Storage Server Resources Window

6.4.4.2 Exporting Volumes from HPSS

An HPSS administrator must be able to export physical volumes that are not allocated to a Storage Server. Unallocated volumes remain available for allocation by a Storage Server needing more storage space until the volumes are removed from HPSS via export. Unallocated volumes can be allocated using the Create Storage Server Resources option discussed in Chapter 6, Figure 6-8.

Only unallocated volumes can be exported. Volumes allocated to a Storage Server can be deallocated by deleting all storage server data structures that describe the volume using the Deleting Storage Server Space procedure described in Section 6.4.4.1.

When exporting a removable cartridge from a robot, the cartridges will be sent to the convenience I/O port if it is available. If the port is not available and a high-capacity I/O region is defined, the cartridges will be placed in that region. If no locations are available for the cartridge to be ejected, an alarm will appear on the HPSS Alarms and Events window and HPSS will periodically retry the eject operation.

Using the Export HPSS Media Window

From the HPSS Health and Status window, click on the **Operations** menu and select the **Export Volumes** option .

The Export HPSS Media window is displayed as shown in Figure 6-13. Refer to the window's help file for more information on the individual fields as well as the supported operations available from the window.

To specify the media to be exported, enter the **Media Label Entry to End of List**, **Fill Count,** and **Fill Increment** fields to generate the export cartridge list. Verify that the export list is correct and then click on the **Export** button to initiate the export request. Once the export is completed, those cartridges residing in robotically mounted systems will be ejected by the PVR controlling that robot and should be removed from the access port of the robot.

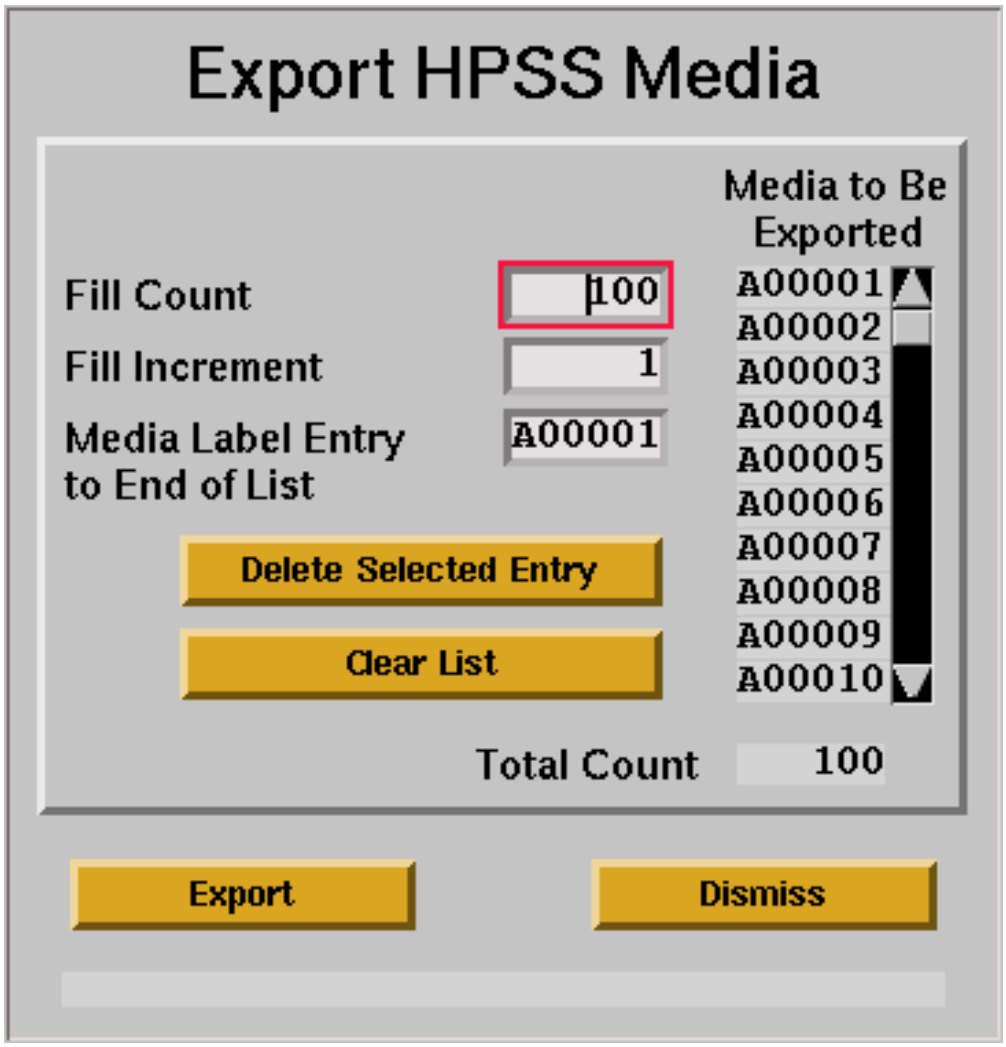


Figure 6-13 Export HPSS Media Window

6.4.5 Forcing Migration

The Migration / Purge Server is set up to run migration periodically with the time interval specified in the migration policy. However, between these automatic migration runs, an administrator can use either the HPSS Active Storage Classes window (Figure 6-9) or the MPS Storage Class Information window (Figure 6-11) to force a migration to take place. When a migration run is forced, the run timer is reset. The next migration will take place by the specified interval after the end of the forced run.

From the HPSS Active Storage Classes Window, select a storage class and then click on the **Force Migrate** button.

From the MPS Storage Class Information Window, click on the **Control** button in the **Migration Status** area and select the **Start Run** option from the popup menu.

The Migration / Purge Server stops the migration run when either the **Free Space Target** threshold in the migration policy is reached or there are no more disk bitfiles or tape virtual volumes eligible for migration.

6.4.6 Forcing Purge

The Migration / Purge Server communicates with the appropriate Disk Storage Server to obtain the disk storage class statistics base on the interval specified in the MPS specific configuration entry. The Migration / Purge Server will then evaluate the used space and start a purge run automatically if needed. However, between these automatic purge runs, an administrator can use either the HPSS Active Storage Classes window or the MPS Storage Class Information window to force a purge run to take place.

From the HPSS Active Storage Classes Window, select a storage class, then click on the **Force Purge** button.

From the MPS Storage Class Information Window, click on the **Control** button in the **Purge Status** area and select the **Start Run** option from the popup menu.

When the force purge is initiated, the Migration / Purge Server checks the used space in that storage class. If the used space exceeded the **Start purge when space used reaches** field specified in the purge policy, the Migration / Purge Server starts to purge bitfiles from disk. The Migration / Purge Server stops the purging when the **Stop purge when space used falls to** specified in the purge policy is reached or there are no more disk bitfiles eligible for purging.

6.4.7 Repacking HPSS Volumes

Over time, the active data on an HPSS virtual volume may become sparse as data is deleted or migrated / purged. The **repack** utility gives the administrator the capability to concentrate data on volumes, thus allowing more efficient use of resources. It can also be used on a stand-alone basis by an administrator when data is to be moved off of specific media due to a damaged physical volume or a need to replace the media. The utility can also be used to repack virtual volumes at the lowest level of a hierarchy where migration is not available.

Care should be taken to ensure that the HPSS and tape drives are not overloaded with repack operations during heavy peaks.

Using the Repack Virtual Volumes Window

From the HPSS Active Storage Classes window, select a storage class, then click on the **Repack Volumes** button.

This will bring up the Repack Virtual Volumes window as shown in Figure 6-14. Refer to the window's help file for more information on the individual fields as well as the supported operations available from the window.

Enter the number of volumes (VVs) you wish to repack, specify the repack threshold (the percentage of actual data stored on volume), indicate whether to repack **EOM** or **Retired EOM** volumes, indicate if **Shelved VVs** should be repacked, and select the appropriate Storage Server name from the pop-up list. Once the fields are set, click on the **Repack** button. SSM will report the number of repacked volumes on the status field of the window when the repack operation completes.

The SSM Repack Virtual Volumes window can only be used for repacking tape virtual volumes. To repack disk virtual volumes, use the **repack** utility. The **repack** utility can also be invoked as a command line utility to repack tape or disk virtual volumes with more flexibility than when it is invoked from the SSM window. Refer to Appendix I for more information on the how to invoke the utility.

After a virtual volume is repacked, it remains in the "Empty" state and is not immediately available for storing data. To make the volume available, it must be reclaimed.

To repack all eligible virtual volumes in a storage class, specify the total number of virtual volumes created on the storage class or a large number of volumes. The repack utility will only repack as many volumes as it can find in the storage class.

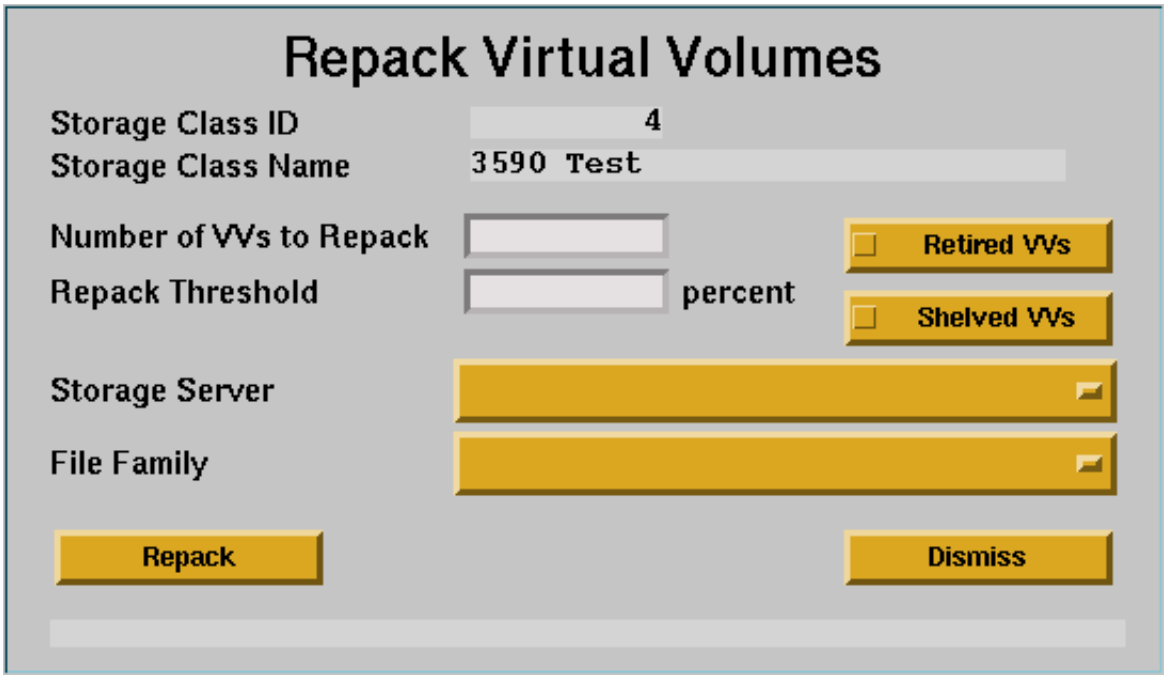


Figure 6-14 Repack Virtual Volumes Window

6.4.8 Reclaiming HPSS Tape Virtual Volumes

Once a tape virtual volume reaches **EOM** or **Retired** state, no new data may be written to the volume. Over time, the amount of data on a volume will decrease as data is deleted or migrated and purged. At some point, a volume will end up in an empty state when no active data resides on the volume through natural attrition, migration and purge, and possibly repack. At this point, the volume becomes a candidate for reclaim. The reclaim process will return a volume to its pristine state, and the volume is again available for use.

Using the Reclaim Virtual Volumes Window

From the Active HPSS Storage Classes window, select a storage class and then click on the **Reclaim Volumes** button.

This will bring up the Reclaim Virtual Volumes window as shown in Figure 6-15. Refer to the window's help file for more information on the individual fields as well as the supported operations available from the window.

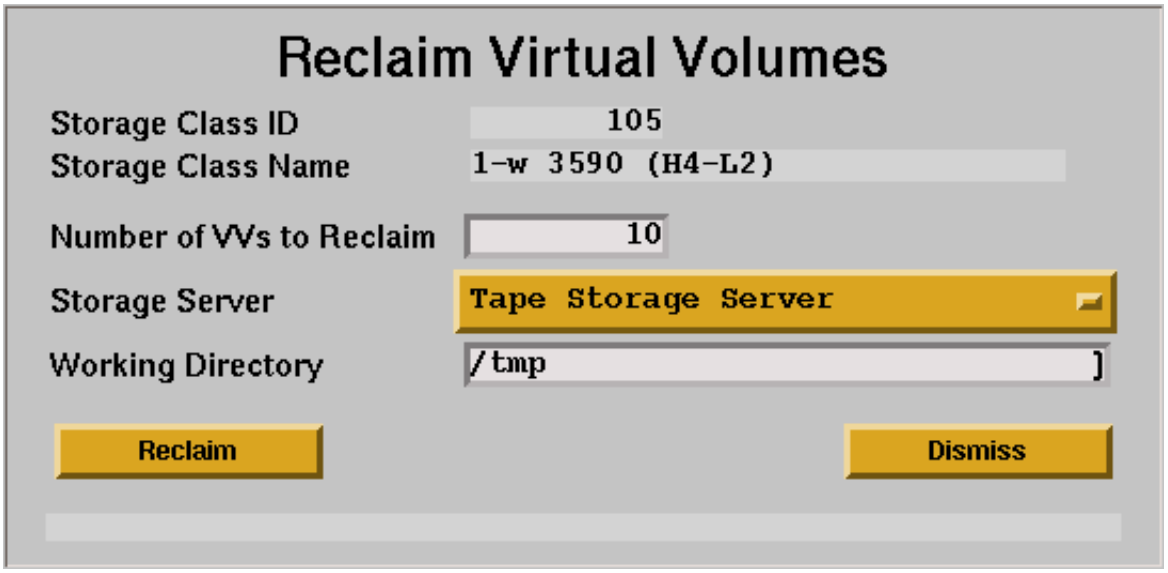


Figure 6-15 Reclaim Virtual Volumes Window

Enter the number of volumes (VVs) you wish to reclaim and select the appropriate Storage Server name from the pop-up list. Once the fields are set, click on the **Reclaim** button. SSM will report the number of reclaimed volumes on the status field of the window when the reclaim operation completes.

Four files are generated during the reclaim procedure. Two of the files contain report information. These report files will generate approximately 150 bytes and 1200 bytes of information per volume reclaimed. The third file will contain the list of virtual volume IDs to reclaim. The size of this file will be 70 bytes per volume. The fourth file will contain metadata on the volume currently being reclaimed and any volume unsuccessfully reclaimed. The size of this file will be approximately 700 bytes per volume. This size is for a virtual volume, which is comprised of a single physical volume. Each additional physical volume in a virtual volume will add approximately 250 bytes to the fourth file. So roughly 2200 bytes per reclaimed volume would be used in the **reclaim** working directory. The files do not accumulate as they are overwritten by the **reclaim** procedure.

The **reclaim** utility can also be invoked as a command line utility with more flexibility than when it is invoked from the SSM window. Refer to Appendix I for more information on how to invoke the utility.

The reclaim utility is intended for reclaiming tape virtual volumes only.

To reclaim all eligible virtual volumes in a storage class, specify the total number of virtual volumes created on the storage class or a large number of volumes. The reclaim utility will only reclaim as many volumes as it can find in the storage class.

6.4.9 New Storage Technology Insertion

As more advanced storage technology becomes available or old storage technology becomes obsolete, there may be a need to replace the existing storage technology used by HPSS. HPSS provides a capability to replace the currently used storage technology with another technology. With this capability, the old technology volumes are marked "Retired" and new technology volumes are then created in the same storage class. Files written to this storage class will be written to new the technology volumes while the old technology volumes are treated by HPSS as read-only. Attrition and repacking will move all of the files from the old technology volumes to the new. When the old technology volumes are empty they can be removed from HPSS.

To replace old technology volumes with new technology volumes in a storage class, perform the following steps:

1. Use the **retire** utility to set the storage map state of all old technology volumes to "**Retired**".
2. Shut down the NS, BFS and MPS to acquiesce the system.
3. Delete and then recreate the storage class definition. The new storage characteristics must reflect the characteristics of the new storage technology volumes. In addition, they should be compatible with other storage levels in the hierarchy. *Note: Changes to the storage class metadata do not effect volumes that already exist when the changes are made. The characteristics of established volumes stay the same throughout their lifetime. The Storage Class characteristics only effect the characteristics of volumes when they are created.*
4. Import and create new storage technology volumes.
5. Restart the NS, BFS, and MPS.
6. Invoke the **repack** utility to repack all "**Retired**" volumes in the storage class.
7. Invoke the **remove** utility to removed all Retired/Empty volumes. This utility will delete all resources associated with these volumes and export them out of HPSS.

6.4.10 Changing Storage Characteristics

Generally, once a storage description has been defined, basic changes to this description can only be done in special situations and must be done very carefully. The following sections provide some detail on changing various parameters in storage descriptions.

6.4.10.1 Changing Storage Class Definition

A storage class definition can be changed by bringing up the Storage Class Configuration window (Figure 5-10 or Figure 5-10) and making the desired modifications,

Fields that can be changed in the storage class definition without major impact are the **Migration/Purge Server**, **Migration Policy**, **Purge Policy**, **Warning Threshold,** and **Critical Threshold**. These fields basically affect migration, purging, and warning of storage space shortage conditions. They can be changed at the discretion of the administrators to reflect desired behavior.

Optimum Access Size, **Stripe Transfer Rate,** and **Average Latency** are used only in an advisory capacity by the HPSS system unless your site has written a special interface using the Client API that takes advantage of the COS Hints capability and uses these fields in doing so. Generally, they can be changed at will, but should be set to accurately reflect their intended usage.

Maximum Storage Segment Size and **Average Number of Storage Segments** can be changed as desired for disk storage classes. For appropriate settings of these values, consult section 2.8.

To change the **Media Type**, **VV Block Size**, **Stripe Width**, **Media Block Size**, **Blocks Between Tape Marks**, or **Storage Segment Size**, extreme caution should be exercised. This is considered to be an unusual procedure, and not following proper procedures can result in serious consequences including possible data loss or data corruption. The guidelines for this process given below must be followed exactly. Note that to change the value of any of these fields requires deleting the storage class entry and re-creating it. For safety purposes, these fields are not directly updateable from the SSM menus.

- Delete all storage space from this storage class. This can only be done if no bitfiles have storage on a device in this storage class. This can be accomplished by file deletion or (for disk) by migrating all of the data off the devices in this storage class and running purge. When doing this, you must guarantee that no new data is added to the storage class in the process. One way to do this is to set the COS containing this storage class to **Read Only** for the time period needed to accomplish this task. You must also turn off any automatic staging by setting the **COS Stage Code** to **No Stage**. In addition, no applications should be issuing stages that result in this storage class having data staged to it.

- Make the desired changes to the storage class definition (by deleting its configuration then re-creating). Make sure that the changes you are making are consistent with the definition of the storage class, the hierarchy, and the COS this storage class belongs to.

- Recycle the BFS and the MPS.

- At this point, you should be able to use the new definitions.

6.4.10.2 Changing Storage Hierarchy Definition

A storage hierarchy definition can be changed by bringing up the Storage Hierarchy Configuration window (Figure 5-12) and making the desired modifications,

The only modification to the storage hierarchy definition that can be made is the addition of an additional storage level at the bottom of the hierarchy. Note that even in this case you must exercise caution. If you have a hierarchy with disk and you are migrating to tape, then you could add another tape level. If you do this and then turn on multiple copies, files that have already been migrated will not have a second copy unless they are staged to disk and migrated again. To modify the storage hierarchy definition, you must delete and recreate the definition.

6.4.10.3 Changing Class of Service Definition

A COS definition can be changed by bringing up the Class of Service Configuration window (Figure 5-13) and making the desired modifications,

Fields on the HPSS Class of Service window that can be changed without major impact are the **Access Frequency**, **Optimum Access Size**, **Average Latency,** and **Transfer Rate**. These fields are advisory in nature and not currently used by any HPSS interfaces except the Client API. If you have written any interfaces using the COS Hints capability via the Client API and you are using one of

these fields, you will need to assess the impact of your change. The basic impact will be in the selection of the COS at file create time.

The **Maximum File Size** can be changed, but care should be exercised when changing this field. Increasing the maximum file size may result in storing files that are inappropriate for the hierarchy supporting this COS. For example, suppose you have defined a hierarchy with disk at the top level with a storage segment size of 64 KB. Also assume that the **Maximum File Size in** this COS was set to 1 MB. Changing the maximum file size to 2 GB in this class would be inappropriate because it would result in large files having excessively large amounts of metadata. Associated with the **Maximum File Size** is the **Enforce Maximum File Size** indicator. Changing this can have a significant impact. If the indicator is originally off and is then turned on, any files that already exceed the **Maximum File Size** cannot grow, and any files under this limit will then be limited to the **Maximum File Size**. If the indicator was originally on and is then turned off, the impact is less severe. In this case, files will no longer be constrained in size. The administrator needs to take this into account if this change is to be made.

Changing **Minimum File Size** can have an impact on COS selection. Currently, the PFTP interface and FTP interfaces (if client supports **alloc**), use this field in selecting an appropriate COS based on file size.

Changing the **Stage Code** should be done with care. Changing between synchronous and asynchronous stage is not a major problem. The primary difference is that if the stage is synchronous and the open is successful, the user knows with certainty the stage was successful. Changing to **No Stage** can have a significant impact. Turning staging off means that repeated reads to files that have been migrated will generally be satisfied from lower levels in the hierarchy that will likely involve tape. Also, if you are writing only parts of a file at a time with significant delays between the writes, migration may result in the file being stored in a more fragmented manner on tape.

You cannot change the hierarchy associated with a COS without deleting all files in this COS or moving them to another COS. Failure to observe this condition will potentially result in lost and corrupted files. Currently, the **scrub** utility can be used to change the COS of a file. Refer to Section 8.3 for more information

6.4.11 Deleting Storage Characteristics

Deletion of COS, storage hierarchy, or storage class definitions is not an operation that is expected to occur in normal HPSS operation. The current release does not provide appropriate support to make this a reasonably simple operation. Later releases will provide more capability in this area. Deletion of any one of these objects currently is a complex process; it is recommended that sites do not attempt these operations. The sections below provide explanations of what is involved in deleting one of these objects.

6.4.11.1 Deleting Storage Class Definition

To delete a storage class definition, you must ensure that no data exists in this storage class. You can only delete a storage class if it is no longer referenced in any hierarchy. Normally, deleting storage classes would be done when a COS and a hierarchy that contained the storage class are being removed. It is possible to remove a storage class entry if it is at the bottom of the hierarchy. To do this, you would have to ensure that all data in this storage class was staged up to a higher level, and migration to the lower level would have to be shut off. Once the storage class is empty, it could be removed from the hierarchy definition. All of the virtual volumes in this storage class should then

be deleted. The storage class definition then can be removed. HPSS Release 4.1.1 does not provide needed utilities to facilitate this operation.

6.4.11.2 Deleting Storage Hierarchy Definition

Essentially, the same rules that apply for deleting a COS also apply for deleting a storage hierarchy. Before deleting a hierarchy, you must delete all the files in all of the Classes of Service that use this hierarchy. Note that normally a 1 to 1 relationship between hierarchies and Classes of Service is expected. Again, HPSS does not provide the kind of utilities in Release 4.1.1 needed to facilitate the operation.

6.4.11.3 Deleting Class of Service Definition

To delete a COS, you must ensure that all files in the COS have either been deleted or moved to another COS. This is not a normally anticipated operation and HPSS Release 4.1.1 does not provide utilities to delete all the files in a given COS, move all files in a given COS to another COS, or count the number of files in a given COS. Future releases of HPSS will provide utilities of this nature.

6.5 Managing Filesets and Junctions

6.5.1 Creating Filesets and Junctions

SSM can be used to create DFS/HPSS and HPSS-only filesets. SSM can also be used to create the junctions that link the filesets to the HPSS name space.

Ensure that the Name Server and the DMAP Gateway are up and running before creating the filesets and junctions. Also ensure that SSM is connected to both of the servers.

6.5.1.1 Creating a DFS/HPSS Fileset

From the Health and Status window (Figure 6-1), select the **Operations** menu and click on the **Create DFS/HPSS Fileset** option. The Create DFS/HPSS Fileset window will be displayed as shown in Figure 6-16. Enter the appropriate information and click on the **Create** button.

To update or delete an existing fileset, refer to Section 6.5.1.4 for more information on how to change a fileset definition.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

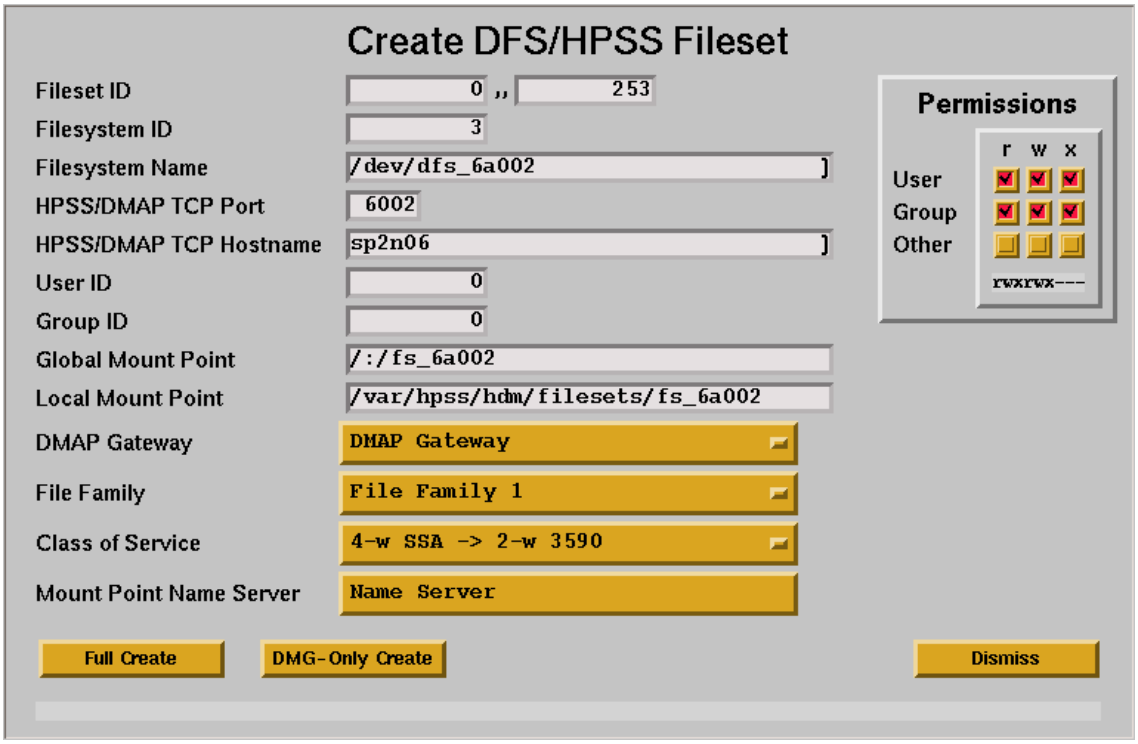


Figure 6-16 Create DFS / HPSS Fileset Window

Create DFS/HPSS Fileset Variables

Table 6-5 lists the fields on the Create DFS / HPSS Fileset window and provides Class of Service configuration information.

Table 6-5 Create DFS/HPSS Fileset Variables

Display Field Name	Description	Acceptable Values	Default Value
Fileset ID	The fileset ID to be assigned to the fileset. A fileset ID consists of two numbers (high and low) separated by a double comma.	Any unsigned 32-bit integers. The value must match that of a fileset already created in DFS.	Blank
Filesystem ID	The ID number of the DFS filesystem where the DFS fileset resides.	Any unsigned 32-bit integer. The value must match that of a filesystem already created in DFS.	None

Table 6-5 Create DFS/HPSS Fileset Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Filesystem Name	The device path name of the DFS filesystem corresponding to the Filesystem ID field. An example would be /dev/dsk/c0t0s1.	A character string up to 255 bytes in length. The name must match that of an actual DFS filesystem.	None
HPSS/DMAP TCP Port	The TCP/IP port number used by the HPSS/DMAP process, which manages the DFS fileset, to listen for requests from HPSS.	Any integer from 1 to 65535. The value must match that of an actual HPSS/DMAP server.	None
HPSS/DMAP TCP Hostname	The name of the host on which the HPSS/DMAP process, which manages the DFS fileset, runs.	A character string up to 63 bytes in length. The value must match that of an actual host where the HPSS/DMAP server runs.	None
User ID	The Unix UID identifying user owning the fileset.	Any valid Unix UID.	None
Group ID	The Unix GID identifying the group owning the fileset.	Any valid Unix GID.	0
Global Mount Point	The DFS pathname to the directory where the fileset is mounted. This pathname resides on the same machine as the HDM.	A character string up to 255 bytes in length. The value must match the mount point defined when the fileset was created in DFS.	None
	Advice: This field applies only to mirrored filesets. For archived filesets, it should be left blank. A DFS mount point usually looks like **/.../dce.cell.name/fs/file/set/name**, where **dce.cell.name** is the name of the DCE cell where the fileset resides, and **file/set/name** is some subpath of the fs CDS name space junction.		
Local Mount Point	The Unix pathname to a directory where a local fileset will be mounted.	A character string up to 255 bytes in length. The directory must already exist, and must be unique for each local fileset.	None
	Advice: This field applies only to mirrored filesets in the local DCE/DFS cell. For archived filesets, it should be left blank. For remote mirrored filesets (in another DCE/DFS cell), anything in this field will be ignored.		
DMAP Gateway	The name of the DMAP Gateway to which the fileset create request will be sent.	Any DMAP Gateway in the popup selection list.	None
File Family	The name of the file family to be associated with this fileset.	Any file family in the popup selection list, or blank to specify no file family association.	None

Table 6-5 Create DFS/HPSS Fileset Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Class of Service	The name of the Class of Service to be assigned to this fileset.	Any COS in the popup selection list, or blank to specify no assigned COS.	None
Mount Point Name Server	The name of the Name Server which will manage the HPSS mount point for the fileset.	Any Name Server in the popup selection list.	None
Permissions	The initial permissions to be assigned to the fileset.	Any valid combination of Unix file permissions.	rwx------

6.5.1.2 Creating an HPSS-only Fileset

From the Health and Status window (Figure 5-1), select the **Operations** menu and click on the **Create HPSS-only Fileset** option. The Create HPSS-only Fileset window will be displayed as shown in Figure Figure 6-17. Enter the appropriate information and click on the **Create** button.

To update or delete an existing fileset, refer to Section 6.5.1.4 for more information on how to change a fileset definition.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

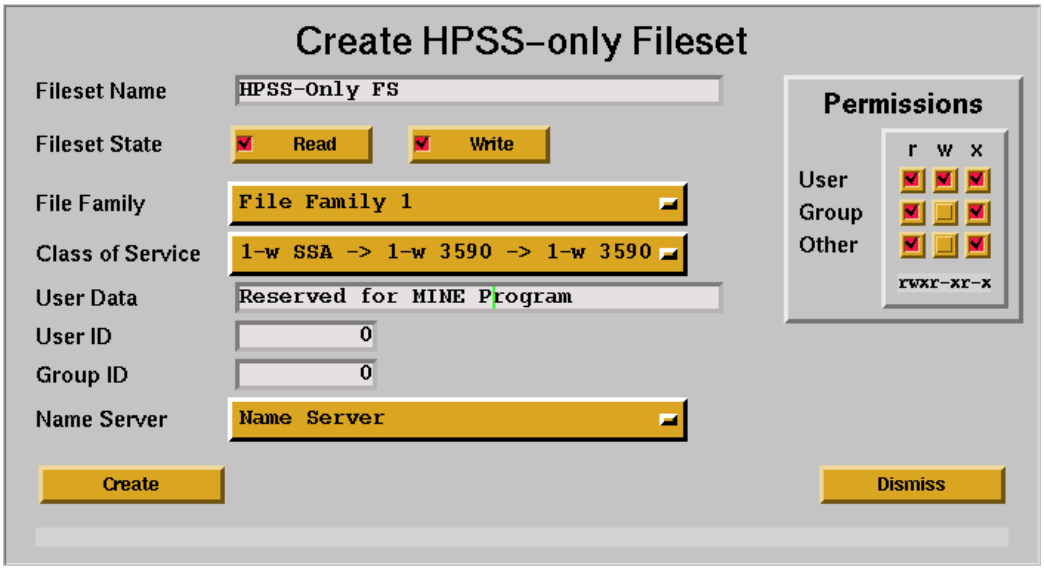


Figure 6-17 Create HPSS-only Fileset Window

Create HPSS-only Fileset Variables

Table Table 6-6 describes the create HPSS-only fileset variables.

Table 6-6 Create HPSS-only Fileset Variables

Display Field Name	Description	Acceptable Values	Default Value
Fileset Name	The name to be assigned to the fileset.	A character string up to 127 bytes in length.	None
Fileset State	The initial state of the fileset. If Read is ON, the fileset will be available for reading. If Write is ON, the fileset will be available for writing.	ON, OFF	ON for both Read and Write
File Family	The name of the file family to be associated with this fileset.	Any file family in the popup selection list, or blank to specify no file family association.	None
Class of Service	The name of the Class of Service to be assigned to this fileset.	Any COS in the popup selection list, or blank to specify no assigned COS.	None
User Data	Any name or other data that the administrator wishes to associate with the fileset. HPSS does not use this field in any way; it is strictly for user convenience in annotating the fileset.	A character string up to 128 bytes in length. Non-printable or binary bytes may also be entered using backslash-octal notation. See the window help file for more information.	None
User ID	The Unix UID identifying the user owning the fileset.	Any valid Unix UID.	0
Group ID	The Unix GID identifying the group owning the fileset.	Any valid Unix GID.	0
Name Server	The name of the Name Server to which the fileset create request will be sent.	Any Name Server in the popup selection list.	None
Permissions	The initial permissions to be assigned to the fileset.	Any valid combination of Unix file permissions.	rwxr-xr-x

6.5.1.3 Creating a Junction

From the Health and Status window (Figure 6-1), select the **Operations** menu and click on the **Create Junction** option. The Create Junction window will be displayed as shown inFigure 6-18. Enter the appropriate information and click on the **Create** button.

To delete an existing junction, refer to Section 6.5.2 for more information.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

The **/usr/lpp/hpss/bin/crtjunction** and **/usr/lpp/hpss/bin/deljunction** utilities may be also used to create and delete junctions.

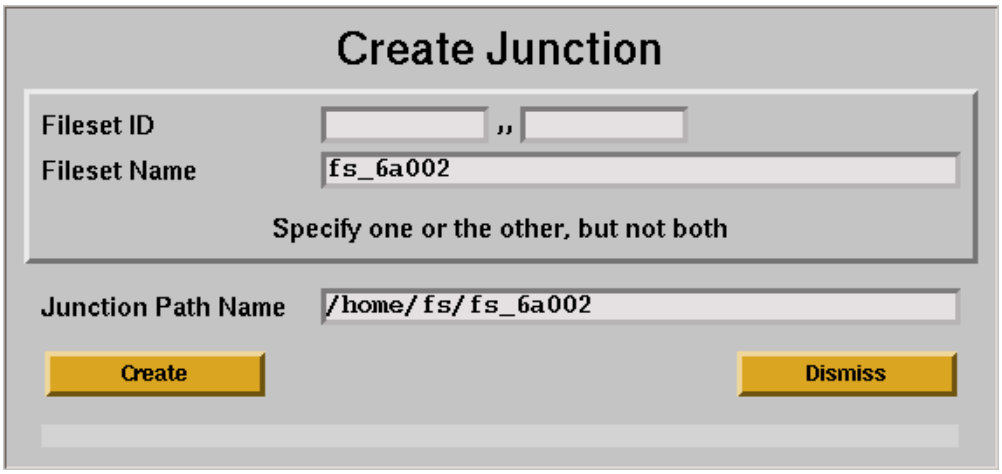


Figure 6-18 Create Junction Window

Create Junction Variables

Table Table 6-7 describes the create junction variables.

Table 6-7 Create Junction Variables

Display Field Name	Description	Acceptable Values	Default Value
Fileset ID	The fileset ID which identifies the fileset to which the junction will point. The fileset may be identified either by ID or by name (below). A fileset ID consists of two numbers (high and low) separated by a double comma.	Any unsigned 32-bit numbers. The values must match the ID of an existing fileset.	None

Fileset Name	The name which identifies the fileset to which the junction will point. The fileset may be identified either by name or by ID (above).	A character string up to 127 bytes in length.	None
Junction Path Name	The absolute path name, in the HPSS name space, of the junction that is to be created.	A character string up to 1023 bytes in length.	None

6.5.1.4 Managing Existing Filesets

The DMAP Gateway allows the SSM user to view, update, or delete the information for a DFS/HPSS fileset. Such a fileset is known to both DFS and HPSS. The Name Server allows the SSM user to view, update, or delete the information for a DFS/HPSS fileset or for an HPSS-only fileset. An HPSS-only fileset is known to HPSS, but not to DFS. Note that a DFS/HPSS fileset has two sets of information, one from the DMAP Gateway and one from the Name Server.

From the Health and Status window (Figure 6-1), click on the **Monitor** menu, select the **HPSS Objects** and click on the **Filesets** option. A popup window as shown in Figure 6-19 will be displayed to allow the user to specify a fileset and select which kind of fileset information is desired. Clicking the **Get NS Info** button will cause the Name Server Fileset Information window to be displayed as shown in Figure 6-20. Clicking the **Get DMG Info** button will cause the DMAP Gateway Fileset Information window to be displayed as shown in Figure 6-21. Note that clicking the **Get DMG Info** button for an HPSS-only fileset will result in an invalid error.

To update fileset information, modify the desired fields then press the **Enter** key.

To delete a DFS/HPSS fileset, bring up the DMAP Gateway Fileset Information window then click on the **Delete Fileset** button. Note that the **Delete Fileset** option from the **Operations** menu from the Health and Status window can also be used to delete a fileset.

To delete an HPSS-only fileset, bring up the Name Server Fileset Information window then click on the **Delete Fileset** button. Note that the **Delete Fileset** option from the **Operations** menu from the Health and Status window can also be used to bring up the same fileset information window for deleting a fileset.

Refer to the help files for these windows for more information on the individual fields as well as the supported operations available from the windows.

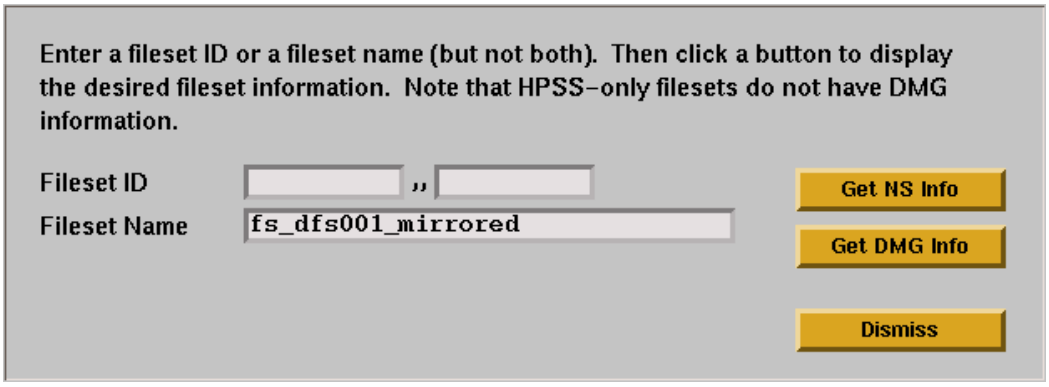


Figure 6-19 Identify Fileset Window

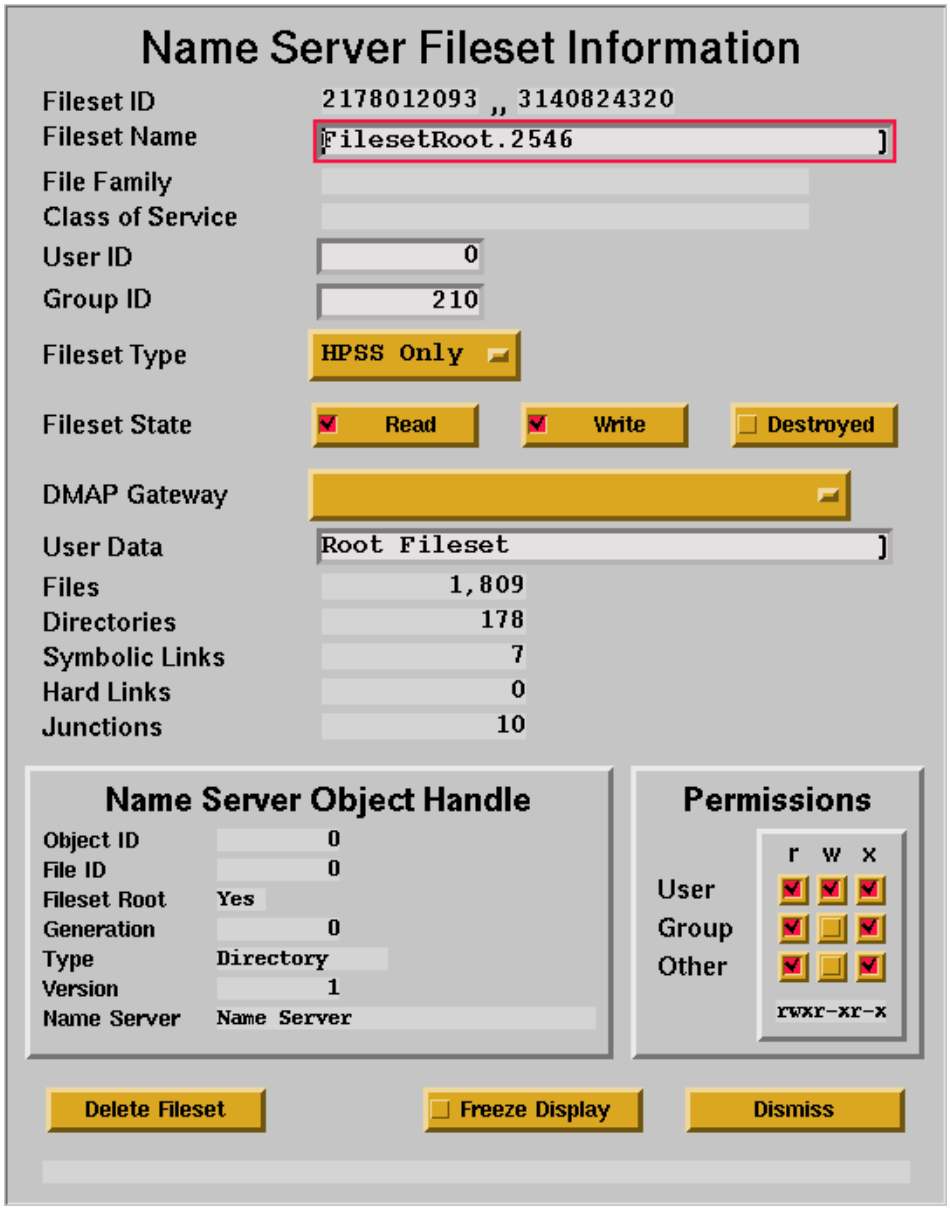


Figure 6-20 Name Server Fileset Information Window

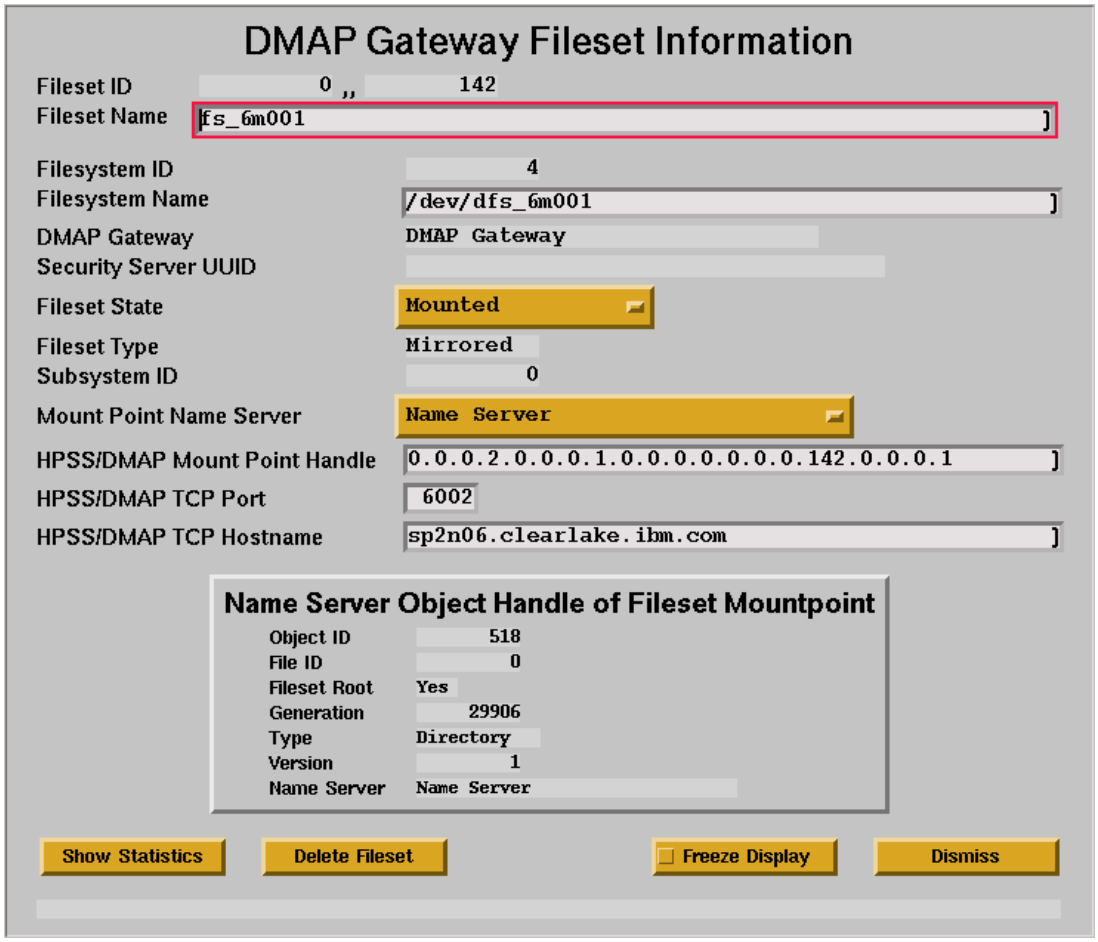


Figure 6-21 DMAP Gateway Fileset Information Window

6.5.2 *Deleting a Fileset Junction*

From the Health and Status window (Figure 6-1), select the **Operations** menu and click on the **Delete Junction** option. The Delete Junction window will be displayed as shown in Figure 6-22. Enter the appropriate information and click on the **Delete** button.

Figure 6-22 Delete Junction Window

6.6 Managing HPSS Devices and Drives

Managing the HPSS devices and drives consists of monitoring the configured devices / drives, adding a new device / drive, and deleting an existing device / drive. A device / drive references the same physical drive and is managed by both a Mover and a PVL. The Mover, which calls a drive a device, is in charge of all commands and data going to and from the device itself. The PVL is in charge of mounting the media on the drive, and uses the Mover to label a media and poll a drive for newly mounted media. Because both servers are involved with different aspects of a drive, most administrative actions should be simultaneously performed on the Mover and the PVL.

Administrators and operators with StorageTek, or IBM devices at their sites should review Appendix K (StorageTek), Appendix J (IBM) or Appendix L (ADIC AML) before attempting a device addition, deletion, or state change.

6.6.1 Monitoring the HPSS Devices and Drives

After the devices / drives are configured and are made available for services, one of the SSM user's task is to monitor their statuses. The goal is to detect any problems reported by the controlling PVLs and Movers so that corrective actions can be taken to ensure that the HPSS services are not disrupted due to errors or failures in the HPSS devices and drives.

When a device / drive experiences a problem, the PVL and / or the Mover issues one or more alarms to inform the SSM user of the problem and to notify SSM of the changes in the device / drive state. SSM reports the state change on the Device / Drives Status field on the HPSS Health and Status window. SSM also reports the new state on the HPSS Devices and Drives window and the Mover Device Information window or the PVL Drive Information window if they are currently displayed. It is the responsibility of the SSM user to investigate whether the problem is hardware-related, or whether it can be corrected through the HPSS servers or through manual manipulation.

The HPSS devices and drives can be monitored through the HPSS Health and Status window, the HPSS Devices and Drives window, the Mover Device Information window, and the Drive Information window. Refer to Section 6.2 for more information on the HPSS Health and Status window. The other windows are described in the following subsections.

Using the HPSS Devices and Drives Window

From the HPSS Health and Status window (Figure 6-1), click on the **Monitor** menu and select the **Devices and Drives** option.

The HPSS Devices and Drives window will be displayed, appearing initially as shown in Figure 6-23. This window works similarly to the HPSS Servers window (Figure 6-2). As with the HPSS Servers window, the actual display can vary greatly by the setting of window preferences.

To set preferences for the HPSS Devices and Drives window, click the **Preferences** button on the window. Or, from the HPSS Health and Status window, click on the **Session** menu, select the **Preferences** option, and then select **Devices and Drives**. The HPSS Device and Drive List Preferences window is displayed [as shown in Figure 6-24]. Refer to the window's help file for more information on setting, saving, and reloading the preferences.

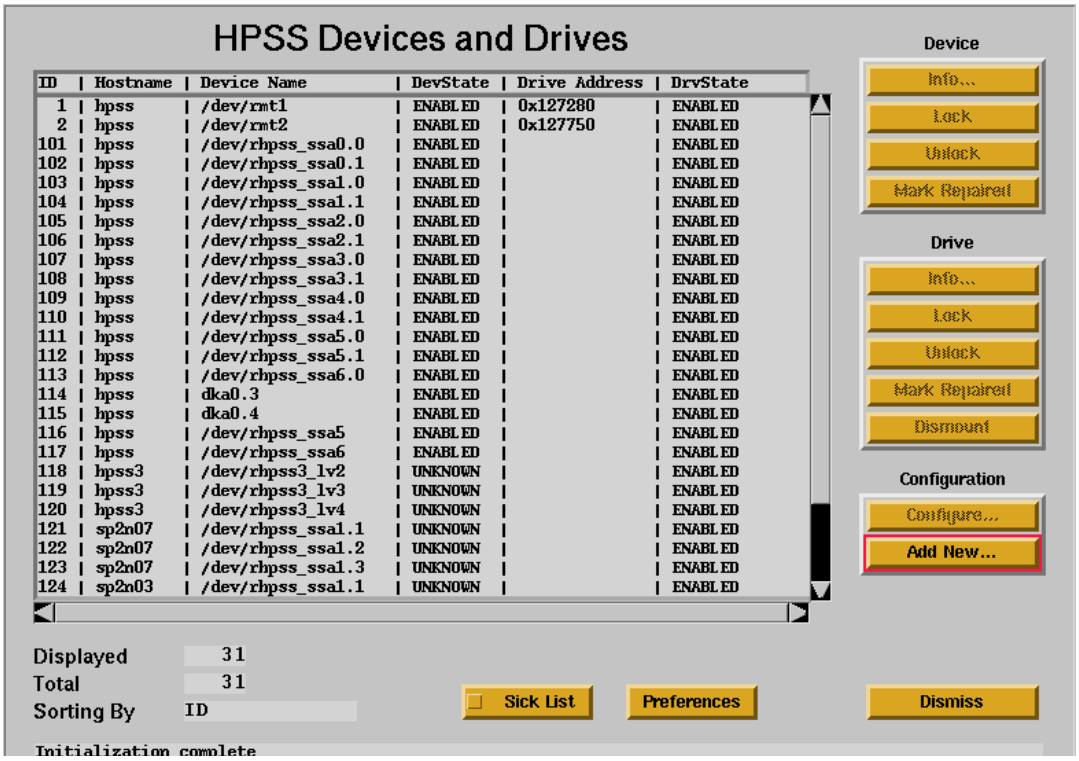


Figure 6-23 HPSS Devices and Drives Window

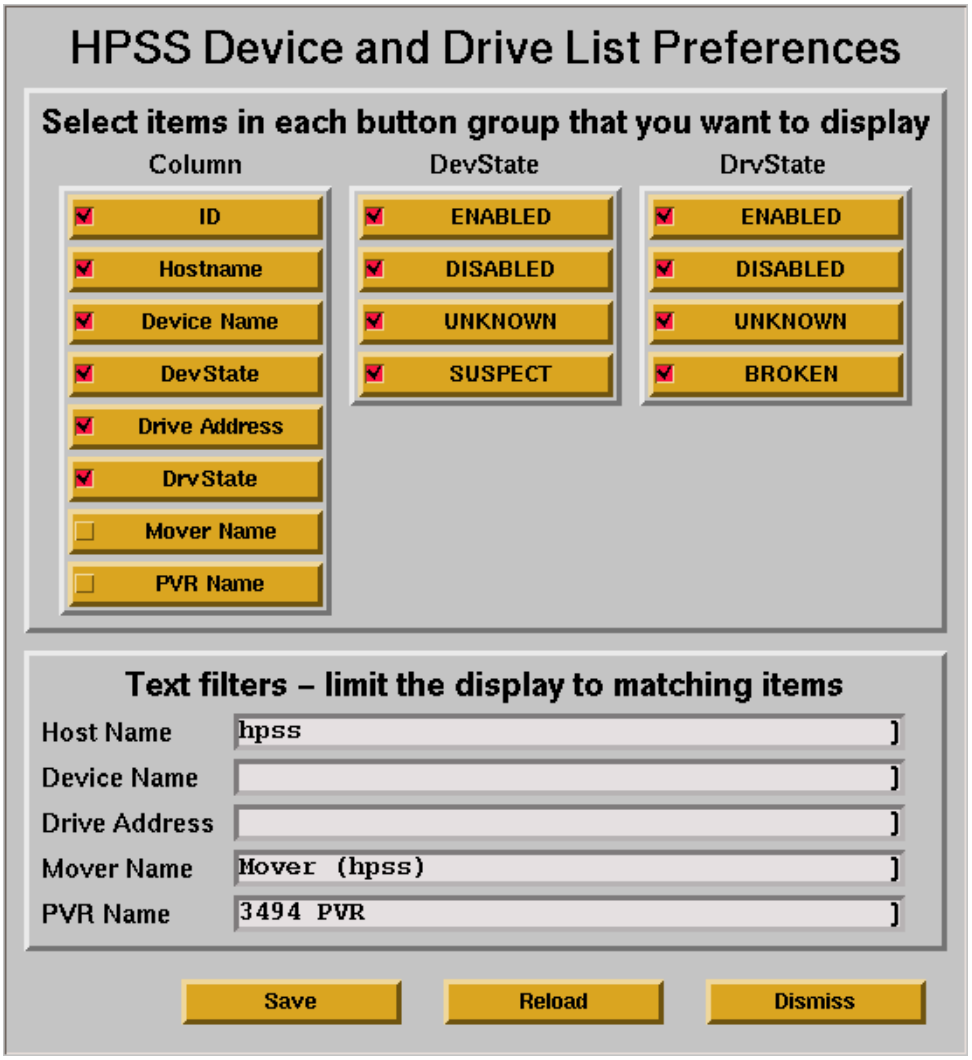


Figure 6-24 HPSS Device and Drive List Preference Window

The HPSS Devices and Drives window allows the user to view the data for all configured devices and drives. It also allows the user to control the devices and drives. SSM reports the current states for a device / drive on the HPSS Devices and Drives window as follows:

1. **DevState**. The current operational state of the device as viewed by the HPSS Mover. Possible states are:

   - ENABLED—The device is working normally.
   - SUSPECT—The Mover had detected errors on the device, but it is still functioning.
   - DISABLED—The device has been administratively made inaccessible to HPSS.
   - UNKNOWN—The device's state is unknown to SSM.

2. **DrvState**. The current operational state of the drive as viewed by the PVL. Possible states are:

   - ENABLED—The drive is operational.
   - DISABLED—The drive is not available for use. This can occur if an SSM user disabled the drive through SSM, or an error occurred when mounting or dismounting a cartridge on that drive.
   - BROKEN —This value will only occur when the PVL is terminating because of a fatal error condition.
   - UNKNOWN—The drive's state is unknown to SSM.

Refer to the window's help file for more information on the individual fields as well as the supported operations available from the window.

Using the PVL Drive Information Window

From the HPSS Devices and Drives window (Figure 6-23), select the appropriate device / drive and click on the **Info...** button from the **Drive** button group.

The PVL Drive Information window will be displayed as shown in Figure 6-25. The PVL Drive Information window is typically used to lock and unlock a drive. It is also used to determine which volume is mounted on the drive when the drive reports a mount error condition. Refer to the PVL Drive Information window's help file for more information on the individual fields as well as supported operations available from the window.

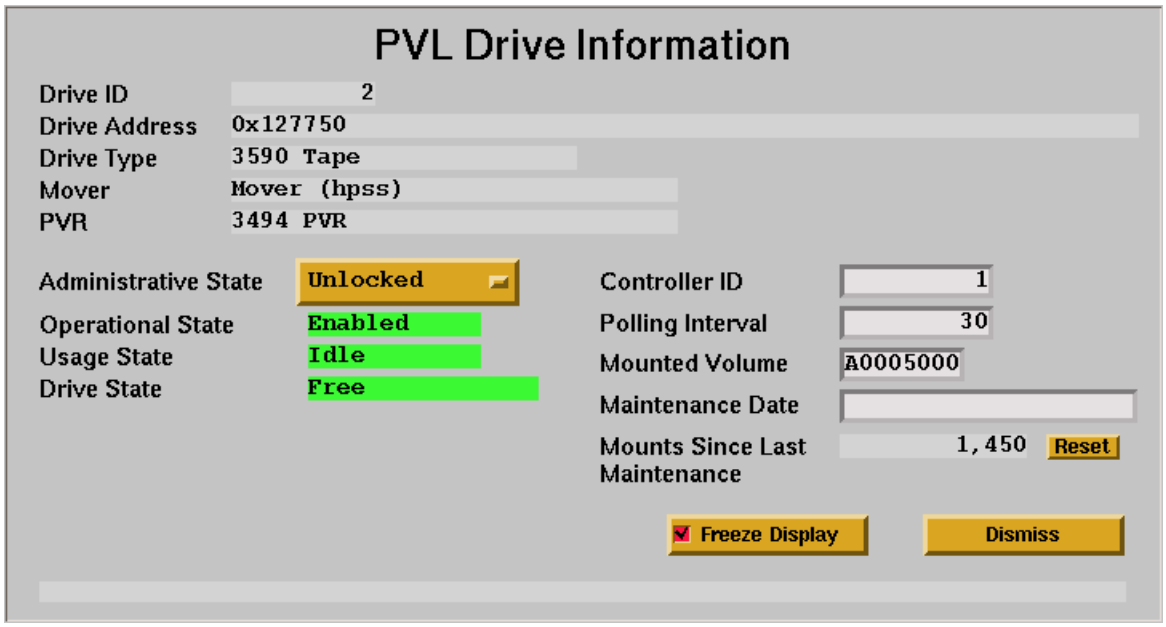


Figure 6-25 PVL Drive Information Window

Using the Mover Device Information Window

From the HPSS Devices and Drives window (Figure 6-23), select the appropriate device / drive and click on the **Info...** button from the **Device** button group.

The Mover Device Information window will be displayed as shown in Figure 6-26. The Mover Device Information window reports the current statistics for the device. In addition to viewing the mover device data, the Mover Device Information window is used to lock and unlock a mover device. It is also used to determine the work load history of the device since the startup of the controlling Mover. In addition, it can be used to control the I/O aspects of the device.

Refer to the window's help file for more information on the individual fields as well as the supported operations available from the window.

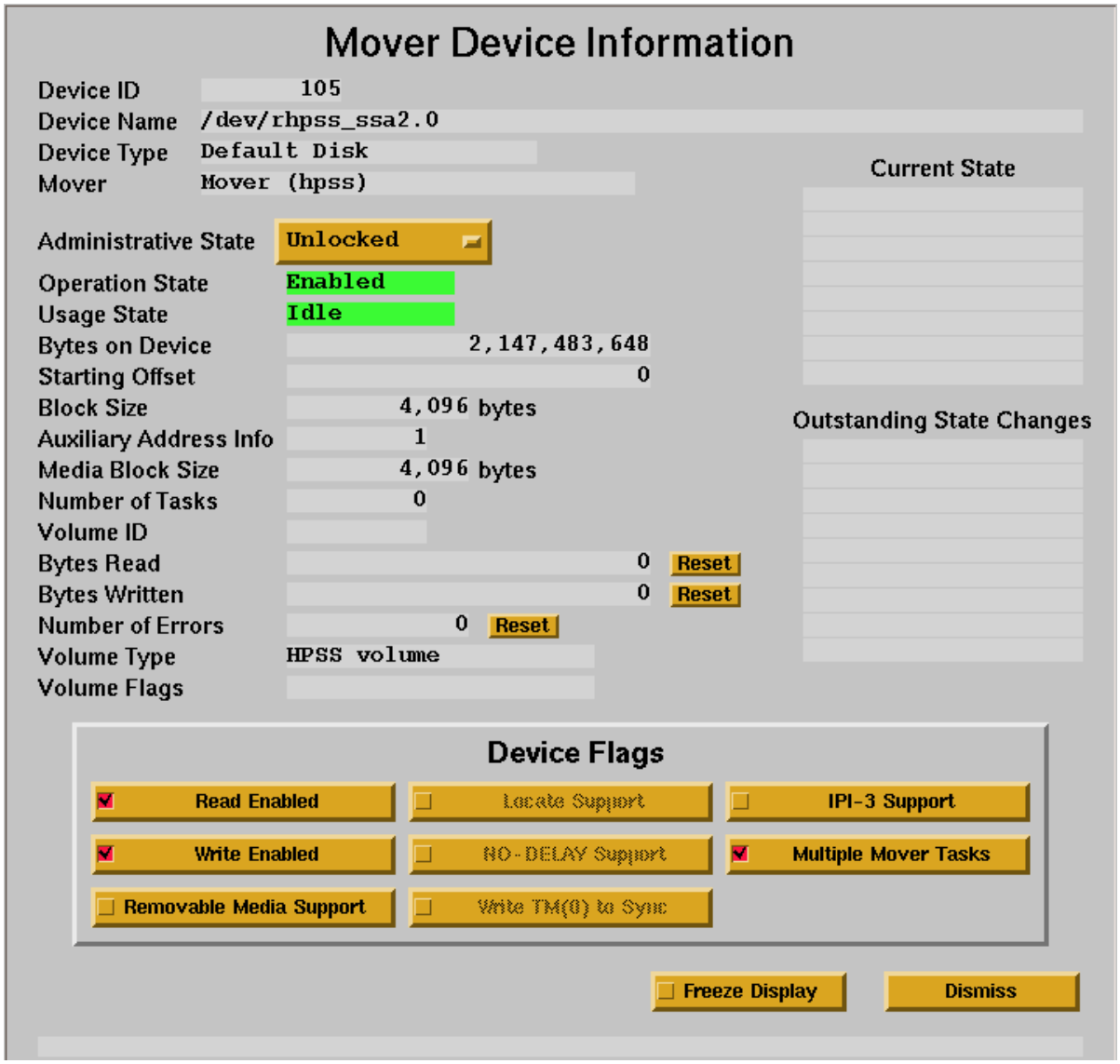


Figure 6-26 Mover Device Information Window

6.6.2 *Adding a New Drive*

A previously unknown drive can be configured into HPSS. After a drive has been configured, its state can be changed (unlocked / locked) using the state change instructions described in Section 6.6.5 below.

Before adding a tape drive to HPSS, the administrator must first fully configure the tape hardware and driver software into the host platform and operating system. For disk drives, the raw disk must first be defined in the operating system. Note that using the block special file name in the device configuration may result in a large performance degradation.

In the current release of HPSS, both the PVL and all executing Movers must be shut down while a new drive

is added to the HPSS configuration. Because of this, all additions should be scheduled during a time when server interruption will have the least impact on HPSS users. If the PVL and the controlling Mover are already running, follow the instructions in Section 6.3.4 to shut down the PVL and Mover. Verify that these servers are no longer running before attempting to configure the new drive.

To add a new drive, refer to Section 5.7 for information. Once the device / drive configuration is created, follow the instructions in Section 6.3.2 to start up the PVL and the controlling Mover. These servers will read the new device / drive configuration and initialize it for services. At this point, the drive's Administrative State must be unlocked before it is available. Refer to Section 6.6.5.1 for instructions on how to unlock a drive.

6.6.3 Deleting a Drive

A drive can be deleted from the HPSS configuration. If the intent is only to take a drive out of service for a finite period of time, the administrator should follow the instructions in Section 6.6.5.2 to lock the drive from being used rather than permanently removing the drive as described in this section.

In the current release of HPSS, both the PVL and the Mover that control the tape drive being deleted must be shut down while the drive is removed from the HPSS configuration. Because of this, all deletions should be scheduled during a time when server interruption will have the least impact on HPSS customers. If the PVL and the controlling Mover are already running, follow the instructions in Section 6.3.4 to shut them down. Verify that these servers are no longer running before attempting to delete the drive.

From the HPSS Devices and Drives window (Figure 6-23), select the desired device and drive entry then click on the **Configure...** button. The Mover Device and PVL Drive Configuration window (Figure 5-35) will be displayed. Verify that the correct device / drive configuration is displayed on the window. Click on the **Delete** button to delete the configuration.

Once the drive deletion is accomplished, HPSS will no longer be able to use the drive's resources. The PVL and Mover can now be restarted. Refer to Section 6.3.2 for more information.

6.6.4 Changing a Drive Configuration

From the HPSS Devices and Drives window (Figure 6-23), select the desired device and drive entry then click on the **Configure...** button. The Mover Device and PVL Drive Configuration window (Figure 5-35) will be displayed. Modify the device and drive data as appropriate. Click on the **Update** button to change the configuration.

In the current release of HPSS, the PVL and the associated Mover must be shut down while the drive is changed in the HPSS configuration. Because of this, all modifications should be scheduled during a time when server interruption will have the least impact on HPSS customers. If the PVL and the Movers are already running, follow the instructions in Section 6.3.4 to shut them down. Verify that these servers are no longer running before attempting to update the drive.

In addition to changing the device and drive configuration through the mover Device and PVL Drive window, the Mover Device Flags, Controller ID and the Polling Interval can be temporarily changed to affect only the current PVL and Mover execution.

To change the Mover Device flags, bring up the Mover Device Information window (Figure 6-26) and update the desired flags. To change the Controller ID or Polling Interval field, bring up the Drive Information window (Figure 6-25) and update the desired fields. Note that these changes affect only the current executions of the PVL and the Mover. The servers will use their current configuration data when restarted.

6.6.5 Changing a Drive State

A drive's administrative state can be changed to **Unlocked** or **Locked** to control whether HPSS can use a drive. Changing a drive's state can be performed from the HPSS Devices and Drives window.

6.6.5.1 Unlocking a Drive

A drive's administrative state can be unlocked to allow HPSS to make use of a drive. Before unlocking a drive, ensure that its hardware is functional and is fully configured into its host's operating system and into HPSS. Configuration of an HPSS drive is described in Section 6.6.2.

From the HPSS Devices and Drives window (Figure 6-23), select the desired device / drive entries and then click on the **Unlock** button from the **Drive** button group. The drive can also be unlocked by bringing up the PVL Drive Information window and set its **Administrative State** to **Unlocked**.

6.6.5.2 Locking a Tape Drive

A drive can be locked to disallow it from being used by HPSS. Changing a drive's state to locked will ensure that the drive will not be used for new mounts, but it will not cause the dismount of any cartridges currently on the drive. The drive will be unloaded when the current client using the drive completes and dismounts.

From the HPSS Devices and Drives window (Figure 6-23), select the appropriate device / drive entry and then click on the Drive **Lock** button. The drive can also be locked by bringing up the PVL Drive Information window and set its **Administrative State** to **Locked.**

The Mover devices do not need to be locked or unlocked.

6.6.5.3 Repairing the State of a Device/Drive

A drive can enter an error or suspect state as viewed by the PVL, Mover, or both. After a drive has entered one of the abnormal states, it can be reset to return it to a normal state.

From the HPSS Devices and Drives window (Figure 6-23), select the appropriate device / drive entry and then click on the appropriate **Mark Repaired** button.Another way to repair the device / drive state is to bring up the Drive Information window (Figure 6-25) and the Mover Device Information window (Figure 6-26), change the Administrative State to **Mark Repaired**.

Repairing the state of a device/drive is only an instruction to the server to reset the device/drive state value. It does not correct any underlying problems that might still exist. Rather, it is a means by which the administrator can notify the server that the underlying problem has been addressed. It is used for problems that the server cannot fix by itself or problems that have been fixed unbeknownst to the server, such as a drive being offline or a tape getting stuck in a drive.

6.6.5.4 Resetting Drive Statistics

Both the PVL and Mover maintain statistics related to a drive. The PVL maintains a count of the number of times a cartridge is mounted on the drive. The Mover maintains counts of the number of errors encountered and the number of bytes read and written on the drive. These values can all be reset to zero by selecting the **Reset** button located next to the statistic field on the appropriate window.

6.7 Monitoring HPSS Information

SSM allows the authorized user to monitor, view and update the following HPSS Information:

- Server Information (See Section 6.3.1.3)
- Storage Classes (See Section 6.4.2.2)
- Devices and Drives (See Section 6.6.1)
- PVL Jobs (See Section 6.7.1)
- PVL Tape Mount Requests (See Section 6.7.2)
- PVL Tape Check-In Requests (See Section 6.7.13)
- PVL Volumes (See Section 6.7.3)
- PVR Cartridges (See Section 6.7.4)
- Storage Server Volumes (See Section 6.7.5)
- Security Information (See Section 6.7.6)
- LS Statistics (See Section 6.7.7)
- NFS Statistics (See Section 6.7.8)
- DMAP Gateway Fileset Statistics (See Section 6.7.9)
- Log Files (See Section 6.7.10)
- Bitfiles (See Section 6.7.11)

Since SSM will set up data change registration operations by default when an information window is open, these windows should be closed when they are no longer needed. This will prevent unnecessary notification message traffic to unused or ignored SSM windows.

6.7.1 Monitoring PVL Jobs

From the HPSS Health and Status window (Figure 6-1), click on the **Monitor** menu and select the **PVL Jobs** option.

The PVL Job Queue window will be displayed as shown in Figure 6-27. This window shows all outstanding jobs in the PVL. From this window, the user can issue a request to view more information for a particular PVL job or cancel a PVL job. Each PVL job represents a mount (or series of mounts for a striped mount) that is in some state of completion ranging from "pending due to resource contention" to "in the process of being dismounted."

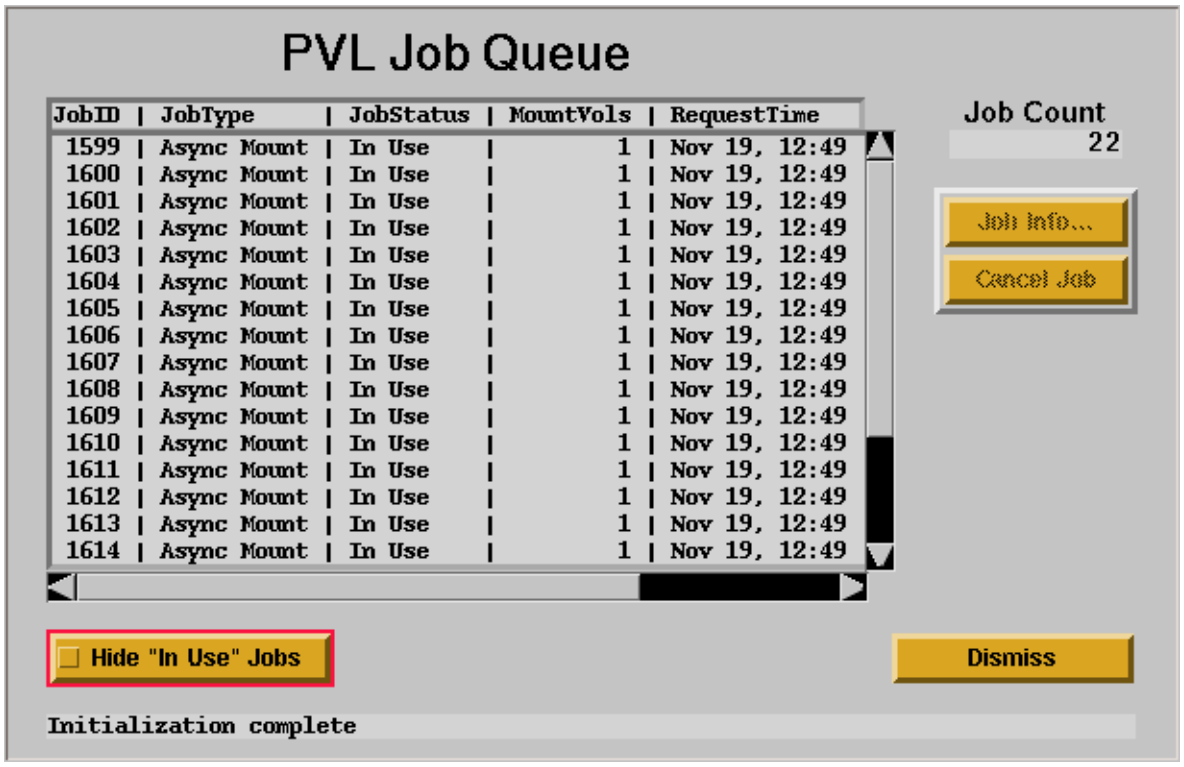


Figure 6-27 PVL Job Queue Window

JobStatus of **In Use** denotes currently mounted disk and tape mount. These jobs can be hidden on this display by toggling the **Hide "In Use" Jobs** button. Unless hidden in this manner, a disk mount job for each mounted disk volume will be displayed as long as Disk Storage Servers are active. Note that this also hides the currently mounted tape jobs.

To cancel a PVL job, select the desired job entry and click on the **Cancel Job** button. A confirmation window will be popped up to require the user to confirm the cancel request. Click on the **Confirm** button to initiate the cancel request.

Canceling a PVL job should be done with great care. The Tape Storage Server reserves the device a cartridge is mounted on while reading/writing. Canceling the PVL job which represents the mounted cartridge will result in a dismount request issued to the reserved device. The PVL generated dismount request (issued periodically) will fail (generating an alarm) until the Tape Storage Server releases its reservation of the device. Canceling a PVL job will NOT result in the termination of Storage Server request and the cartridge dismount will NOT occur until the Tape Storage Server releases its reservation on the device.

Refer to the PVL Job Queue window's help file for more information on the individual fields as well as the supported operations available from the window.

To view more detailed PVL job information, select the desired job entry and click on the **Job Info...** button. The PVL Request Information window will be displayed as shown in Figure 6-28. Refer to the window's help file for more information on the fields and information displayed on the window.

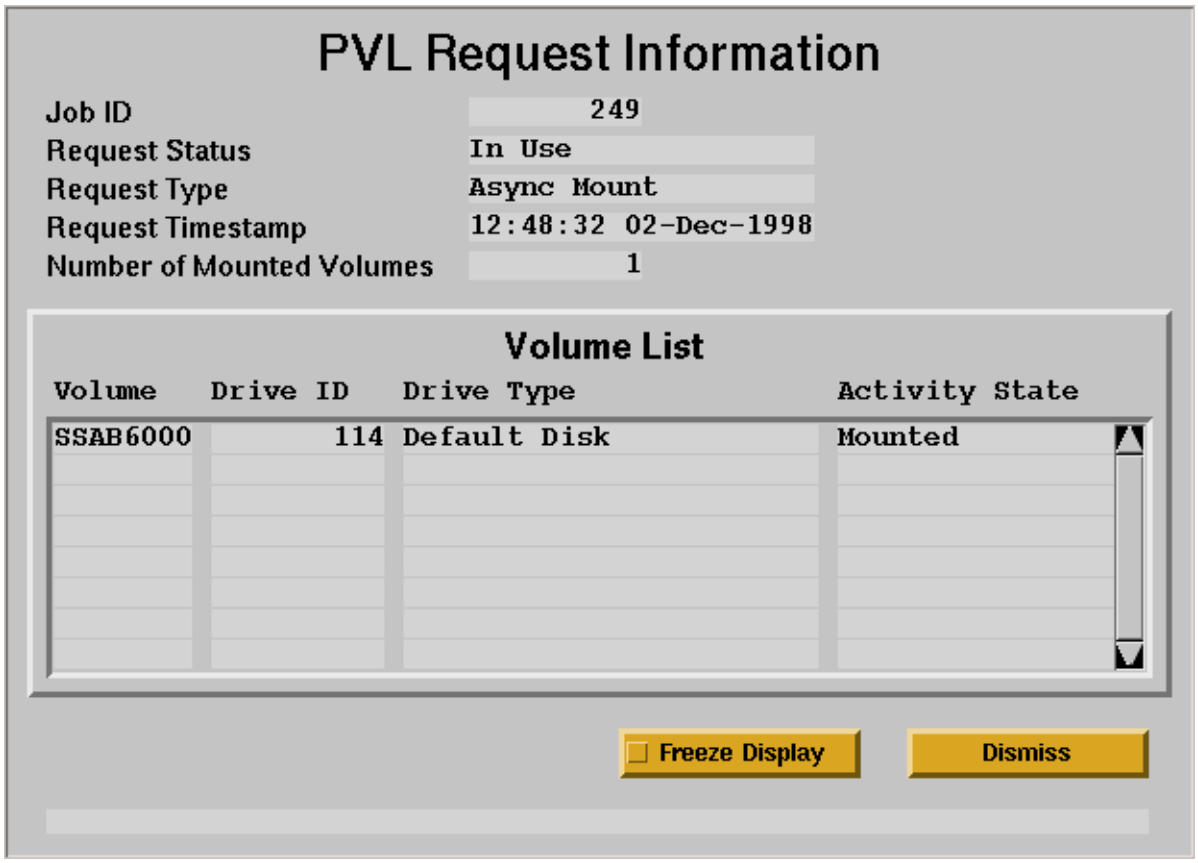


Figure 6-28 PVL Request Information Window

6.7.2 Monitor the PVL Tape Mount Requests

All HPSS tape mount requests, including robotic tape mounts and operator tape mounts, will be displayed on the Tape Mounts window. Outstanding tape mounts, along with the associated PVR, are displayed in chronological order. The operator will typically use this window to help in the process of hand mounting cartridges for a vault operation or to detect a robot that has a hardware failure.

Using the Tape Mount Requests Window

From the HPSS Health and Status window (Figure 6-1), click on the **Monitor** menu and select the **Tape Mount Requests** option. The Tape Mount Requests window will be displayed as shown in Figure 6-29. Refer to the window's help file for more information on the individual fields as well as supported operations available from the window.

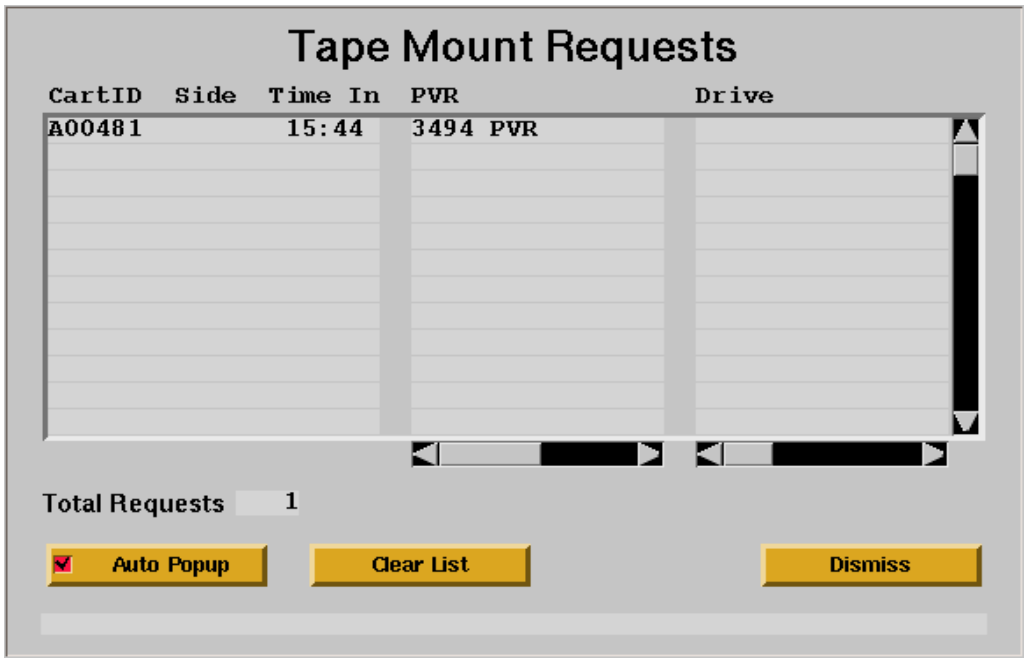


Figure 6-29 Tape Mount Requests Window

The **Clear List** button on the Tape Mount Requests window can be used to remove all mount requests displayed, including currently pending mounts. This option can be used to remove stale mount requests (cartridges mounted but not removed from the window) from being displayed. Pending mounts will reappear in the window as the PVR periodically retries the mounts.

Mount requests will no longer be displayed once they are satisfied. However, at times, the window may display an old mount request due to lost notifications. Click on the Clear List button to refresh the window.

The maximum number of tape mounts that can be displayed on the Tape Mounts window is 100. When this limit is exceeded, some mount notifications will be lost. SSM will inform the user when this condition occurs.

When a new mount notification is received and the Tape Mounts window is not currently displayed, SSM will bring up the window to notify the user of the new mount notification. If this is not desired, the Auto Popup button can be turned OFF, after which SSM will not automatically bring up the window. The Auto Popup button can be turned back ON by re-selecting the Tape Mount Requests option from the Monitor menu of the Health and Status window.

6.7.3 Monitoring the PVL Volumes

The PVL Volume Information window allows the SSM user to view the data for an imported volume. Before using the window, the user should know the 6-character label of the PVL volume. The **dumppv_pvl** utility described in Appendix I can be used to list all the volumes being managed by the PVL.

Using the PVL Volume Information Window

From the Health and Status window (Figure 6-1), click on the **Monitor** menu, select the **HPSS Objects** and click on the **Cartridges and Volumes** option. The Identify Cartridge or Volume window (Figure 6-30) will appear that allows the operator to enter the 6-character volume label and the partition information. Press **Enter** after typing the requested data then click on the **PVL Volume** button.

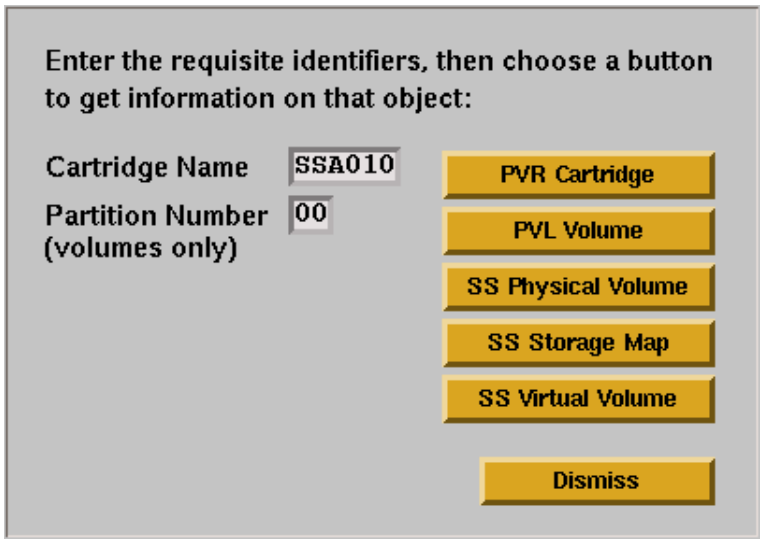


Figure 6-30 Identify Cartridge or Volume Window

If the volume has been imported into HPSS, the PVL Volume Information window will be displayed as shown in Figure 6-31. The window is used primarily to determine the import status of a volume. If the volume is imported, the user can also determine whether the volume is allocated, who the allocated owner of the volume is, and what PVR Server is managing the volume.

Refer to the window's help file for more information on the individual fields as well as the supported operations available from the window.

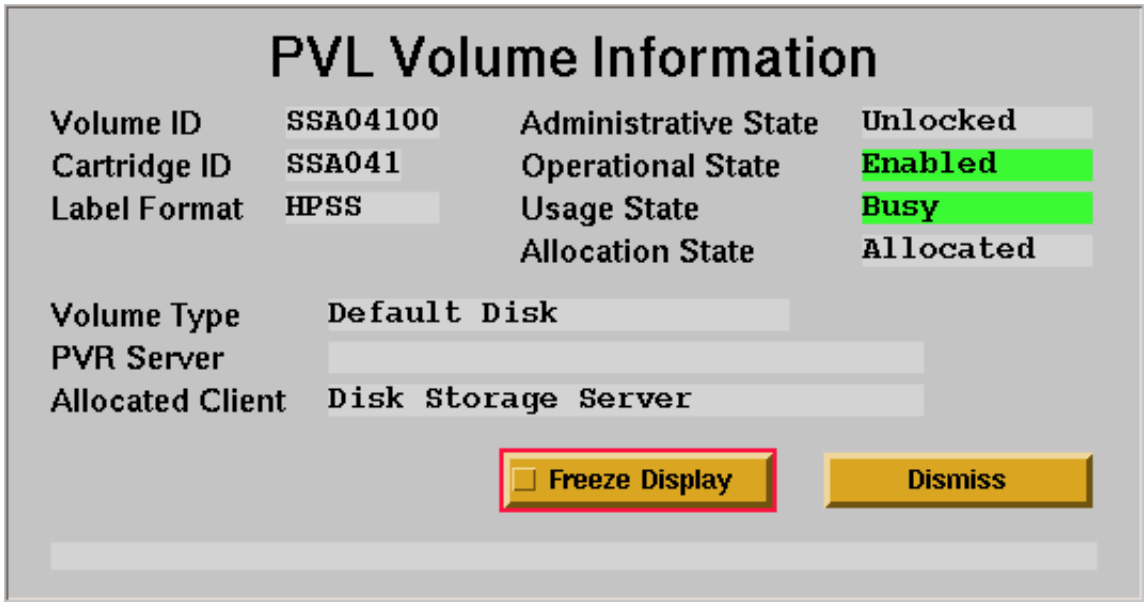


Figure 6-31 PVL Volume Information Window

6.7.4 Monitoring the PVR Cartridges

The PVR Cartridge Information window allows the SSM user to view the data about a cartridge managed by one of the HPSS PVRs. Before using the window, the user should know the 6-character label of the cartridge. The **dumppv_pvr** utility described in Appendix I can be used to list all the cartridges being managed by the PVR.

Using the PVR Cartridge Information Window

From the Health and Status window (Figure 6-1), click on the **Monitor** menu, select the **HPSS Objects** and click on the **Cartridges and Volumes** option. The Identify Cartridge or Volume window (Figure 6-30) will appear that allows the operator to enter the 6-character volume label and the partition information. Press **Enter** after typing the requested data then click on the **PVR Cartridge** button.

If the cartridge has been imported into HPSS, the PVR Cartridge Information window will be displayed as shown in Figure 6-32. This window displays the current information about the cartridge. Note that the **Location Type** fields cannot be determined for certain types of robots, and **Slot Unit**, **Slot Panel**, **Slot Row**, and **Slot Column** will be displayed as zero. If it is necessary to locate a cartridge in one of these robots, the robot's own operator interface must be used.

The only operations possible from this window are the resetting of two fields. If the cartridge is maintained, for example, by retensioning the entire cartridge, the **Mounts Since Maintenance** field can be reset. The **Maintenance Date** field can also be reset. Note that resetting one field does not affect the other.

Refer to the window's help file for more information on the individual fields as well as the supported operations available from the window.

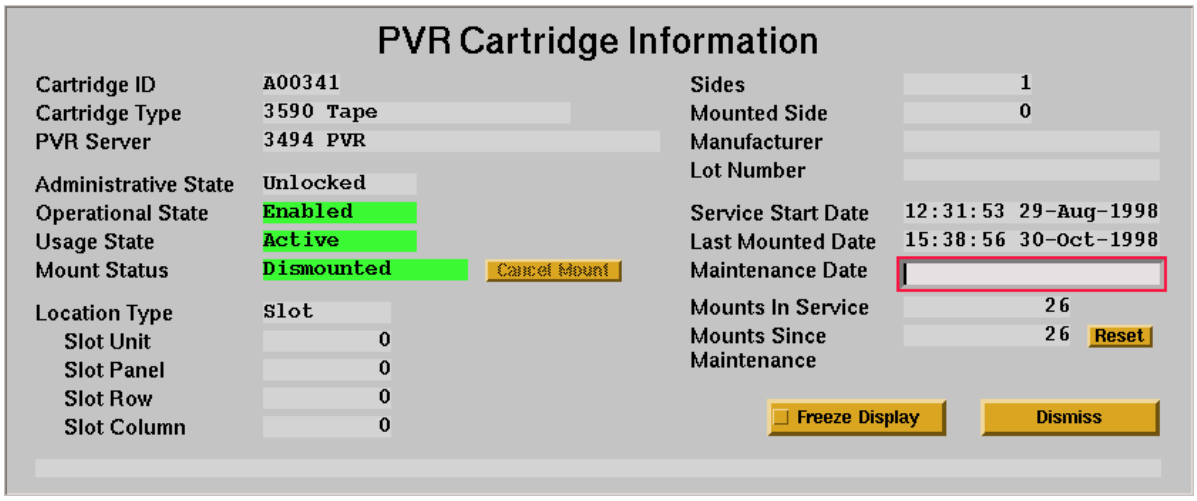


Figure 6-32 PVR Cartridge Information Window

6.7.5 *Monitoring Storage Server Volumes*

The Storage Server allows the SSM user to view and / or update its volume information that defines the HPSS storage space. The SSM user can monitor the following information:

- Physical Volumes
- Virtual Volumes
- Storage Maps
- Current Tape Storage Segments

The above information can be viewed and updated through the Physical Volume Information window, the Virtual Volume Information window, the Storage Map Information window and the Storage Segment Information window, respectively.

Since the SS storage segment has a non-human-readable name, the information for this SS object can only be requested through the Physical Volume Information window.

Before using the SS volume information windows, the user must know the 6-character naming label of the cartridge. The **dump_sspvs** utility described in Appendix I can be used to find the names of all volumes currently allocated to the Storage Server.

6.7.5.1 *Monitoring Storage Server Physical Volumes*

The Storage Server allows the SSM user to view the current status of Storage Server physical volumes. It also allows the user to control the availability of the volumes through the use of the Administrative State fields.

Using the Physical Volume Information Window

From the Health and Status window (Figure 6-1), click on the **Monitor** menu, select the **HPSS Objects** and click on the **Cartridges and Volumes** option. The Identify Cartridge or Volume window (Figure 6-30) will appear that allows the operator to enter the 6-character volume label and the partition information. Press **Enter** after typing the requested data then click on the **SS Physical Volume** button.

The Physical Volume Information window will be displayed as shown in Figure 6-33. This window is typically used to determine and control the usage and the availability of a physical volume. In addition, it allows the user to determine the virtual volume that the physical volume is a part of.

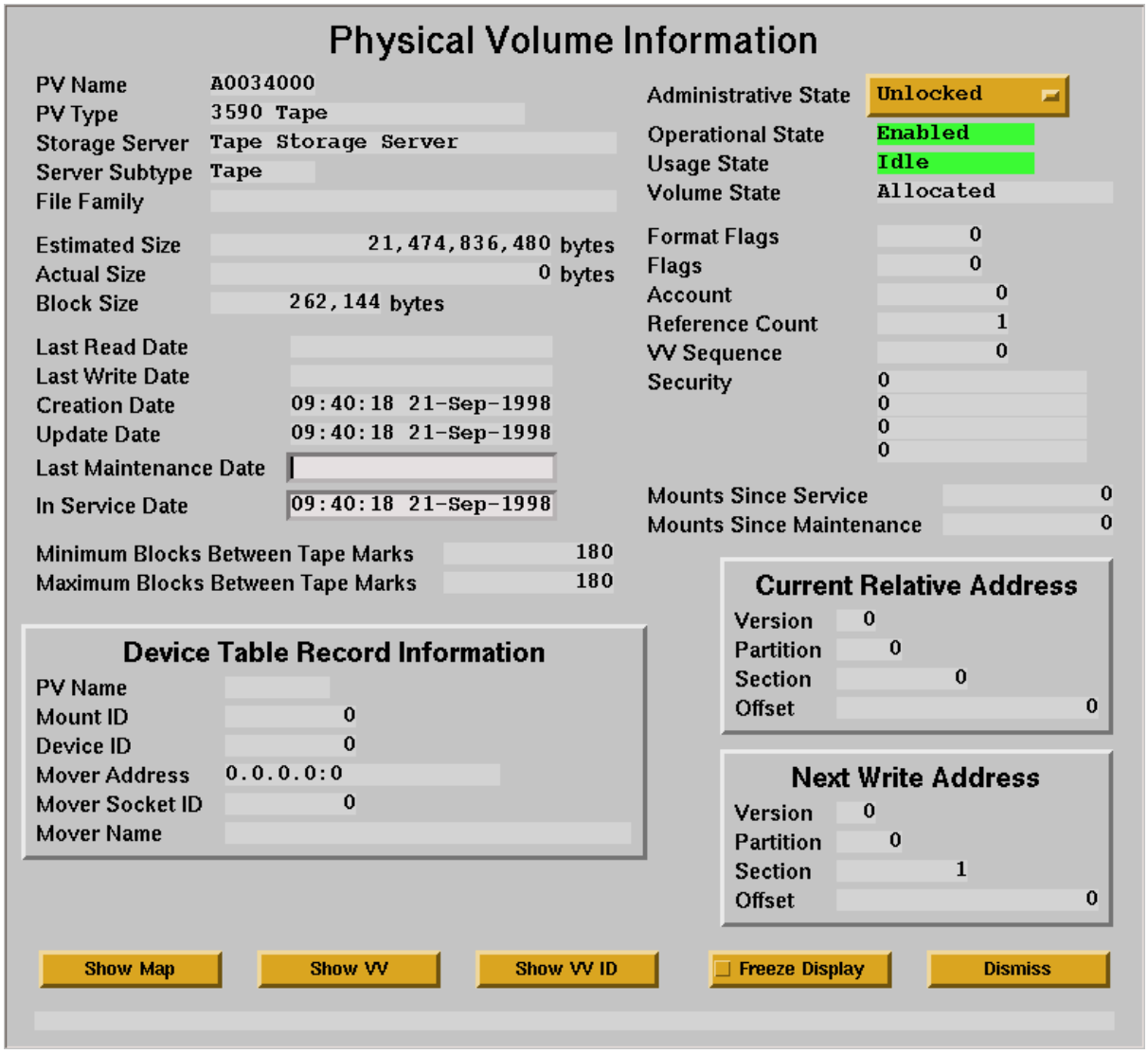


Figure 6-33 Physical Volume Information Window

The Administrative State field in both the tape and disk Physical Volume Information windows can be used to make a tape or disk PV unavailable to HPSS. Selecting either **Locked** or **Shut Down** from the option list will cause all subsequent attempts to read, write, or mount the volume to fail. Selecting **Unlocked** from the option list will unlock the PV, allowing read, write, and mount requests to proceed

Setting the Administrative State of a disk or tape PV to Locked or Shut Down is a possible way of making the individual PV unavailable to the system while leaving the remaining PVs available. However, since most operations on PVs are actually operations on the VV of which the PV is a part, and since the Storage Servers do not perform I/O operations on individual PVs, the recommended practice is to modify the Administrative State of the entire VV and/or storage map rather than an individual PV. Locking a PV will not prevent the

storage server from selecting the volume for writing, if the volume is writable. See Section 6.7.5.2 and 6.7.5.3 for details.

If the **Flags** field of a disk Physical Volume Information window has the 0x08 bit set, the disk physical volume has been marked "off-line" by the Disk Storage Server. The server marks the disk "off-line" when it receives an error from the PVL while attempting to mount the disk during start-up. Any attempt to read or write the disk while "off-line" will receive an error. The disk can only be brought back "on-line" by correcting the mount problem and restarting the Disk Storage Server.

Refer to the Physical Volume Information window's help file for more information on the individual fields as well as the supported operations available from the window.

6.7.5.2 Monitoring Storage Server Virtual Volumes

The Storage Server allows the SSM user to view the current status of storage server virtual volumes. It also allows the user to control the availability of the volumes through the use of the volumes' Administrative State fields.

Using the Virtual Volume Information Window

From the Health and Status window (Figure 6-1), click on the **Monitor** menu, select the **HPSS Objects** and click on the **Cartridges and Volumes** option. The Identify Cartridge or Volume window (Figure 6-30) will appear that allows the operator to enter the 6-character cartridge label of one of the physical volumes that belongs to the virtual volume and the partition information. Press **Enter** after typing the requested data then click on the **SS Virtual Volume** button.

The Virtual Volume Information window will be displayed as shown in Figure 6-34. This window displays the Virtual Volume Information for a virtual volume that the specified physical volume is a part of. The window is typically used to determine and control the usage and availability of a set of physical volumes that made up the virtual volume.

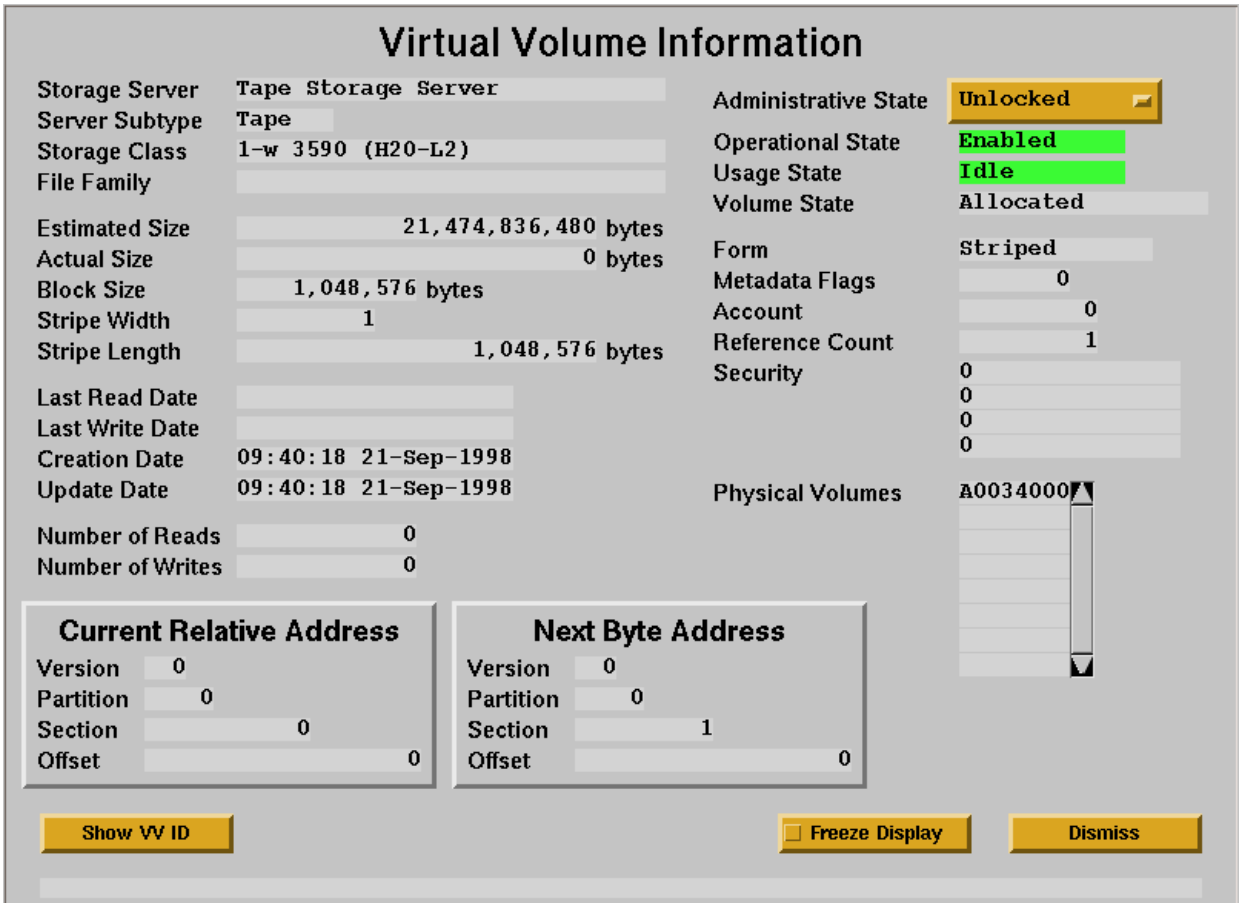


Figure 6-34 Virtual Volume Information Window

The Administrative State field in both the tape and disk Virtual Volume Information windows can be used to make a tape or disk VV unavailable to HPSS. Selecting either **Locked** or **Shut Down** from the option list will cause all subsequent attempts to read, write, or mount the volume to fail. Selecting **Unlocked** from the option list will unlock the VV, allowing read, write, and mount requests to proceed.

Setting the Administrative State of a VV to Locked or Shut Down is the recommended way of making the VV and all of its component PVs unavailable to HPSS. Locking individual PVs is not recommended. A locked VV cannot be mounted, read or written, but, a locked VV can be selected for writing, if the volume is writable. To set a VV to "read-only" mode, use the Storage Map interface described in Section 6.7.5.4.

If the **Metadata Flags** field of a disk Virtual Volume Information window has the 0x04 bit set, the disk virtual volume has been marked "off-line" by the Disk Storage Server. The server marks the VV "off-line" when it receives an error from the PVL while attempting to mount one or more of the VV's disks (PVs) during server start-up. Any attempt to read or write the VV while "off-line" will receive an error. The VV can only be brought back "on-line" by correcting the mount problem and restarting the Disk Storage Server. If more than one of the underlying PVs is "off-line," the VV will remain "off-line" until all PVs are "on-line."

Refer to the Virtual Volume Information window's help file for more information on the individual fields as well as the supported operations available from the window.

6.7.5.3 Monitoring Storage Server Tape Storage Maps

The Tape Storage Server allows the SSM user to view the status of storage maps associated with virtual volumes. It also allows the user to control the availability of a map through the map's Administrative State.

Using the Tape Storage Map Information Window

From the Health and Status window (Figure 6-1), click on the **Monitor** menu, select the **HPSS Objects** and click on the **Cartridges and Volumes** option. The Identify Cartridge or Volume window (Figure 6-30) will appear that allows the operator to enter the 6-character cartridge label of one of the physical volumes that belongs to the virtual volume and the partition information. Press **Enter** after typing the requested data then click on the **SS Storage Map** button. Note that the **VV Name** field on this window always displayed the first physical volume in the VV set.

The Storage Map Information window will be displayed as shown in Figure 6-35. This window displays the Tape Storage Map Information associated with the virtual volume that the specified physical volume is a part of. The window is typically used to determine and control the usage information and availability of the virtual volume.

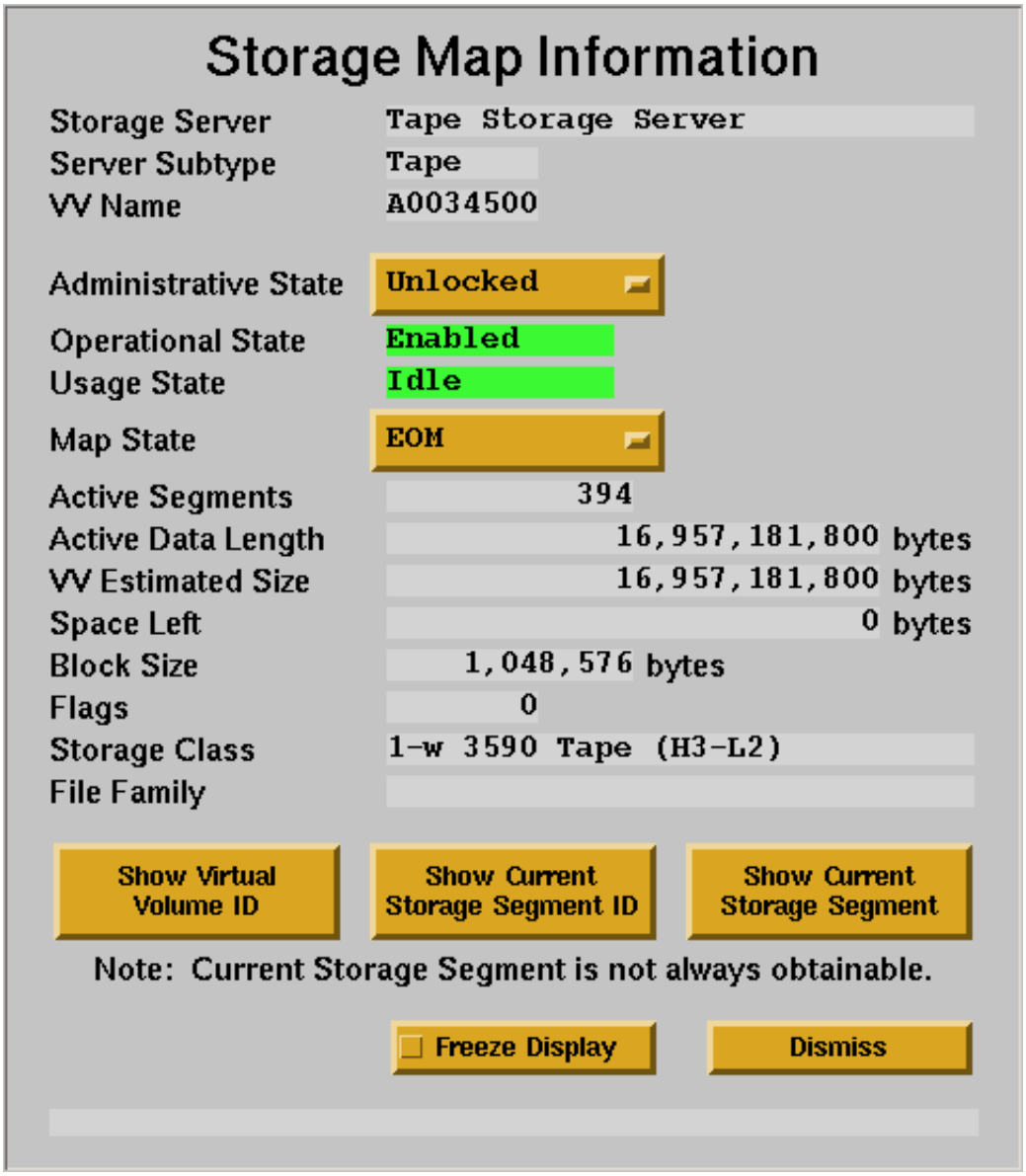


Figure 6-35 Tape Storage Map Information Window

The Administrative State field in the tape Storage Map Information windows can be used to prevent storage space from being allocated on the VV, making the VV "read-only." Selecting either Locked or Shut Down from the Administrative State popup options will cause all subsequent attempts to allocate space on the VV to fail (another VV in the same storage class will be selected, if available). Selecting Unlocked from the popup options will unlock the storage map, allowing allocation requests to proceed.

Setting the Administrative State to Locked or Shut Down on a tape storage map does not prevent the Storage Server from deleting existing storage space from the volume. Tape VVs can be emptied by setting the storage map Administrative State to Locked, then forcing the storage segments on the volume to move to other volumes. This is done with the repack utility. As segments are moved or deleted, the Number of Active Segments will decrease toward zero. When this field reaches zero, the VV is empty.

To prevent the system from allocating space, or reading or writing a VV, set the Administrative State of the storage map to Locked (prevents new space allocations), and set the Administrative State of the VV to Locked (prevents mounts, reads and writes). Do not change the MapState to EOM unless the change is intended to be permanent. Once MapState is changed to EOM, it cannot be changed back to Free.

If the Administrative State of a tape storage map is set to **Shut Down** and the tape is being written at the time the change is made, the Administrative State will not change to **Locked** until the write operation completes; otherwise, the state will change immediately. Setting a disk Administrative State to **Shut Down** causes the state to change to **Locked** immediately.

Refer to the Tape Storage Map Information window's help file for more information on the individual fields as well as the supported operations available from the window.

6.7.5.4 Monitoring Storage Server Disk Storage Maps

The Disk Storage Server allows the SSM user to view the status of storage maps associated with virtual volumes. It also allows the user to control the availability of a map through the map's Administrative State.

Using the Disk Storage Map Information Window

From the Physical Volume Information window (Figure 6-33), click on the **Show Map** button.

The Disk Storage Map Information window will be displayed as shown in Figure 6-35. This window displays the Disk Storage Map Information associated with the virtual volume that the specified physical volume is a part of. The window is typically used to determine and control the usage information and availability of the virtual volume.

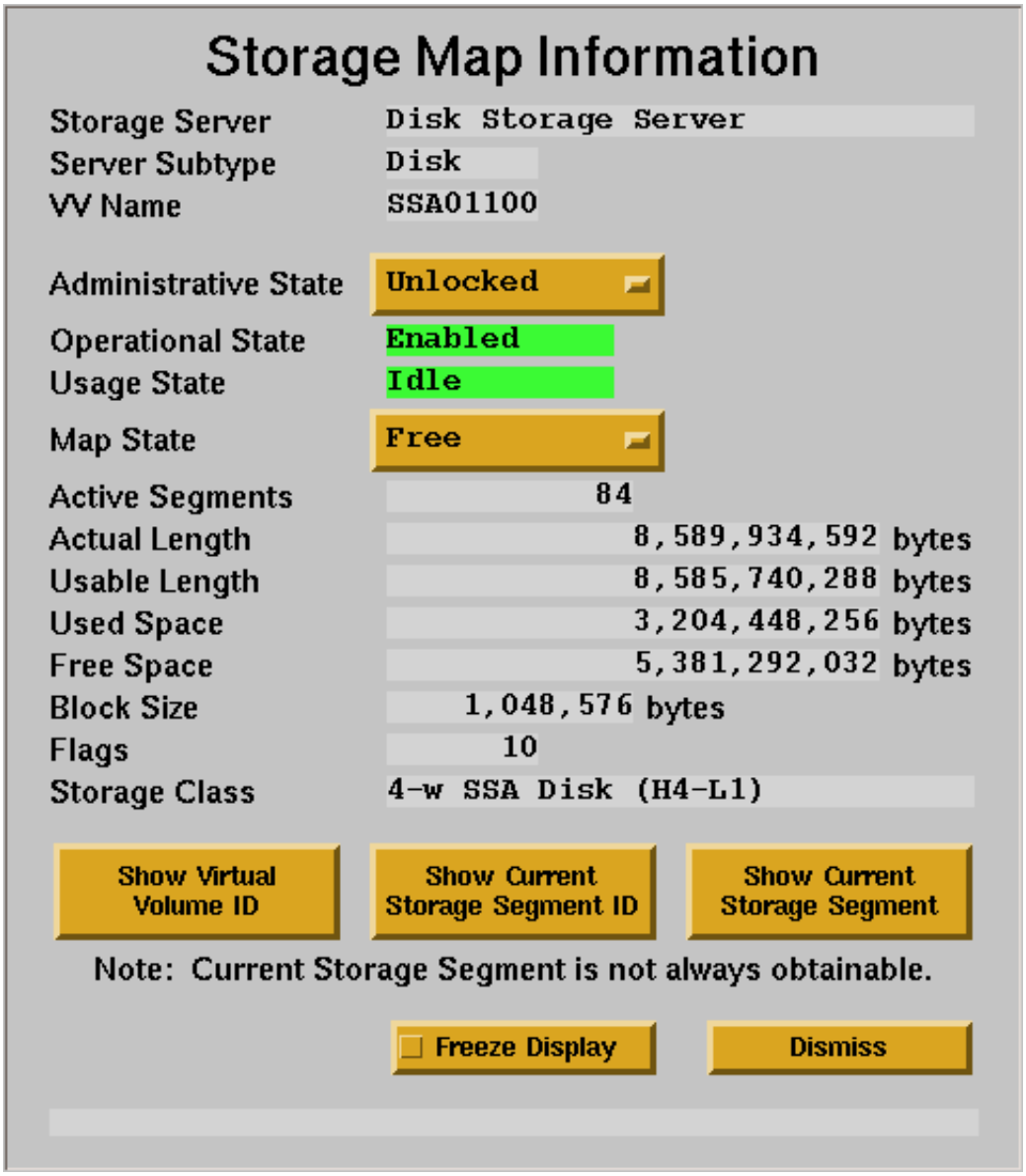


Figure 6-36 Disk Storage Map Information Window

The Administrative State field in the disk Storage Map Information windows can be used to prevent storage space from being allocated on the VV, making the VV "read-only." Selecting either Locked or Shut Down from the Administrative State sub-menu will cause all subsequent attempts to allocate space on the VV to fail (another VV in the same storage class will be selected, if available). Selecting Unlocked from the sub-menu will unlock the storage map, allowing allocation requests to proceed.

Setting the Administrative State to Locked or Shut Down on a disk storage map does not prevent the Storage Server from deleting existing storage space from the volume. Disk VVs can be emptied by setting the storage map Administrative State to Locked, then forcing the storage segments on the volume to move to other volumes. This can be done with MPS by setting the migration and purge parameters so that all files are migrated and purged in the storage class. If an individual VV is to be purged, repack can be used. As segments are moved or deleted, the Number of Active Segments will decrease toward zero. When this field reaches zero, the VV is logically empty.

To prevent the system from allocating space, or reading or writing a VV, set the Administrative State of the storage map to Locked (prevents new space allocations), and set the Administrative State of the VV to Locked (prevents reads and writes).

Setting a disk Administrative State to Shut Down causes the state to change to Locked immediately.

If the **Flags** field of a disk Storage Map Information window has the 0x02 bit set, the disk virtual volume has been marked "off-line" by the Disk Storage Server. For more information, see the discussion of off-line disk VVs in Section 6.7.5.2.

The **Actual Length** field shows the length of the disk VV, in bytes. This value will match the length of the VV as shown on the VV Information window.

The **Usable Length** field shows the length of the disk VV, in bytes, after the space needed by HPSS for disk volume labels is subtracted. This length is the amount of space that is available for use by storage segments. The sum of these values, for all VVs in the storage class, will match the "Total Space" field in the Storage Class Information window.

The **Used Space** field shows the number of bytes in the VV that have been allocated to storage segments. Not all of these bytes may have been written.

The **Free Space** field shows the number of bytes in the VV that are available for assignment to storage segments. The sum of **Free Space** and **Used Space** will equal the **Usable Length**.

Refer to the Storage Map Information window's help file for more information on the individual fields as well as the supported operations available from the window.

6.7.5.5 Monitoring Storage Server Current Tape Storage Segments

The Storage Server allows the SSM user to view the status of the current storage segment associated with a tape virtual volume. Current storage segments are those that are located at the ends of the tape virtual volumes.

Using the Storage Segment Information Window

From the Storage Map Information window (Figure 6-35), click on the **Show Current Storage Segment** button, if it is activated.

The Storage Segment Information window will be displayed as shown in Figure 6-37 if the **Show Current Storage Segment** button was activated. If the button is dimmed (not activated), no storage segment is considered active by the Storage Server for the volume at this time.This is the case for tapes that have never been written, and tapes on which certain routine errors have occurred. Lack of a **Current Storage Segment** is not an error or cause for concern.

Refer to the window's help file for more information on the individual fields as well as the supported operations available from the window.

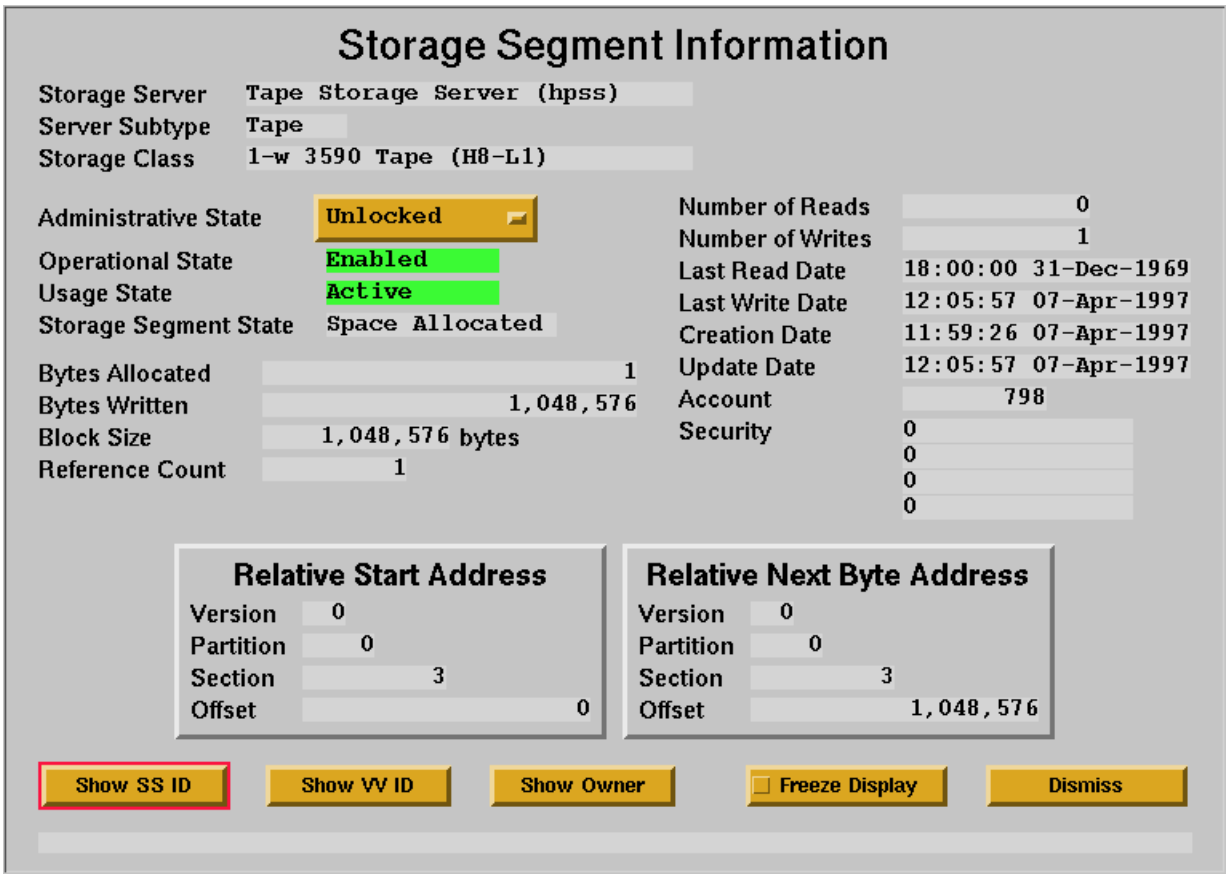


Figure 6-37 Storage Segment Information Window

6.7.6 *Monitoring the HPSS Security Information*

From the From the Health and Status window (Figure 6-1), click on the **Monitor** menu, select the **HPSS Objects** and click on the **Security** option. The HPSS Security Information window will be displayed as shown in Figure 6-38.

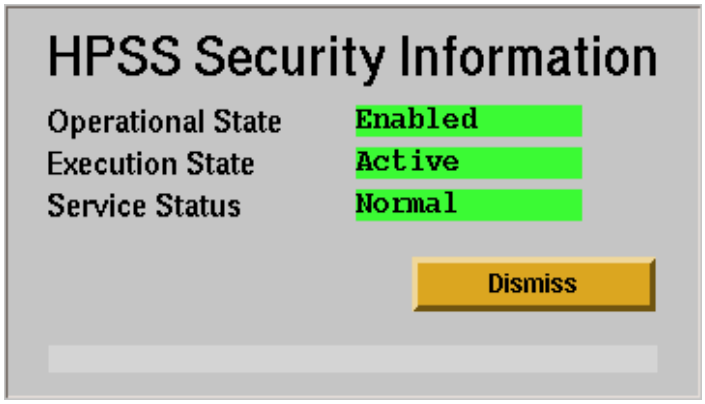


Figure 6-38 Security Information Window

This window allows the user to view the status of the DCE Security components, which are monitored to determine the Security status. The fields in this display are set based on the status of the DCE Security Daemon, the DCE Security Registry Service, and the DCE ACL Manager. If any or all of these DCE components are unavailable, the security state is degraded. The **Operational State** field is set to **Disabled** if all three of the DCE security components are inaccessible. The **Execution State** field is set to **Inactive** if all three of the DCE security components are inaccessible. The **Service Status** field is set to **Minor** if one of the components is inaccessible, **Major** if two are inaccessible, and **Critical** if all three are inaccessible.

Refer to the window's help file for more information on the individual fields as well as the supported operations available from the window.

6.7.7 Monitoring the Location Server Statistics

From the Health and Status window (Figure 6-1), click on the **Monitor** menu, select the **Servers** option. The HPSS Servers window (Figure 6-2) will appear. Click on a Location Server's entry in the list and then click the **Statistics** button. The Location Server Statistics window will be displayed (as show in Figure 6-39).

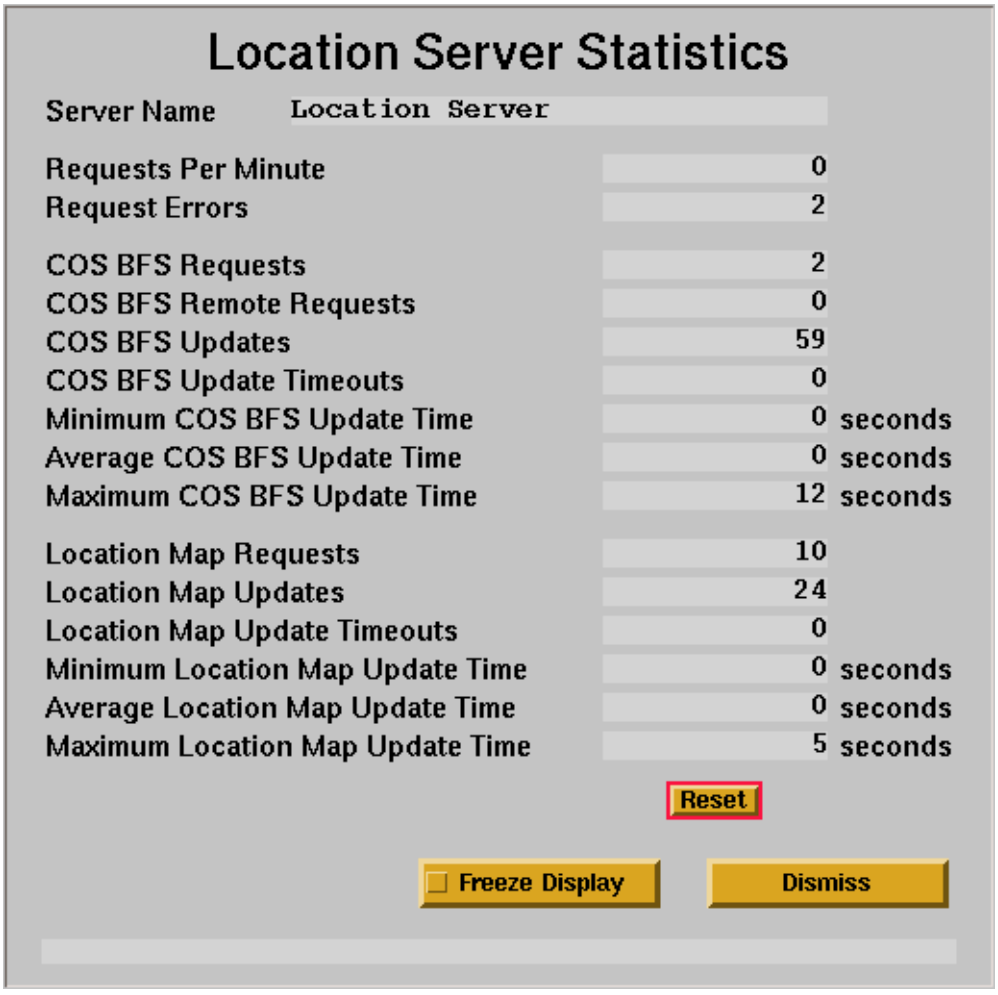


Figure 6-39 Location Server Statistics Window

The Location Server Statistics window identifies the server and displays statistics for that server's request processing and background updating functions.

The first block displays the current request load as well as the total number of errors returned to clients. The maximum number of requests a single Location Server can process in one minute varies depending on the machine load caused by other processes as well as the CPU performance. Generally the Location Server will report warning messages when it becomes loaded.

The next block displays statistics for requests for and background updating of Class of Service information. The following block displays similar information for Location Map requests and background remote Location Server updates. If time-outs occur often, consider increasing the corresponding time-out fields on the Location Policy window.

Refer to the window's help file for more information on the individual fields as well as the supported operations available from the window.

6.7.8 Monitoring the NFS Statistics

From the HPSS Servers window (Figure 6-2), click on the **Statistics...** button from the **Server Info** button group. The NFS Statistics window will be displayed as shown in Figure 6-40.

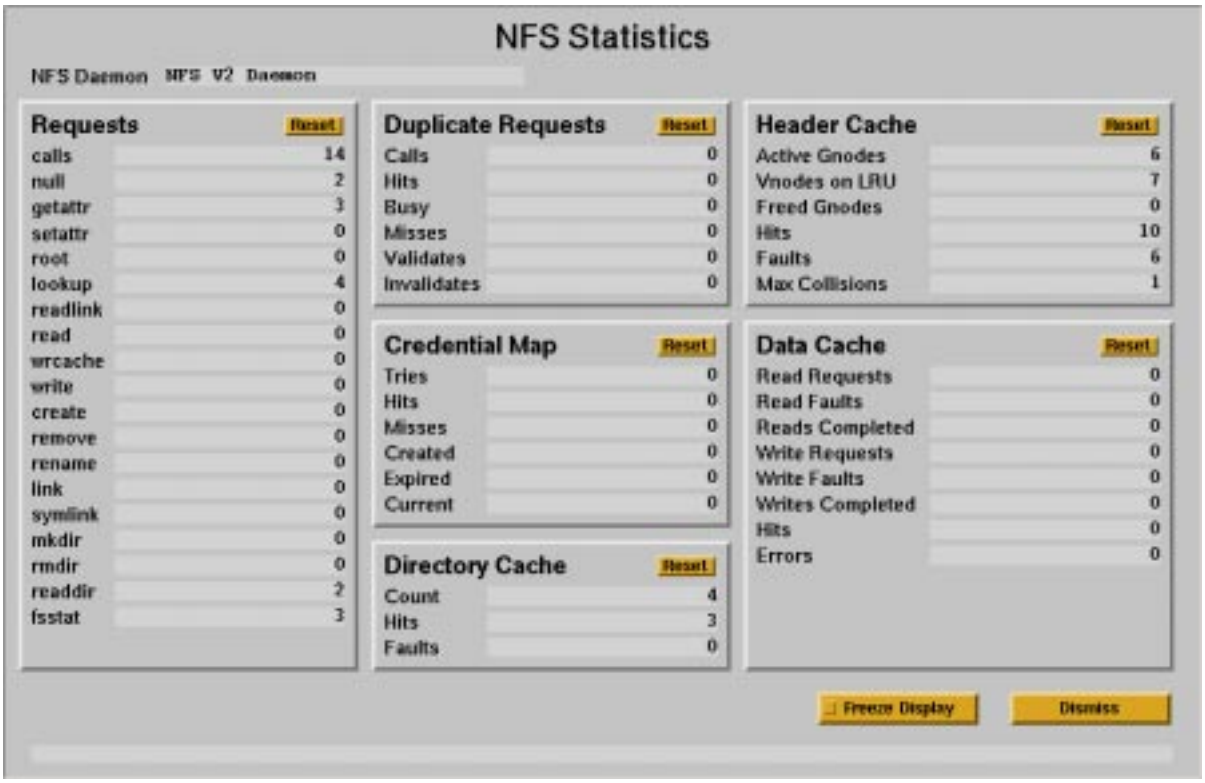


Figure 6-40 NFS Statistics Window

The NFS Statistics window identifies the server and displays statistics for that server's NFS request processing, credentials map cache, attribute cache, and data cache operations. NFS request processing statistics include counters for each type of NFS call and statistics on the processing of duplicate NFS requests. Request counters are contained in the block labeled **Requests**. Duplicate request processing statistics are contained in the block labeled **Duplicate Requests**. The credential map cache contains entries that can be used to map NFS client user credentials to HPSS user credentials. Credentials map cache operations and statistics are contained in the block labeled **Credential Map**. These fields will be non-zero if the UIDMAP option is being used. Attribute cache statistics are contained in the blocks labeled **Directory Cache** and **Header Cache**. Statistics for data cache operations are contained in the block labeled **Data Cache**.

Refer to the window's help file for more information on the individual fields as well as the supported operations available from the window.

6.7.9 Monitoring the DMAP Gateway Statistics

From the DMAP Gateway Fileset Information window (Figure 6-21), click on the **Show Statistics ...** button. The DMAP Gateway Fileset Statistics window will be displayed as shown in Figure 6-41.

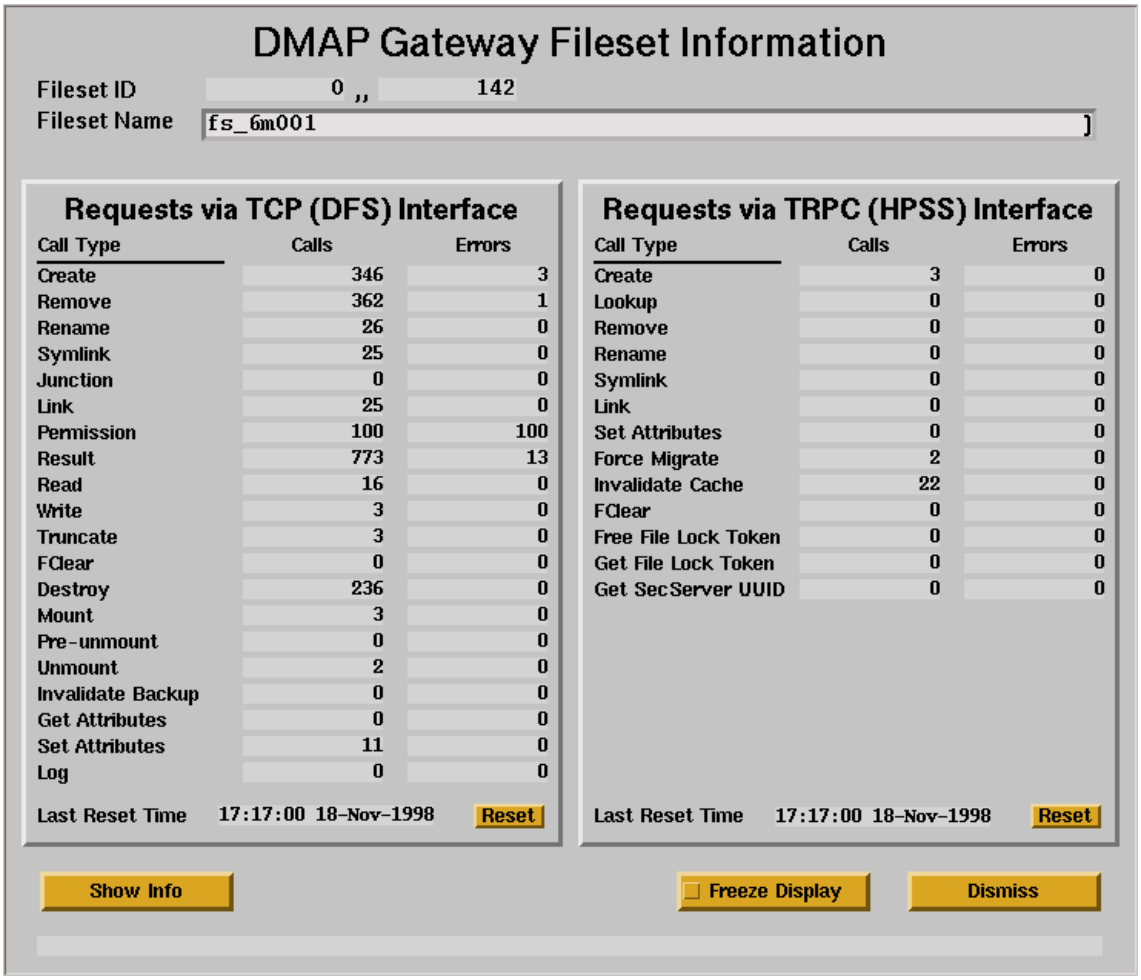


Figure 6-41 DMAP Gateway Fileset Statistics Window

6.7.10 Monitoring HPSS Log Files

From the Health and Status window (Figure 6-1), click on the **Monitor** menu, select the **HPSS Objects** and click on the **Logfile 1** or **Logfile 2** button. The Log File Information window will be displayed as shown in Figure 6-42.

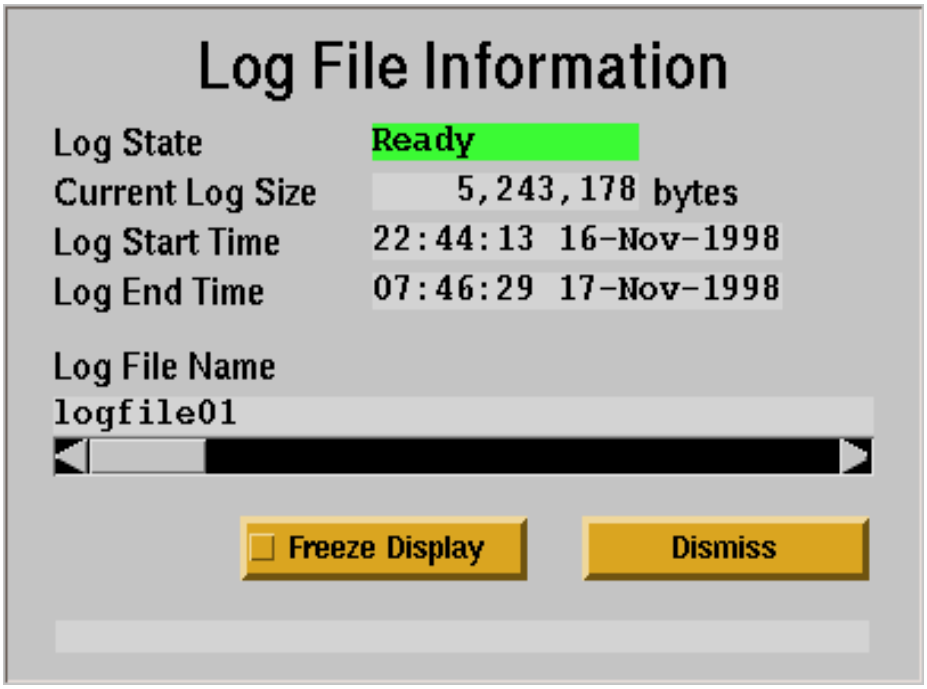


Figure 6-42 Log File Information Window

The Log File Information window provides information about an HPSS log file. Log file information includes the state of the log file, the current size of the log file in bytes, the time at which the log file was marked in use, the time at which the log file was last active, and the log file name. The **Log State** will be one of the following:

- IN USE—file is currently active
- READY—file is ready for use
- ARCHIVING—file is being archived
- ARCHIVE—file is marked for archival
- INVALID—file is not valid (should not occur)

Refer to the window's help file for more information on the individual fields as well as the supported operations available from the window.

6.7.11 Monitoring HPSS Bitfiles

From the Health and Status window (Figure 6-1), click on the **Monitor** menu, select the **HPSS Objects** and click on the **Bitfiles** option. A window will pop up to allow the user to enter the bitfile's full pathname. Press the **Enter** key after entering the request data then click on the **Get Info** button.

If the file exists, the Bitfile Information window will be displayed as shown in Figure 6-43. This window gives information about the file that is extracted both from the Bitfile Server and the Name Server.

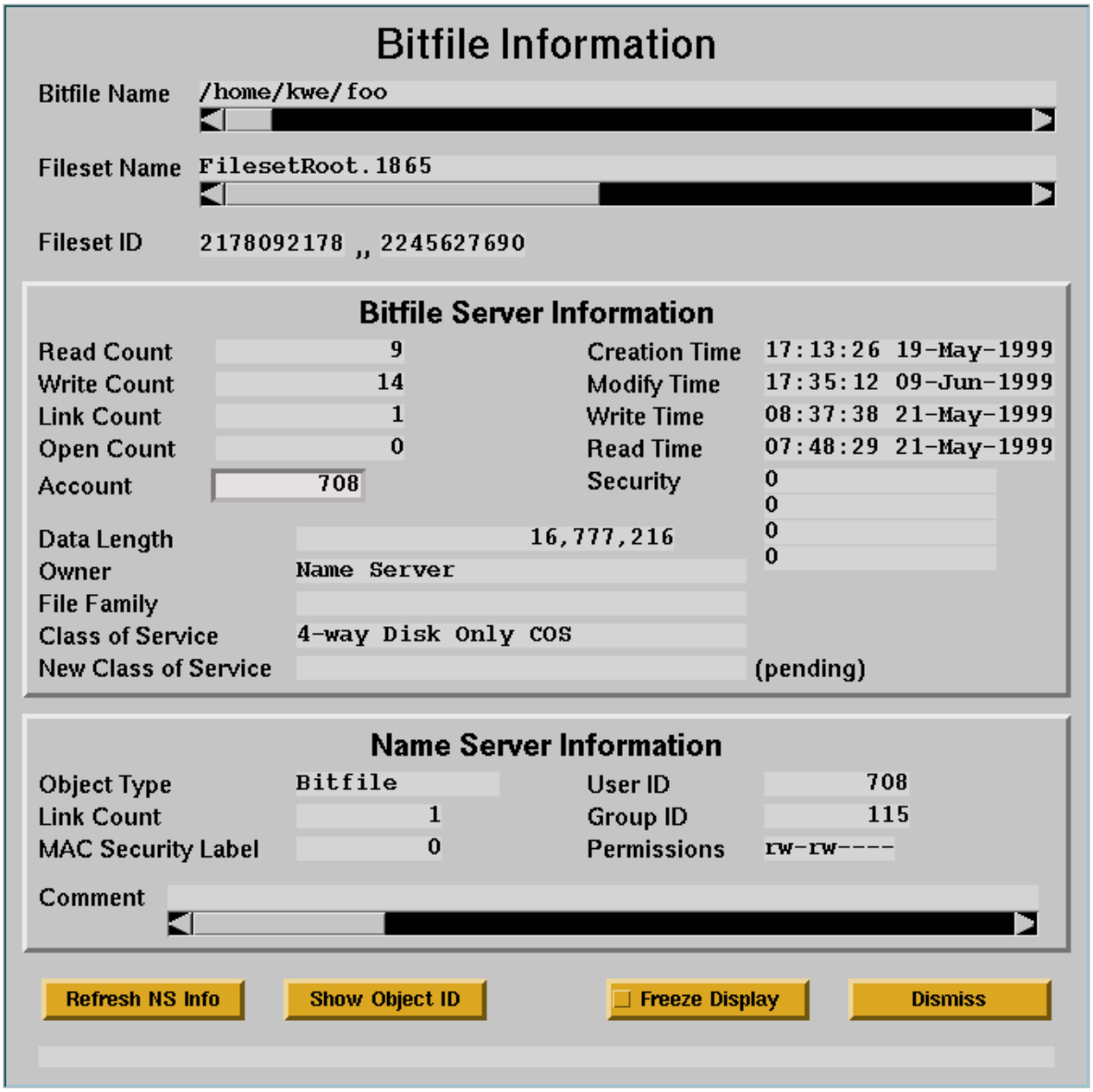


Figure 6-43 Bitfile Information Window

The Bitfile Server information includes access statistics, COS information, and file size. This information is updated in the window on a real-time basis as this information changes.

The Name Server information includes User ID, Group ID, and permissions. This information is not updated in real time as it changes. To see the latest information, click on the **Refresh NS Info** button.

Refer to the window's help file for more information on the individual fields as well as the supported operations available from the window.

6.7.12 Viewing HPSS Objects By SOID

A SOID (standard storage object identifier) is an internal HPSS identifier that uniquely identifies an HPSS resource. The capability to view HPSS objects by SOID allows the SSM user to determine the owner of a HPSS object by its SOID for problem diagnosis. The user must have extensive knowledge of the HPSS internal structures to be able to take advantage of the provided information. The types of SOID which can be viewed through this window are bitfile, virtual volume, and storage segment.

From the Health and Status window (Figure 6-1), click on the **Monitor** menu, select the **HPSS Objects** and click on the **Specify By SOID** option. The Identify HPSS Object By SOID window will be displayed (as shown in Figure 6-44). Enter data in the **Type**, **Server UUID**, and **Object UUID** fields and click on the **Get Info** button. The appropriate object window will be displayed for the user to view.

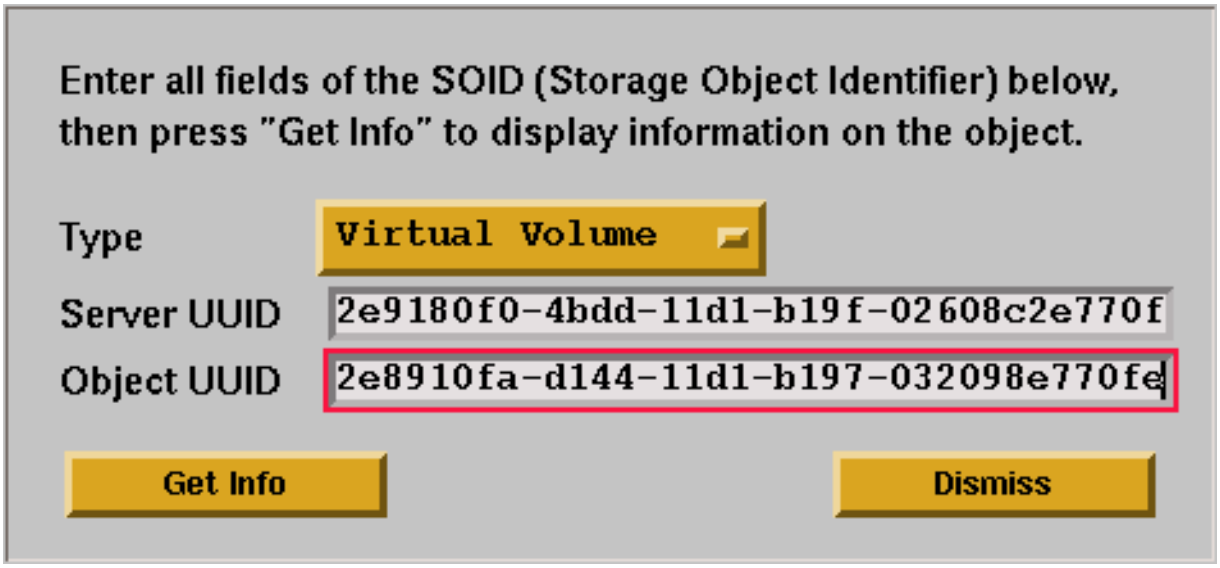


Figure 6-44 Identify HPSS Object By SOID Window

6.7.13 Monitor the PVL Tape Check-In Requests

All HPSS tape check-in requests will be displayed on the Tape Check-In Requests window. Outstanding tape check-in requests are displayed in chronological order. The operator will typically use this window to view which shelved cartridges require manual mounting.

Using the Tape Check-In Requests Window

From the HPSS Health and Status window (Figure 6-1), click on the **Monitor** menu and select the **Tape Check-In Requests** option. The Tape Check-In Requests window will be displayed as shown in Figure 6-45. Refer to the window's help file for more information on the individual fields as well as supported operations available from the window.

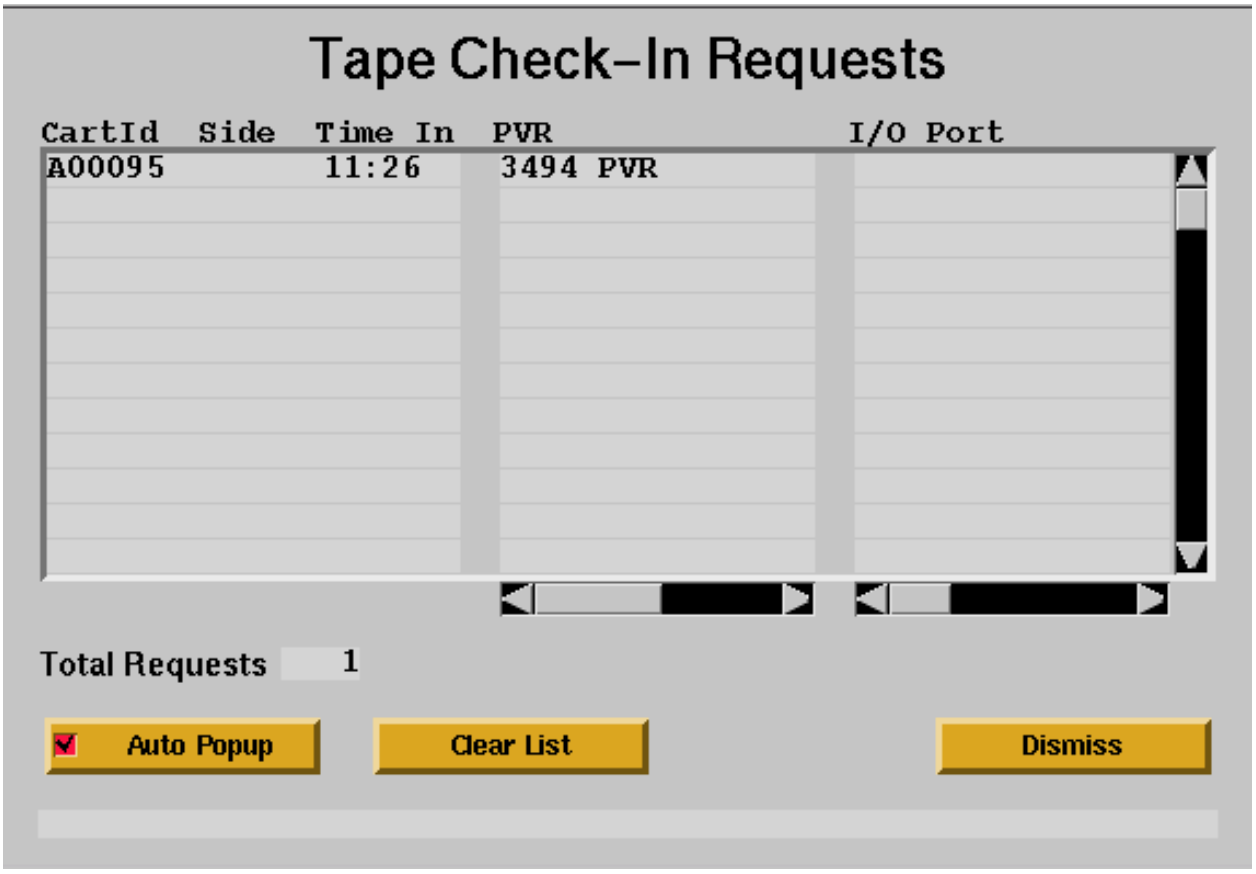


Figure 6-45 Tape Check-In Requests Window

The **Clear List** button on the Tape Check-In Requests window can be used to remove all tape check-in requests displayed, including any pending check-in requests. Pending check-in requests will reappear in the window as the PVR periodically retries the check-in of cartridges.

Tape check-in requests will no longer be displayed once they are satisfied. However, at times, the window may display an old tape check-in request due to lost notifications. Click on the Clear List button to refresh the window.

The maximum number of tape check-in messages that can be displayed on the Tape Check-In Requests window is 100. When this limit is exceeded, some tape check-in notifications will be lost. SSM will inform the user when this condition occurs by putting a message to the right of the Total Requests field.

When a new tape check-in notification is received and the Tape Check-In Requests window is not currently displayed, SSM will bring up the window to notify the user of the new tape check-in notification. If this is not desired, the Auto Popup button can be turned OFF, after which SSM will not automatically bring up the window. The Auto Popup button can be turned back ON by re-selecting the Tape Check-In Requests option from the Monitor menu of the Health and Status window.

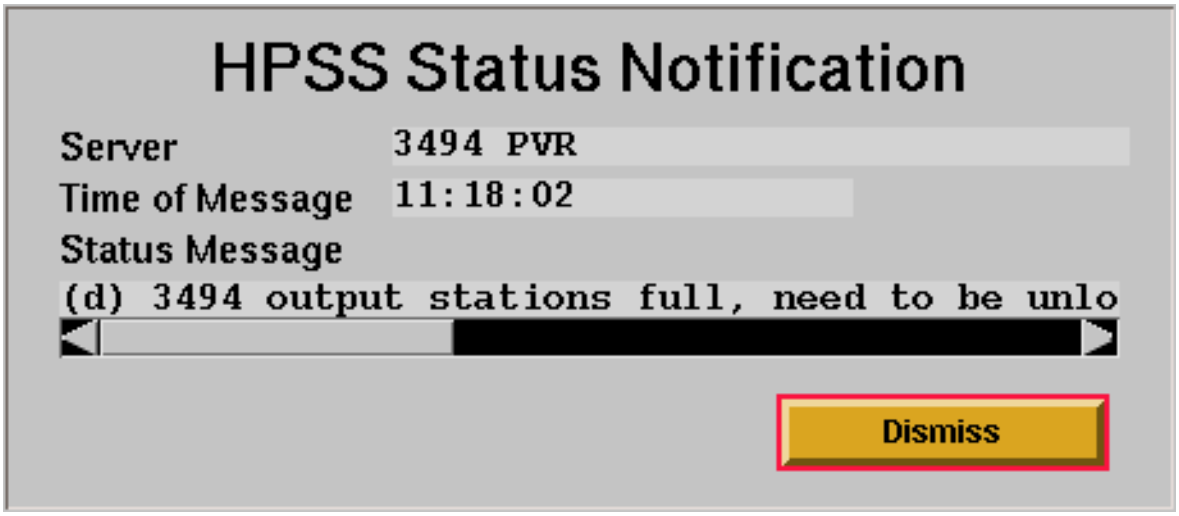


Figure 6-46 HPSS Status Notification window

Using the HPSS Status Notification Window

If a Tape Check-In Request has been displayed, and the corresponding tape has not been checked in, the PVR will begin logging alarms and creating HPSS Status Notification windows with messages indicating that the appropriate tape has yet to be checked-in. The frequency of these alarms and messages is controlled by the **Shelf Tape Check-In Alarm** field of the PVR-specific configuration screen.

After the appropriate tape(s) have been checked-in, the HPSS Status Notification windows may be dismissed.

6.8 Generating an Accounting Report

An accounting report can be started that will accumulate accounting information from metadata and generate a report file. Before beginning an accounting report, you must create and fully configure an Accounting Policy. The initial Accounting Policy should be set up before any files are created in the HPSS system. The Accounting Policy fields can be updated any time after the initial setup.

From the HPSS Health and Status window (Figure 6-1), click on the **Operations** menu and select the **Start Accounting** option. Messages which indicate the progress of the accounting run will appear in the HPSS Alarms and Events window. Accounting will display a completion status

message when it finishes. If an accounting run is interrupted, it can be restarted using the same mechanism.

6.9 Managing HPSS Logging Services

This section describes how to manage and control the HPSS Logging Services. The administrator needs to determine which type of log messages are to be logged, which types of log messages are to be sent to SSM, which type of logs are to be maintained, and whether the central log should be archived into HPSS. The section also describes how the logging information can be used to help the administrator in tracing and resolving problems encountered by the HPSS servers.

6.9.1 Controlling HPSS Logging Services

The ability to configure and control the HPSS logging services allows the site administrators to set up a logging environment appropriate for the site's requirements. The primary purpose is to record data to provide a trail of events that occur in HPSS. This data may include alarm, event, request, security, accounting, debug, status, or trace information. Logging services include central logging, local logging, and logging policies to determine what information is logged and what information is relayed to SSM for display.

Control of logging services is performed through the modification of HPSS configuration metadata for some or all of the logging components. A standard configuration for logging services is usually set by the administrator during the HPSS system configuration. Specialized configurations can be set up and used to temporarily (or permanently) provide greater or lesser logging capabilities for site-specific or shift-specific operational situations. Increasing the level of logging may slow down overall system efficiency due to the overhead of extra messages. Decreasing the amount of logging reduces overhead, but may eliminate certain log messages that could prove useful in a postmortem analysis of problem situations. For any new logging configuration to take effect, one or more components of the logging service may need to be reinitialized.

6.9.1.1 Controlling an HPSS Local Log File

The **Log Messages To:** field in the Logging Client configuration window (Figure 5-29) can be modified to control the destinations for the messages logged by the HPSS servers running in a node. This parameter consists of a set of options that apply to the local logging. These options are:

- **Log Daemon**: Send log messages to the central log.
- **Local Logfile**: Send log messages to the local log file.
- **Syslog**: Send log messages to the syslog.
- **stdout**: Send log messages to standard output.

It is an error to specify both the **Local Log File** and the **Standard Output** options. Any other combinations of these options are supported. If neither the **Local Log File** option nor the **Syslog** option is specified, no local logging will occur. If the **Log Daemon** option is not specified, no messages from the HPSS servers running on the same node as the Log Client will be written to the central log.

The administrator should choose only the local logging options that will satisfy the site's need, either temporarily or permanently. After the Logging Client configuration parameter has been changed, the associated Log Client must be reinitialized for it to reread the new configuration. For detailed information on Log Client configuration fields, refer to Section 5.6.10.

The syslog option should be used with care. The syslog file will grow without bound until deleted/ truncated, or until the file system runs out of space.

6.9.1.2 Controlling the HPSS Central Log Files

The HPSS central log files can be controlled by changing the **Archive Logfiles** and the **Switch Logfiles** flags on the Logging Daemon Configuration window (Figure 5-28). These flags define the operational behaviors of the central log files. The **Archive Logfiles** flag dictates whether the log files will be automatically archived when they are filled. The **Switch Logfiles** flag dictates whether a switch to the second log file should be done if the archive of the second log file has not yet completed. The administrator needs to consider whether archiving of the log files are needed and then, if so, set these flags appropriately. For detailed information on Log Daemon configuration fields, refer to Section 5.6.9.

6.9.1.3 Controlling HPSS Server Logging

A server's logging policy can be modified to control the volume of messages to the chosen logging destinations. Typically, during normal operations, the level of logging may be decreased to only **Alarm**, **Event**, and **Status** to reduce overhead. However, when tracking an HPSS problem, it may be desirable to include more log message types such as **Debug**, **Trace**, and **Request** to obtain more information for debugging purposes. The administrator should keep in mind that an excessive level of logging may increase the system overhead. However, too little logging may eliminate the log messages that could provide useful information for problem analysis.

To make it possible to diagnose data transfer problems, it is recommended that Debug logging is turned on for all Movers.

As a default, if no logging policy is configured for an HPSS server, all log messages except for **Trace** messages will be logged. We recommend that a logging policy be created for each configured server so that the level of logging for each server can be adjusted as needed.

After a server's logging policy has been modified, the Log Client running on the same node as the server must be reinitialized for it to reread the new logging policy. For detailed information on Logging Policy configuration fields, refer to Section 5.4.4.

If a Mover is running in non-DCE mode, the Log Client that runs on the same node as the Mover DCE/Encina process is used by **ALL** parts of that Mover. For a change in the Mover's logging policy to take complete effect, the Log Client **AND** the Mover must be reinitialized (the Mover reinitialization is required for the updated logging policy to be communicated to the non-DCE/Encina processes).

6.9.1.4 Controlling HPSS Server Notifications to SSM

HPSS alarms, events, and status messages are asynchronous notifications sent by the HPSS servers to SSM. A server's logging policy can be used to control the log message types to be sent to SSM as

asynchronous notifications. The log messages of the specified types will be forwarded to SSM by the logging services.

If the server's notifications to SSM are not desired, they can be turned off in the server's logging policy. The notifications will not be displayed on the SSM windows, but they will still be logged to the appropriate log files unless their log options are also turned off.

After a server's logging policy has been modified, the Log Client running on the same node as the server must be reinitialized for it to reread the new logging policy. For detailed information on Logging Policy configuration fields, refer to Section 5.4.4.

If a Mover is running in non-DCE mode, for a change in the Mover's logging policy to take complete effect, the Log Client **AND** the Mover must be reinitialized (the Mover reinitialization is required for the updated logging policy to be communicated to the non-DCE/Encina processes)

In addition to modifying the logging configuration to control logging, the HPSSLOG_SHMKEY, HPSSLOG, and HPSSLOGGER environment variables described in Section 4.3 can also be used.

6.9.2 Viewing the HPSS Log Messages and Notifications

This section describes how to use the HPSS log messages and notifications to monitor the HPSS health and status. It also describes how to use the log messages to obtain more information for problem analysis.

6.9.2.1 Viewing HPSS Alarms and Events

From the HPSS Health and Status window (Figure 6-1), click on the **Monitor** menu and then select the **Alarms/Events** option.

The HPSS Alarms and Events window will be displayed as shown in Figure 6-47. This window displays all the alarms and events received by SSM and uses the following color scheme to indicate the severity level of the alarms and events:

- Red background: Critical and Major alarms.
- Yellow background: Warning and Minor alarms.
- Gray background: Events and Information.

For each alarm or event, the following data is displayed:

- Acknowledgment Indicator.
- Timestamp.
- Severity level.
- Message number.
- Message text.

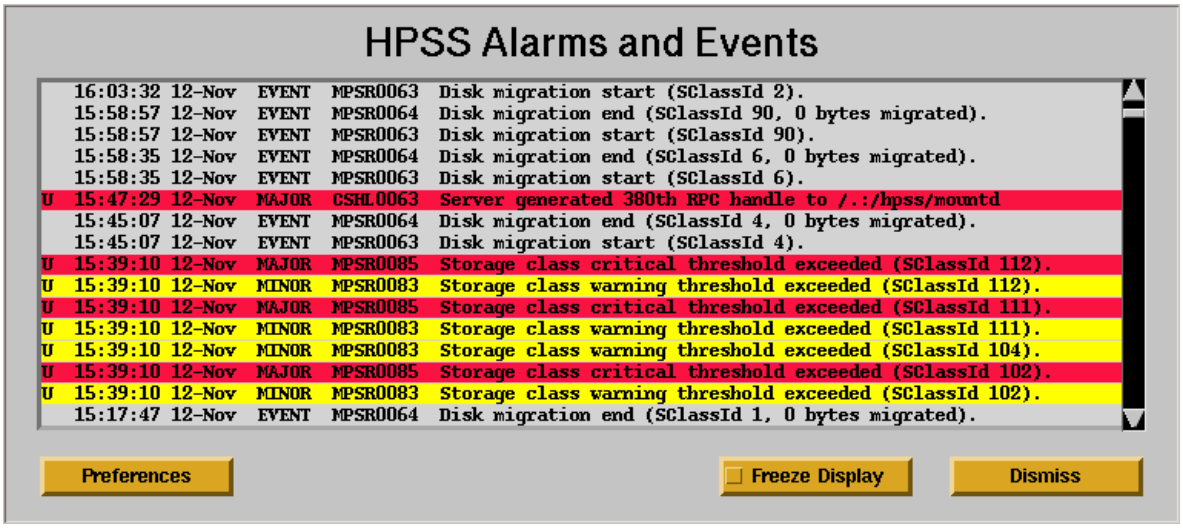


Figure 6-47 HPSS Alarms and Events Window

Events displayed on the HPSS Alarms and Events window provide an indication that a significant event has occurred in a server and the event may be of interest to the user. An alarm, however, indicates that a server has detected an abnormal condition. The user should investigate the problem as soon as possible to ensure timely resolution. The user may use the information provided by the alarm to obtain further information on the problem as follows:

- Click the left mouse button on a message. The Alarm / Event Information window will be displayed (Figure 6-48), showing complete information on the message.
- Use the alarm message number to look up the alarm information in the *HPSS Error Messages Reference Manual*, HPSS Release 4.1.1. For each documented message, the manual provides more detailed information on the problem and the possible source of problem. If available, the manual also provides recommendations on how to resolve the problem.
- If additional information is needed, use the alarm timestamp to delog the HPSS central log for the log messages received prior to and after the problem is reported. In addition to obtaining the messages logged by the server that reported the problem, it may be necessary to obtain log messages from other HPSS servers that interface with the server. Refer to Section 6.9.2.2 for more information on delogging the HPSS log messages.
- Alarms can be acknowledged by clicking on them with the right mouse button.

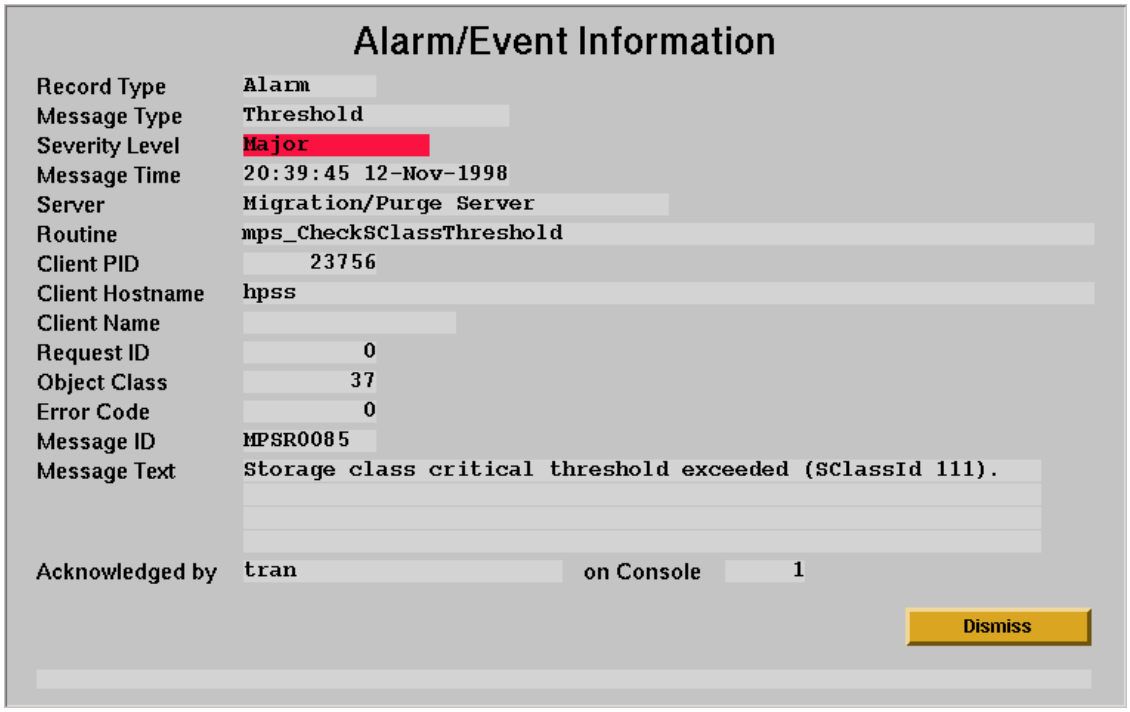


Figure 6-48 Alarm / Event Information Window

Filtering HPSS Alarms and Events

It is advisable to temporarily remove old alarms and events from the HPSS Alarms and Events window so that it is easier to keep track of the outstanding alarms and events. We recommend that the user acknowledge the resolved or returned-to-normal alarms. In addition, the alarm / event filter feature should be used to filter out the acknowledged alarms and any message types that are of no interest to the user.

From the HPSS Health and Status window (Figure 6-1), click on the **Session** menu, select the **Preferences** option, and then select **Alarms and Events**.

Or

From the HPSS Alarms and Events window (Figure 6-47), click on the **Preferences** button.

The HPSS Alarm and Event Preferences window will be displayed as shown in Figure 6-49. Messages can be filtered by severity level, alarm or event type, and acknowledgment status. They can also be filtered by subsystem name and by any text in the body of the messages. Refer to the HPSS Alarm and Event Preferences window help file for more information on the individual fields as well as the supported operations available from the window.

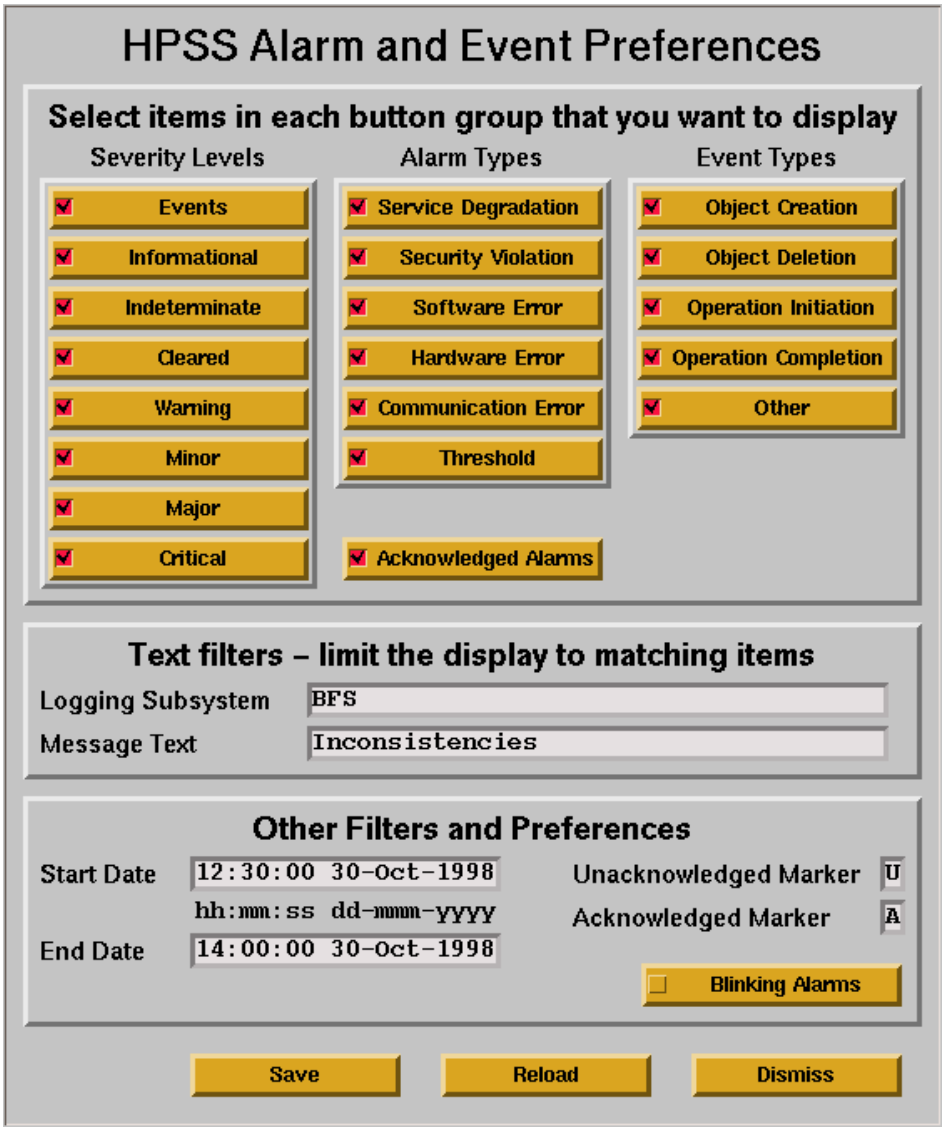


Figure 6-49 HPSS Alarm and Event Preferences Window

6.9.2.2 Viewing HPSS Log Messages

Viewing the Local Log

The local log file is an ASCII text file and can be viewed as such. The name of the local log file can be found in the Log Client configuration entry. Unlike the HPSS central log file, which contains messages from all HPSS servers, the local log file only contains messages from the HPSS servers running in the same node as the Log Client. Because this file restarts at the beginning when full, the oldest messages will be lost.

Viewing the Central Log

The ability to retrieve and examine HPSS log records provides a convenient tool for analyzing the activity and behavior of HPSS. The retrieval and log record conversion process is referred to as "delogging." The delog process retrieves specific records from the HPSS central log files, converts the records to a readable text format, and outputs the text to a local POSIX file. Delogged records can be viewed on the Delogged Messages window.

To effectively view log records, the user should know what kinds of activities or behaviors are of interest. Since it is possible to configure logging policies to include almost all activity in HPSS (see Section 6.9.1), in most cases, it will be easier to examine and analyze log records by filtering out any records that are not of interest. The Specifications for Delogging HPSS Messages window allows selective filtering of log records based on start time, end time, server descriptive name, log record type, and user name.

From the HPSS Health and Status window (Figure 6-1), click on the **Monitor** menu and select the **HPSS Message Log** option.

The Specifications for Delogging HPSS Messages window will be displayed as shown in Figure 6-50. Enter the desired delog specifications and click on the **Execute** button.

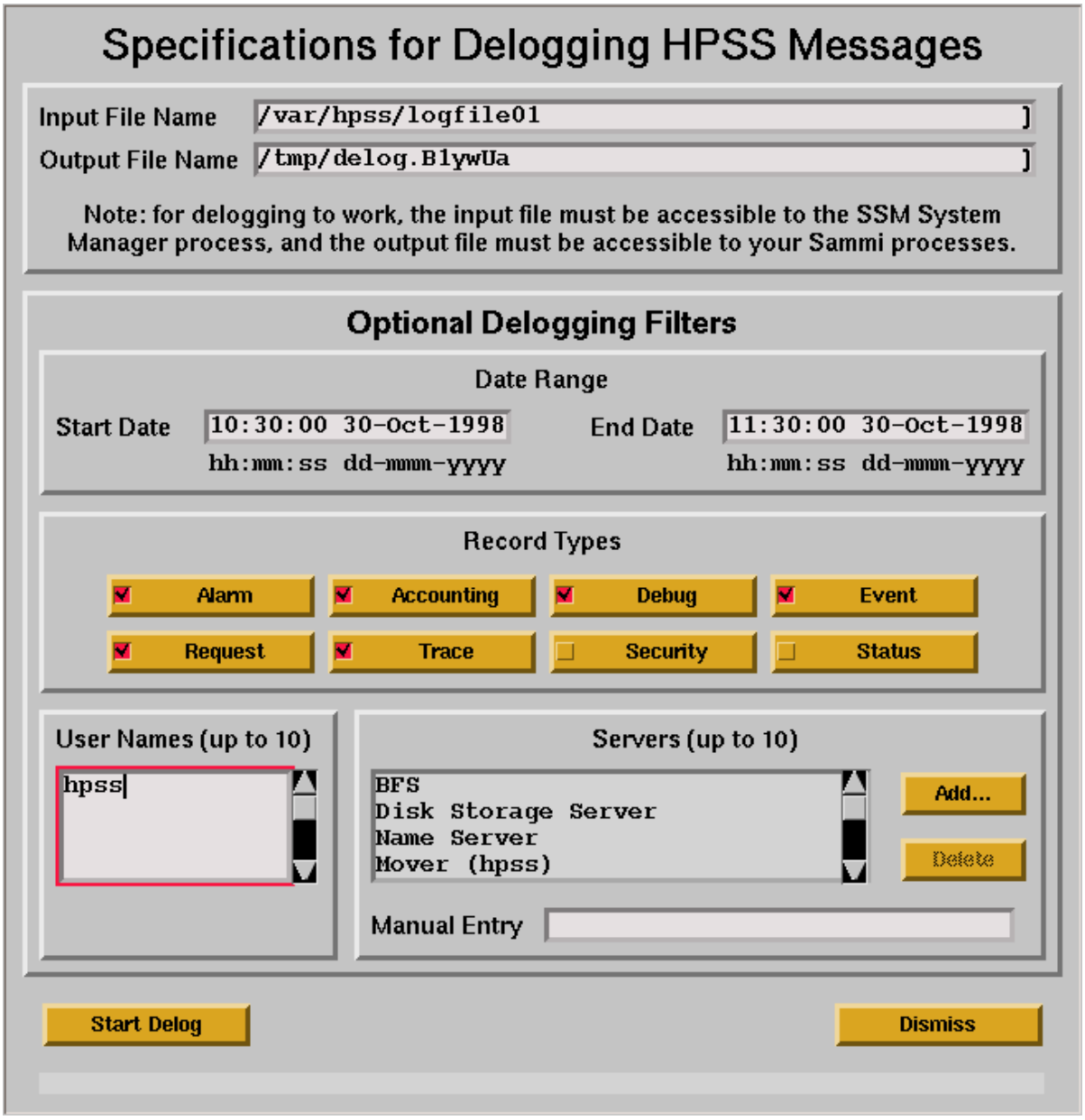


Figure 6-50 Specifications for Delogging HPSS Messages Window

The central log file names available for delogging are logfile01 and logfile02. These files reside in the directory name specified in the Log Daemon configuration entry. To determine whether a message of interest is in a log file, bring up the associated Log File Information window (Figure 6-42) to obtain the start and end times of the log file and compare them with the message timestamp.

If log files that have been archived to HPSS are to be delogged, they will reside in the **/log** directory. To determine the log times for an archived log file, list the log files. The timestamp of the time when the file was archived to HPSS is appended to the end of the log file name. The format of the archive file name is **logfile01_YYMMDDHHMM** or **logfile02_YYMMDDHHMM**.

Delog runs as a separate "on-demand" process for each SSM delog request. Each process runs to completion and exits when all selected records are retrieved, formatted, and written to the specified output file. When the delog is completed, SSM will display the Delogged Messages window as shown in Figure 6-51.

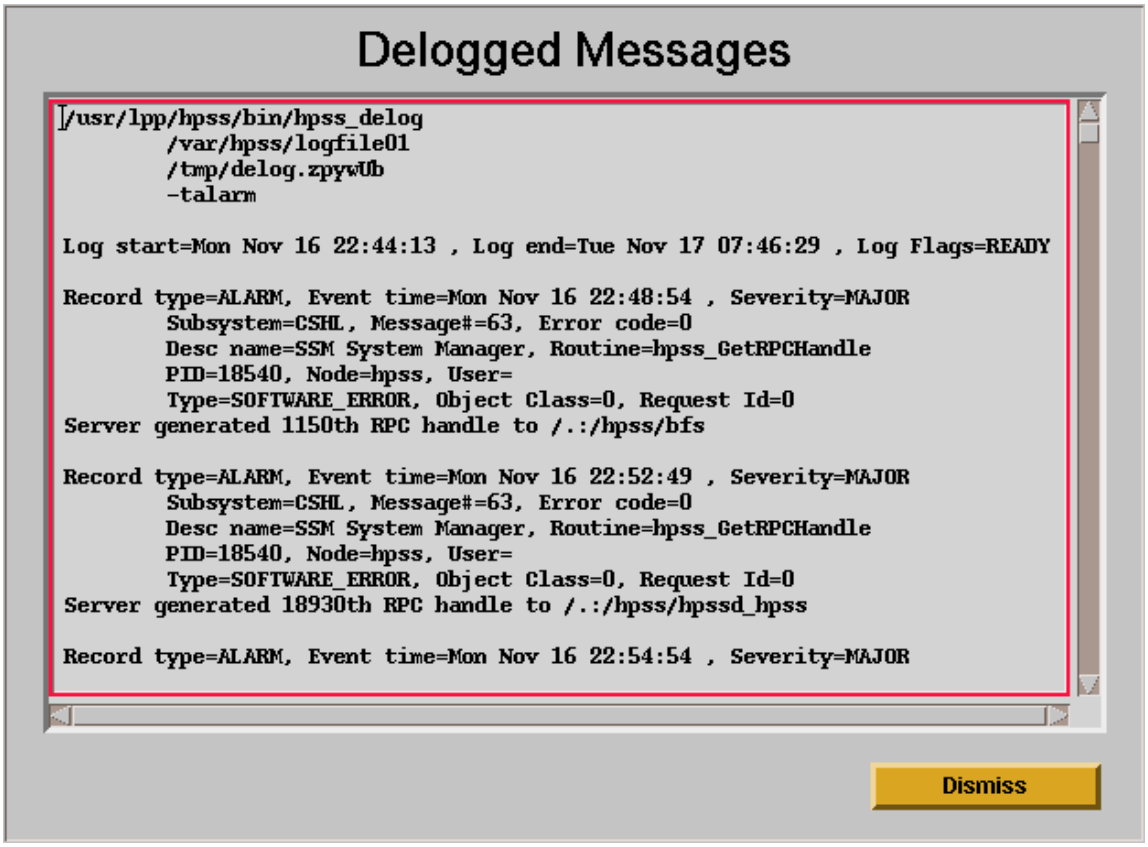


Figure 6-51 Delogged Messages Window

The Delogged Messages window allows the user to view the delogged messages. The user can also view these messages by opening the output file. Refer to the Specifications for Delogging HPSS Messages window help file and the Delogged Messages window help file for more information on the individual fields as well as the supported operations available from the windows.

For the SSM user to be able to view the delogged messages from the SSM window, either the Log Daemon and the SSM session must be running on the same node or the delogged messages file must be accessible to the Sammi processes (e.g., via NFS).

SSM will not be able to properly view the delog output file unless the SSM user has read/write permission on the file. Since the delog output file is created by the SSM System Manager, this means that the System Manager must be running with a umask setting which does not prohibit group read/write access, and that SSM users should be in the same Unix group as the System Manager.

Viewing large numbers of delogged messages in SSM will cause the Sammi process s2_event to consume large amounts of system memory. For this reason, the SSM user should always use the Specifications for Delogging HPSS Messages window to limit a delog request to the messages of interest, rather than delogging an entire log file.

Delete the delogged messages file when it is no longer needed.

Delog Command Line Input

A delog may also be requested from a command line input. The format of the **hpss_delog** command is as follows:

hpss_delog 'log file name' 'delog file name'
-s'start time' -e'end time' -d'server descriptive name' -t'record type' -p'log principal name'
-k'HPSS keytab pathname' -l'Location Server group name'

where:

log file name (required).
The UNIX path name of the primary or secondary log file (e.g., **/var/hpss/logfile01**, **/var/hpss/logfile02)**, or the pathname of an HPSS **archive log file** (e.g., **/log/logfile01_9504171230**)

delog file name (required).
UNIX path name of the Delog output file (required)

start time of first record to retrieve (optional)
- option = -s
- default = first record in log
- time is in YYYY:MM:DD:HH:MM:SS format (year is optional)
- at most, one -s option can be specified

-end time of last record to retrieve (optional)
- option = -e
- default = last record in log
- time is in YYYY:MM:DD:HH:MM:SS format (year is optional)
- at most, one -e option can be specified

server descriptive names (optional)
- option = -d
- default = all
- multiple -d options can be specified

record type (optional)
- option = -t

- default = all
- multiple types may be entered
- options = alarm, event, request, security, accounting, debug, status, and trace
- multiple -t options can be specified

log principal name (optional)
- option = -t
- default = hpss_log
- at most one -n option can be specified

HPSS keytab pathname (optional)
- option = -k
- default = /krb5/hpss.keytabs
- at most one -k option can be specified

Location Server rpc group name (optional)
- option = -l
- default = /.:/hpss/ls/group
- at most one -l option can be specified

Examples:

```
hpss_delog /var/hpss/logfile01 /mydir/delog.out

hpss_delog /var/hpss/logfile02 /mydir/delog.out
-s01:21:01:00:00 -e01:21:03:30:00

hpss_delog /var/hpss/logfile01 /mydir/delog.out
-s95:1:21:01:00:00 -e94:01:21:01:00:00
-talarm -tevent -d'Storage Server'

hpss_delog /var/hpss/logfile01 /var/hpss/delog.out
-tsecurity

hpss_delog /var/hpss/logfile02 /mydir/delog.out
-s01:24:12:59:59 -e01:24:16:00:00
-talarm -tdebug -d'STK PVR' -d'STK PVR'
```

6.10 Backing Up HPSS Metadata

Backing up the Encina SFS volumes containing HPSS metadata is essential for disaster and media-failure recovery. Without properly and systematically backing up this data, a disk failure, for example, on an Encina SFS volume could render the entire HPSS system useless and all or part of HPSS user data could potentially be lost.

Backing up HPSS metadata involves backing up the following:

- SFS server configuration and restart files
- SFS log volumes

- SFS data volumes

Note that the HPSS **backman** utility, as described in Appendix I, can be used to generate most of the backup operations described in this section. In addition, there is a set of utilities which provides a more automated way of handling backing up an SFS. These are described later in this chapter.

It is very important that DCE and Encina data are backed up properly to avoid data loss. The backman utility can be used to perform these backup operations. In addition to backing up DCE and Encina data, it is advised that full system backups be done periodically while HPSS, Encina, and DCE are not running. Refer to Section 2.11 for more information on the Encina backup guidelines.

The Encina documentation referenced below should be consulted in order to obtain a complete understanding and explanation of issues and procedures associated with enabling full recoverability of Encina data.

- *Encina Overview, IBM ITSO Redbook,* GG24-2512
- *Encina Structured File Server Administrator's Guide and Reference for AIX*
- *Encina Server Administration: System Administrator's Guide and Reference for AIX*

As a safety net mechanism in case the backup procedures mentioned in this section are not properly followed or in case of, for example, a tape being damaged, it is recommended that a complete data dump (e.g., using the dd command) be made of all Encina volumes on a regular interval. HPSS and Encina SFS must both be brought down before these raw backups can be made.

6.10.1 Backing Up SFS Configuration and Restart Files

After initially configuring the SFS server, and after any change to the server's start up options (changing the SFS buffer size, thread-pool size, etc.), back up the following files for each node running an Encina SFS server:

/var/encina/EncinaServers
/var/encina/EncinaServers.vc

When SFS is first configured and started, it generates a file containing important restart information. A current, valid copy of this file must be available for SFS to start, whether in the administrative or normal operations mode. Not only must it be backed up after initial configuration but it must also be backed up any time changes are made in the configuration of Encina volumes. The restart data is mirrored by being stored in two separate file system files. Assuming your SFS server name is **/.:/encina/sfs/hpss**, the files are:

/opt/encinalocal/encina/sfs/hpss/restart
/opt/encinamirror/encina/sfs/hpss/restart.bak

Ideally, the mirrored file (i.e., **restart.bak***) should be written to a separate disk. The directory* **/opt/encinamirror** *should be separately mounted to another disk or linked to another file system residing on a separate disk.*

6.10.2 Backing Up SFS Log Volumes

Encina SFS logs information as transactions are initiated and resolved. The Encina log volume, which is a disk volume created during HPSS initial configuration, is used to store this transaction log information. Media archive files (which are running backups of the SFS transaction log)

generated by SFS can be used with other backup files to restore SFS data volumes. Archive files can also be used to restore a log volume that is damaged or destroyed by media failure. Archive files, for this reason, should be periodically moved offline to secure storage and deleted from disk to make room for new files.

To accomplish this, media archiving must be enabled for each SFS server. To check whether media archiving is enabled, ensure you are DCE logged in as **encina_admin** and then issue the following:

% tkadmin query mediaarchiving -server <serverName>

If the output of this command indicates media archiving has not been enabled, it must be enabled using the following:

% tkadmin enable mediaarchiving -server <serverName>

Assuming your SFS server name is **/.:/encina/sfs/hpss**, these media recovery archive (or MRA) files are stored, by default, in:

/opt/encinalocal/encina/sfs/hpss/archives

Assuming a SFS log volume name of **logVol1**, the files are named **logVol1.LA.1, logVol1.LA.2**, etc.

The archive file with the highest sequence number (e.g., **logVol1.LA.9**) is the active, most recently created archive file. The server may still be writing data into that archive file. For that reason, *the active archive file should* **never** *be moved offline*. Completed archive files can safely be moved offline for storage using an operating system utility such as **tar**. The HPSS **`backman`** utility can be used for this task.

The frequency at which administrators need to move archive files offline depends on the amount of log data generated from HPSS activity. Minimizing the number of archive files that remain online for any period of time also minimizes the potential of lost data in case of disaster. Generally, these files should be moved to tape daily and then erased from disk in order to make room for new backup files. Multiple copies of these MRA files, as well as the TRB data volume backup files, should be written to separate tapes.

As SFS data volumes are backed up, older log volume backups become obsolete. The command **tkadmin query backup** can be used to determine the oldest MRA files still required for full recoverability. For example, if the command returned

Earliest MRA file required for this backup: logVol1.LA.39

this indicates that MRA files 1 through 38 are obsolete and are no longer needed. However, it is advisable to maintain a copy of the MRA files associated with the last five complete SFS data volume backups.

6.10.3 Backing Up SFS Data Volumes

All SFS data volumes should be backed up using the **tkadmin backup lvol** command. This command can be used to generate backup files and can be issued while the SFS and HPSS servers are running. The HPSS **`backman`** utility can be used as a front-end to this **`tkadmin`** command.

After SFS and HPSS are initially configured, daily incremental backups of all SFS data volumes should be made. These incremental backup files, together with the appropriate log archive backup files, will allow you to recover your SFS server(s) when necessary. SFS supports up to a maximum limit of 512 backup files.

Suppose you had two 2 GB data volumes and you wanted to generate backup files that were 32 MB each. Since it would take 64 of these 32 MB backup files to complete a 2 GB backup cycle, you could generate 8 backup files per day, giving you a complete backup every 8 days.

Each backup file has 1 KB of overhead. This overhead must be added to the desired size for the backup files. For instance, in the above example, the size value you must specify to the **backman** utility (or the **sfsbackup** or **tkadmin** utilities) is 32769 KB (which is 32 MB + 1 KB).

Assuming that the 2 GB volumes were named **sfsVol1** and **sfsVol2, backman**, by default, will place the backup files in **/archive/sfsVol1** and **/archive/sfsVol2** respectively. Files for **sfsVol1** begin with the name **sfsVol1.TRB.000000** and are incremented numerically.

After backup files are generated, they should be moved offline to tape and removed from disk.

In order to free up space used to keep information about old backup files, the **tkadmin retain backups** command can be used to tell SFS to keep information about a specified number of backups, starting with the most recent backup. For example, to truncate all but the last five complete backup versions of **sfsVol1**, the following can be entered:

tkadmin retain backups sfsVol1 5

Upon successful completion, this command displays a list of backup files about which information has been deleted as a result of this command. Note that the **backman** utility can also be used to perform this function.

The administrator should resist the temptation to decrease the amount of on-line storage space needed for backups by decreasing the size of backup files and increasing their number. This decision leads to long backup cycle times and to potentially a very time consuming data volume recovery, should one be needed. Not only will SFS demand all of the data volume backup files needed to complete a backup cycle, but it will also demand all of the associated log archive (MRA) files, which may be a very large number of files. The larger the number of files, the longer the recover will take and the greater the chance that one or more of the files will be unreadable, causing the recovery to fail.

6.11 Automated SFS Backup Utilities (an alternative to backman)

A recent addition (as of version 4.1.1.2) to the suite of HPSS utilities are the automated SFS backup utilities. These tools are designed to handle the generation and management of storage for the files that back up an SFS. The utilities handle copying the backup data to other media (such as tapes) and removing the backup files from the filesystems where they are generated when they are no longer needed or are safely backed up.

6.11.1 Architecture

There are two programs that perform the bulk of the work performed by the backup utilities: the backup run script and the backup monitor. Each SFS server serviced by the backup utilities uses both of these processes. The run script is scheduled to run periodically through the UNIX **cron** program. The run script generates several TRB files for each of its SFS server's data volumes and backs up these TRB files and any LA files that are not already backed up. The run script can be configured to make several copies of the backup files. Each of these copies may reside on a pool of tapes or in a directory in the filesystem where they can be backed up by some other means. In the case of a copy made to tape, the run script automatically manages the usage of the tapes: mounting tapes, writing data to them, dismounting them, and overwriting tapes that no longer contain any usable data.

The monitor can be run in one of two ways: either as a "daemon" of sorts, running continuously in the background, or scheduled to run periodically via a **cron** job. This program watches space usage in the filesystems where the LA and TRB files are generated. When the space usage reaches a configurable threshold, the monitor deletes backup files which are no longer usable for a restoration or which are usable but have been successfully backed up.

6.11.2 Prerequisites

In order to run the Backup Utilities, a Perl interpreter (version 5.003 or later) is needed. Perl is freely available from **http://www.perl.com**. It is also strongly recommended that the utilities be used with the TXSeries version of the Encina SFS. While there is no apparent reason why the utilities should not work with earlier versions, all testing has been done with this version of Encina.

6.11.3 Installation

In general, the utilities are intended to run on the same machine as the SFS server they are backing up. A single installation of the utilities could be used to run the backups for several SFS servers, even on different hosts, as long as the installation and configuration information are available to all hosts through some means (such as NFS; though see the Caveats section). The installation steps are outlined below:

Build Backup Utility Program Files

The utilities can be built at the same time HPSS is built. Locate the section in Makefile.macros that pertains to the utilities and edit those variables as described below before building HPSS. If HPSS has already been built, the utilties can still be built. Edit the Makefile.macros as described below, then change working directory to <*hpss root*>**/tools/sfs/sfs_backup_util** and issue a "make". Note that the utilities program files will not be installed under <*hpss root*>**/bin** but under <*hpss root*>**/tools/sfs/sfs_backup_util/bin**.

Edit the following variables in Makefile.macros:

AML_CTL

Set this variable to have the value "on" if support is needed for controlling AML (Grau) tape libraries, otherwise set it to "off".

AML_ROOT

If AML_CTL is set to "on", set this variable to the path of the directory containing the `inc` and `lib` directories for the ACI software (e.g. /usr/local/aci).

STK_CTL

Set this variable to have the value "on" if support is needed for controlling STK tape libraries, otherwise set it to "off".

STK_ROOT

If `STK_CTL` is set to "on", set this variable to the path of the directory containing the STK software `src` and `lib` directories.

Building a "stand-alone" distribution

It is also possible to build a "stand-alone" version of the utilities separate from the HPSS source tree. To do this, do the following:

$ cd <*hpss root*>**/tools/sfs/sfs_backup_util**

$ make SFS_BU_BUILD_ROOT=<*target dir*> **build-sfs-backup**

After this is done, edit the file Makefile.options under the destination directory. This file contains the same variables as the section in HPSS's Makefile.macros, in addition to the following:

HPSS_DRIVE_CTL

Set this variable to have the value "on" if some of the drives to be used to perform backups are controlled by HPSS, otherwise set it to "off".

HPSSMSG

Set this variable to have the value "on" if it is desired to log notifications of completions of backup runs and purge sweeps to the HPSS logging subsystem, otherwise set it to "off".

LOCAL_ROOT

If either `HPSS_DRIVE_CTL` or `HPSSMSG` is set to "on", set this variable to the path of the installation directory of HPSS (often `/usr/lpp/hpss` on AIX systems).

After the required options have been selected in `Makefile.options`, issue a "make" to build the utilities.

If it is necessary to rebuild the Backup Utilities at some later time, perhaps to configure a component that wasn't originally needed, the following sequence of commands will do so:

$ make clean

$ make

Construct Backup Utility Configuration

Choose a directory that will hold the configuration information for the Backup Utilities. This is the *configuration directory*. This directory will also hold the data files which log the state of the backup process for each SFS server. A specific structure of configuration files and directories must be created under the configuration directory. The Configuration section details how to do this.

As an aid to building the configuration, a utility named **build_cfg** is supplied that will create the configuration directory structure and fill it in with template versions of various configuration files that need to be edited in order to configure the utilities. See the Invocation section for a usage summary for this utility.

When creating the configuration directory for a server, do not simply copy over the configuration directory for an existing server. The Utilities store information in various files under the configuration directory that also record the state of the backups for that SFS server. Copying these files from one server to another is inadvisable since it could cause the backup scripts to fail, or to run, seemingly with no obvious errors, but without backing up needed data. When building a new configuration, always use the build_cfg tool to create the directory structure, then, if you want to use an old configuration as a starting point, copy over and edit only any of the following files (these are explained fully in the Configuration section):

- master.cfg
- drive.cfg
- library.cfg
- tape.cfg
- copy.cfg
- server.cfg

After editing the configuration, it may be desired to check the syntax of the configuration files before actually trying to run the Utilities. A program called **check_cfg** is provided for this purpose. This program will parse a Backup Utility configuration and report if there are any syntax errors in it. (Note that there are other elements of the configuration that could still be erroneous at run time, so it is necessary to check the log file when starting a new instance of a utility to ensure that it was able to configure correctly.) See the Invocation section for a usage summary of this utility.

Select Filesystems for Dumping Archive Files

When the Run Script performs backup activity for an SFS server, it must generate the TRB files on some filesystem accessible from the local host. The Run Script dumps these files into the directory specified in the server's `TRB_DIR` configuration parameter. The filesystem in which this directory resides must have enough space to accommodate all the TRB files generated on one run of the Run Script. See the Configuration section for details on what parameters affect how much backup space is generated on each run.

Furthermore, it is strongly recommended that the filesystems used for the LA and TRB files not be used for anything else. Since the Monitor tries to keep these filesystems within configured thresholds by purging files, if there is significant other activity affecting the space usage in these

filesystems, this could manifest itself as unexpected behavior on the part of the Monitor. For example, the Monitor could purge unexpectedly early if some other activity drives up the space usage in one of these filesystems.

Set Environment / Construct Site-Specific Files

The Backup Utilities require that several environment variables be set in order to run correctly. These environment variables are:

SFS_BACKUP_PERL

This must be set to the absolute pathname of the Perl interpreter that will be used to execute the Backup Utility scripts.

SFS_BACKUP_HOME

This must be set to the absolute pathname of the installation directory of the Backup Utilities. For an installation that is built within the HPSS source tree, this must be <*hpss root*>/tool/sfs/sfs_backup_util.

SFS_BACKUP_CFG_ROOT

This must be set to the absolute pathname of the configuration directory of the Backup Utilities.

SFS_BACKUP_STANDALONE

If the Utilities are being run from a stand-alone installation (as described above) this environment variable must be set to a nonempty value. If this variable is not set, the HPSS_ROOT environment variable must be set to the HPSS installation directory.

A common way to ensure that the environment is properly set up before invoking the Backup Utilities is to construct a script to wrap the invocation of the Utilities that sets these environment variables and starts the Utility desired (run script or monitor).

Prepare Backup Tapes

If any of the copies of the SFS server's backup data are to be made to tape, the tapes to be used must be prepared first. Provided with the Backup Utilities are two programs, **internal_label_write** and **internal_label_read**, for writing and reading the "internal labels" that the Utilities use to identify tapes and ensure that the correct tape is being written upon.

For each tape that is specified in the Tape Configuration File (see the Configuration section), the tape must be internally labeled. To do this, mount the tape in a drive and use the following syntax:

$ internal_label_write <device> -v <vol_id> -o <owner_id>

<device>

This is the device pathname of the drive, such as /dev/rmt0.

-v <vol_id>

This is the volume ID of the tape, which is typically the same as the external label of the tape, though any string can be used, as long as it matches what is specified for the INTERNAL_LABEL field of the tape in the Tape Configuration File (explained in the Configuration section).

-o <owner_id>

This specifies the owner ID of the tape. This field is not currently used by the Utilities, but it can be used to identify the intended use of the tape (e.g. "SFS Backup") for someone to read with the **internal_label_read** utility. The syntax for this utility is:

$ internal_label_read <device>

Schedule Backup Run Script

The next step is to add an entry to the crontab file for the UNIX user ID under which the Backup Utilities will run (typically root). If the user ID under which the utilities will run is not root, it must have permission to read, compress, and delete the SFS server's LA and TRB files.

Enable SFS Server Media Archiving

Before any Backup Utility processes can run successfully, the media archiving option for the SFS server to be backed up must be enabled. This is accomplished by issuing the following command:

$ tkadmin enable mediaarchiving –s <CDS name of SFS server>

Start (or Schedule) Backup Monitor

After the Utilities are correctly configured and the media archiving option is enabled, the only thing to do is to enable the monitor program. The Monitor can be started and left to run in the background, but some provision must be made to restart it in case it is accidentally killed. A better solution is to create a cron job to run the Monitor in "one-pass" mode every so ofter. To use this method, ensure that the Monitor is invoked with the **-o** option to cause it to terminate instead of running continuously.

6.11.4 Invocation

Here are summaries of the invocation options for the various components of the Backup Utilities. In addition to the options listed, each program will print its usage message and exit if the **-?** option is specified, or print the version number of the Utilities and exit if the **-V** option is specified. Note that all the programs mentioned here, except for **clean_tempdir** (which is simply a shell script), are actually shell scripts that wrap the invocation of Perl scripts.

*Run Script (***sfs_backup_run***)*

sfs_backup_run [–d debug_file] –s server [–n] [-r] [-m addr,addr,…]

-d debug_file

This option is used to specify the name of a file that will accept debugging output from the script. If specified, this option should appear first.

-s server

This option is used to specify which SFS server the Run Script is being started for. This parameter specifies the configuration name used for the server, not the CDS name of the server. The configuration name of a server is the name given to the server's subdirectory under the "servers" directory in the configuration directory (see the Configuration section.) This parameter is required.

-n

When specified, this option causes the script to run in "no new TRBs" mode. This means that the script will back up any LA or TRB files that are not already backed up, but it will not generate any new TRB files. This is useful for performing backups of just the LA files several times a day, if desired.

-r

When specified, this option causes the script to retain any tape drives that it uses in the backup process. Normally, when the script uses a drive, it removes any tape which it mounted in the drive, and in the case of an HPSS-controlled drive, unlocks the drive so that HPSS can use it again. When this option is specified, when the script terminates, it will leave any backup tape in the drive, and it will leave an HPSS-controlled drive locked. This is useful for situations where the script must be run several times in a row, such as when a site does not have enough disk space to generate much data volume backup at a time. The script must be run several times in a row, generating a portion of the data volume backup on each run. This option can be specified for every invocation of the script but the last in the sequence, saving the time of mounting and dismounting tapes several times.

-m addr,addr,…

This is a comma-separated list of email addresses that will be mailed any error messages resulting from reading the configuration. Since the normal addresses to be mailed are located within the configuration, if the configuration cannot be read, those addresses will not be mailed with the error message. This parameter provides a fallback so that someone can always be notified in the event of a failure.

*Monitor (*sfs_backup_monitor*)*

sfs_backup_monitor [-d debug_file] –s server [-o] [-m addr,addr,…]

-d debug_file

This option is used to specify the name of a file that will accept debugging output from the script. If specified, this option should appear first.

-s server

This option is used to specify which SFS server the Run Script is being started for. This parameter specifies the configuration name used for the server, not the CDS name of the server. The configuration name of a server is the name given to the server's subdirectory under the `servers` directory in the configuration directory (see the Configuration section.) This parameter is required.

-o

When specified, this option causes the script to perform one "purge sweep" and exit. Normally, the Monitor, once started, will run continuously, monitoring the space usage in filesystems and purging backup files when necessary. Under this option, the script will just check the space usage once, purge backup files if necessary, then exit.

-m addr,addr,…

This is a comma-separated list of email addresses that will be mailed any error messages resulting from reading the configuration. Since the normal addresses to be mailed are located within the configuration, if the configuration cannot be read, those addresses will not be mailed with the error message. This parameter provides a fallback so that someone can always be notified in the event of a failure.

*Configuration Builder (*build_cfg*)*

build_cfg [-d <cfg_root_dir>] [-f] <servers>…

-d <cfg_root_dir>

This is the pathname of the directory under which the configuration should be built. If this parameter is not specified, the program takes the value from the environment variable SFS_BACKUP_CFG_ROOT.

-f

If this flag is specified, the program will not prompt before removing existing files and directories that are in the way of where the configuration files and directories should be.

<servers>…

These are the configuration names of each SFS server to create a configuration for under the main configuration's `servers` directory. Note that all the servers need not be specified in one invocation of the program. After a configuration is initially built, the program may be used to create additional server configurations.

Configuration Checker (check_cfg)

check_cfg [-d <cfg_root_dir>] –s <server>

-d <cfg_root_dir>

This is the pathname of the directory under where the configuration to be checked resides. If this parameter is not specified, the program takes the value from the environment variable SFS_BACKUP_CFG_ROOT.

-s <server>

This is the configuration name of the server whose configuration the program is to check.

*Temporary Directory Cleaner (*clean_tempdir*)*

This utility is provided to clean out any temporary files that have been left behind by the Backup Utilities. It can be run periodically on the directory specified in the Master Configuration's TMP_DIR parameter (see the Configuration section for details). It is safe to run this utility while instances of the Backup Utilities are running, since it uses the process ID values embedded in the temporary file names in order to only delete files associated with processes which are no longer running.

clean_tempdir <temp_dir>

<temp_dir>

The pathname of the directory to be cleaned.

6.11.5 Configuration

This section describes how to prepare the configuration files and directory structure under the Backup Utility configuration directory. It describes both configuration actions that need to be performed only once for all SFS servers serviced by one installation of the Backup Utilities, as well as actions that need to be performed for each SFS server to be backed up.

Notes on Configuration File Formats

All of the configuration files described below share some characteristics. They are all ASCII files consisting of a series of "assignment" statements that set up configuration parameters. They all obey the following syntactic properties:

- Blank lines (empty or consisting of only whitespace) are ignored.
- Lines whose first non-whitespace character is "#" are considered comments and ignored.
- Parameters whose value is a list of items should be specified by separating the items by whitespace.
- In general, a newline is considered the statement terminator. If a statement is to be extended over several lines in the interest of readability, a backslash "\" character can be used as the last character of a line to indicate that the following line is a continuation of the current one. For example, in specifying a parameter which is a list of files:

```
# List of administrator-defined files to back up
ADMIN_DEFINED_FILES = \
      /var/util/tool1 \
      /etc/tool1rc \
      /home/joe/.tool1rc
```

- Several of the configuration files are organized into "stanzas", with each stanza specifying properties of one instance of a type of object. Each stanza begins with a line consisting of the type of object followed by the instance name of the object being defined. After that, there is a sequence of assignment statements specifying the properties of the instance. Then, to end the definition of the instance, there is a line consisting of END_<object type>. For example, to specify two instances of drive objects named drive1 and drive2 in a drive configuration file, the following might be used:

```
DRIVE drive1
      DEVPATH=/dev/rmt0
      LIBRARY=3494
      ROBOT_ADDRESS=349627
END_DRIVE

DRIVE drive2:
      DEVPATH=/dev/rmt1
      LIBRARY=3494
      ROBOT_ADDRESS=349628
END_DRIVE
```

Note that the whitespace before each assignment line and the blank line between stanzas are just provided for readability.

- Any parameter that is a Boolean value may be specified using `TRUE`, `1`, or `YES` for truth and `FALSE`, `0`, or `NO` for falsity.

Directory Structure

The configuration and logging information for the Backup Utilities is organized under a single directory, the directory specified in the `SFS_BACKUP_CFG_ROOT` environment variable. The main directory structure is as follows:

```
SFS_BACKUP_CFG_ROOT/
        master.cfg
        drive.cfg
        library.cfg
        tape.cfg
        master.log
        lock/

        servers/
                <server1 name>/
                .
                . (a directory for each SFS server to be backed up)
                .
                <serverN name>/
                        server.cfg
                        copy.cfg
                        lock/

                        destinations/
                                directories
                                drives
                                tapes/
                                        <tape1 name>
                                        .
                                        . (a file for each tape used for this SFS server's backups)
                                        .
                                        <tapeN name>
```

The formats of many of these files are outlined in detail below. Here is a description of the general purpose of each file:

master.cfg

Contains configuration information common to all SFS servers serviced by this installation of the Backup Utilities.

drive.cfg

Contains configuration information for all tape drives used by this installation of the Backup Utilities.

library.cfg

Contains configuration information for all tape libraries used by this installation of the Backup Utilities.

tape.cfg

Contains configuration information about each tape to be used in making backups for SFS servers.

master.log

The master log file for which log messages are written for all backup activities.

lock (directory)

Holds lock files used to control concurrent access to resources, like the master log file. Each server-level directory also has a lock directory.

servers (directory)

Contains a directory for each SFS server to be managed by the Backup Utilities.

server.cfg

Contains configuration information for one SFS server's backups.

copy.cfg

Contains configuration information for each copy to be made of one SFS server's backups.

destinations (directory)

Contains information logging what backup files have been logged to what destinations. The information contained under this directory is updated by the run script on each backup run. It is not generally intended to be manually modified.

directories

This file maintains a record of the state of each directory copy of this SFS server's backups, including what log archive and data volume files have been backed up to each directory configured as a backup destination for this SFS server.

drives

This file (together with the files in the “tapes” directory) maintains a record of the state of each drive copy of this SFS server’s backups, including an indication of which tape is "active" for each backup copy of this SFS server's metadata. The active tape is the one which will be used next in making backups.

tapes (directory)

This directory contains a file for each tape defined in the Tape Configuration File that records what files are currently on that tape.

Overall Configuration

This section describes configuration actions that must be performed that are common to all SFS server instances to be backed up by the Backup Utilities.

Several configuration files must be created under the SFS_BACKUP_CFG_ROOT directory to configure entities that belong to all SFS servers.

Master Configuration File

The Master Configuration File is named master.cfg. It contains configuration information common to all SFS servers handled by this installation of the Backup Utilities. The following parameters are set in the Master Configuration File:

Mandatory Fields:

ADMIN_EMAIL

One or more email addresses which should be contacted with information about any critical errors encountered in the backup process and with status reports about the completion of backup runs and purges.

MAIL_ERRORS_ONLY

Flag indicating whether or not summary information should be mailed to the administrator addresses at the end of each backup run or purge. If this flag is TRUE, only error messages will be emailed to the administrator addresses.

TMP_DIR

The full pathname of the directory that should be used by the Backup Utilities for creating temporary files (e.g. /tmp, /var/tmp). This path should not indicate a directory that is shared among multiple hosts (by NFS, for example), if the Backup Utility configuration is shared among multiple hosts. The reason for this is that the Utilities store temporary files in this directory identified by process ID. If Backup Utilities running on multiple hosts share this directory, it is possible that a program running on one host could overwrite a temporary file belonging to a program running on another host, producing unpredictable behavior.

LOG_TRIM_THRESHOLD_LINES and LOG_TRIM_TARGET_LINES

Together, these two parameters control the trimming of the master log file. Whenever a Run Script executes, it checks the line count of the master log file to see if it exceeds LOG_TRIM_THRESHOLD_LINES; if it does, the log file is trimmed down to only retain its last LOG_TRIM_TARGET_LINES lines.

COMPRESS_NICE_INCREMENT

The amount by which to increment the base process priority number when executing commands to compress LA or TRB files.

MAX_TRACKED_TRBS

Specifies the maximum number of TRB files that an SFS server should have to track at any given time. If the Run Script detects that it cannot generate more TRB files without pushing its SFS server over this limit, it will forego the generation of new TRB files and log an error. This parameter has a limit which is the maximum number of TRB files that an SFS server can actually track in memory (currently 512). If the SFS server exceeds this number of TRB files, it must dump some of its backup record information into the LA files. Then, operations that require the SFS server to access backup record information can cause the server to ask for LA files which have been moved offline to be brought back online, halting the progress of the SFS server until this has been done.

OS_TYPE

Specifies the type of operating system under which the Backup Utilities are running. This information is used to allow the Backup Utilities to adjust their expectations of the invocation and output of various external programs they use (like `tar` and `df`). Currently, the only supported value is `AIX`.

Optional Fields:

COMPRESSION_PROGRAM

Full pathname of compression program to use. Should function like **compress** or **gzip** - compress a file "in place." If this parameter is not specified, the default compression program is **/usr/bin/compress**.

COMPRESSED_EXTENSIONS

This is a list of extensions (without periods) that files will have after they have been compressed by the `COMPRESSION_PROGRAM`. For example, if gzip is used, this parameter should include “gz”. The default value is just “Z”.

Note that these two parameters must both be specified or neither one must be specified.

In Preparation for Using Remote Devices

The next few sections describe how to configure the utilities to use tape drives and various types of automated tape libraries. Some of the parameters are only applicable to “remote” devices. In general, this means devices that are not directly accessible from the host on which the Monitor and Run Script will run, whether or not the device is physically attached to that host. Keep the following in mind when determining whether a device is remote or not.

For tape drives, the host that is the “local” host for the drive is the host where the drive’s device filename (e.g. `/dev/rmt0`) is valid.

For IBM 3494 tape libraries, the “local” host is the host where the library’s device filename (e.g. /dev/lmcp0) is valid. This is the host where the **mtlib** utility can be run to control the library. The Run Script uses **mtlib** to control 3494 libraries.

For STK libraries, the “local” host is the host where the SSI software runs. The Run Script will run the **stk_ctl** utility on this host to control the STK library.

For AML libraries, the “local” host is the host where the Run Script will run the **aml_ctl** utility to control the library, which is typically the same host where the Run Script will run. Note that this is not the hostname of the OS/2 machine where the DAS server runs - that hostname is what will be specified under the DAS_SERVER parameter in library.cfg (see below).

Library Configuration File

The Library Configuration File is named library.cfg. It consists of a series of stanzas, each one specifying parameters for one tape library or robot to be used in making backups. Each stanza specifies the following parameters:

Mandatory Fields:

TYPE

A string specifying what type of device this library is (e.g. 3494, STK, AML, or OPERATOR).

Libraries have zero or more required parameters in addition to TYPE that depend on what type the library is:

3494:

DEVPATH

The pathname of the device special file for the library (e.g. /dev/lmcp0).

AML:

DAS_CLIENT

The hostname of the host where the **aml_ctl** utility will be run. Typically this is just the hostname of the host where the Run Script will run.

DAS_SERVER

The hostname of the OS/2 machine where the DAS server software runs.

OPERATOR:

The following parameters only need to be specified in the definition for a "library" of type OPERATOR, which is used to configure drives where tapes must be operator mounted, or for any device for which support is not built into the Backup Utilities but can be provided through locally-supplied "hook" programs. Each one specifies the full pathname of a site-supplied external program or script provided to handle this situation.

See the section "Hook Program Interface" below for what the inputs and outputs of these programs should be.

OPERATOR_MOUNT_HOOK

The program to run for mount requests to drives in this library.

OPERATOR_QUERY_HOOK

The routine to be run for queries to determine what tape (if any) is mounted in a drive in this library.

OPERATOR_UNMOUNT_HOOK

The routine to run for dismount requests to drives in this library.

Optional Fields:

The following three parameters are only necessary if the library is associated with a “remote” host. See the previous section for a description of what devices are considered “remote”.

REMOTE_COMMAND

The command to be used to access the library on the remote host with which it is associated. This parameter must begin with an absolute pathname. For example if rsh is to be used to communicate with the library on the remote host, this parameter would be set to `"/usr/bin/rsh <remotehost>"`. Note that this command must be able to operate without any interactive input, such as prompting for a password. Many sites consider the use of rsh, with or without password authentication, to be insecure. Sites may want to investigate using the freely available ssh utility, which functions similarly to rsh, but uses encryption to achieve a higher level of security.

REMOTE_BINDIR

The path where Backup Utility executables can be found on the remote host. This is necessary in order to run the **aml_ctl** or **stk_ctl** utilities on the remote host. The parameter itself is optional. If omitted, the Utilities will assume that the Backup Utility executables are at the same path name on the remote host as on the local host.

MOUNT_RETRIES

This variable specifies the number of times that the utilities should retry mount operations that fail for drives associated with this library. The default value is 3.

Drive Configuration File

The Drive Configuration File is named drive.cfg. It consists of a series of stanzas, each one specifying parameters for one tape drive to be used in the backup process. Each stanza specifies the following parameters:

Mandatory Fields:

DEVPATH

The pathname of the drive's device special file, e.g. /dev/rmt0.

ROBOT_ADDRESS

A string specifying the "address" which is used to identify the drive within its library. This may vary in format depending on what kind of library the drive resides in.

LIBRARY

The name of the library which controls this drive. This parameter must match one of the stanza names from the Library Configuration File.

Optional Fields:

The following three parameters are only necessary if the drive is associated with a “remote” host. See the previous section for a description of what devices are considered “remote”.

REMOTE_COMMAND

The command to be used to access the drive on the remote host with which it is associated. This parameter must begin with an absolute pathname. For example if rsh is to be used to communicate with the library on the remote host, this parameter would be set to "`/usr/bin/rsh <remotehost>`". If the Run Script needs to issue the command “`tctl –f /dev/rmt0 rewind`” to rewind a tape in the drive, it will actually issue “`/usr/bin/rsh remotehost tctl –f /dev/rmt0 rewind`”. Note that this command must be able to operate without any interactive input, such as prompting for a password. Many sites consider the use of rsh, with or without password authentication, to be insecure. Sites may want to investigate using the freely available ssh utility, which functions similarly to rsh, but uses encryption to achieve a higher level of security.

REMOTE_BLOCKSIZE

The number of 512-byte blocks to use as the block size when tarring data to this drive across the network. Please note any restrictions that your operating system’s version of **tar** places on the block size that can be used. For example, AIX restricts this value to a maximum of 20.

REMOTE_BINDIR

The path where Backup Utility executables can be found on the remote host. This is necessary in order to run the internal_label_read utility (and, in the case, of HPSS-controlled drives, the hpss_drive_ctl utility) on the remote host. The parameter itself is optional. If omitted, the Utilities will assume that the Backup Utility executables are at the same path name on the remote host as on the local host.

The following two parameters are optional. They are used to control the Run Script's behavior when trying to acquire a drive in order to make a backup copy.

GRAB_TIMEOUT_SECS

The number of seconds to wait between each attempt to acquire a drive. The default is 300.

MAX_GRAB_TIMEOUTS

The number of times `GRAB_TIMEOUT_SECS` should be allowed to elapse before the Run Script gives up on trying to acquire the drive. The default is 3.

The following parameters are optional; they are needed only to configure a drive that is controlled by HPSS. If such a drive is configured, then all of the following parameters must be specified.

HPSS_DRIVE_CDS

CDS pathname for drive's PVL.

HPSS_DRIVE_PRINCIPAL

DCE principal to use to enable / disable drive.

HPSS_DRIVE_KEYTAB

Keytab file to use to authenticate as `HPSS_DRIVE_PRINCIPAL`.

HPSS_DRIVE_PVL_DESC_NAME

Descriptive name of drive's PVL.

HPSS_DRIVE_CFG_FILE

CDS pathname of generic serverconfig file to use.

HPSS_DRIVE_ID

Numeric HPSS drive ID of drive.

Tape Configuration File

This file is named tape.cfg. It is used to configure all tapes that are used for making backups for SFS servers. The file consists of a series of stanzas, each one specifying configuration parameters, either for one tape or for a range of tapes.

A stanza which defines a single tape begins with `TAPE <ext. label>` and ends with `END_TAPE`. It specifies the following parameters:

Mandatory Fields:

LIBRARY

The name of the library which will be used to mount this tape. This parameter must match one of the stanza names from the Library Configuration File.

MEDIA_TYPE

Specifies the media type of the tape (e.g. 3590). This is needed by some library types (currently only the AML.), but is ignored by others.

CAPACITY_KB

This is the estimated capacity of the tape in kilobytes. This parameter is used by the Utilities in order to fit files onto tapes such that end-of-tape is not encountered while writing a tar archive.

Optional Fields:

INTERNAL_LABEL

What is stored in the volume ID field of the tape's internal label. After a tape is mounted, its internal label is read and checked against this value to guard against accidentally overwriting another tape if the wrong tape is mounted. If this parameter is not specified, the internal label will be assumed to be the same as the external label (i.e. the NAME of the tape).

MAX_OVERWRITES

Each time a tape is overwritten from the beginning, a count of the number of times this has happened is updated. This information is stored in the tape's log. If this parameter is specified, when the count of overwrites exceeds its value, the tape will be considered "retired" and no longer available for backups. If this parameter is not specified, this check will not be performed for the tape.

A stanza which defines a range of tapes begins with TAPE_RANGE <prefix> and ends with END_TAPE_RANGE. It can specify each of the parameters that can occur in the stanza for a single tape, except INTERNAL_LABEL. The internal label for each tape in the range is assumed to be the same as its external label. In addition, the stanza must specify the following two parameters:

Mandatory Fields (Tape Range):

START

Specifies what should be appended to the prefix to make the name of the first tape. This must be a numeric value.

END

Specifies what should be appended to the prefix to make the name of the last tape in the range. This must be a numeric value.

Example:

Suppose one needs to define tapes named A00050 through A00888 (an outrageous number of tapes, but for the sake of example…). The following stanza could be used.:

```
TAPE_RANGE A00
  START=50
  END=888
  LIBRARY=huge_library
  MEDIA_TYPE=3590
  MAX_OVERWRITES=30
END_TAPE_RANGE
```

This will configure tapes A00050 through A00888, each of which is assumed to have an internal label the same as its external label, will reside in the library defined by huge_library, and will be retired after they have been overwritten thirty times. Note that the suffixes (050 – 888) are padded out to the length of the longer of the two values of the START and END parameters.

Per-SFS Server Configuration

A configuration name must be chosen for each SFS server that the Backup Utilities are to manage. This is the name that was passed to the build_cfg utility to build the directory structure for the SFS. This configuration name is what will be passed to the Backup Utility scripts with the –s option in order to identify the server.

Main Server Configuration File

Under the server-level directory, a file called server.cfg must be created. This file contains configuration parameters for the server_t representing the SFS server. The following parameters are specified in the file:

Mandatory Fields:

CDS_NAME

CDS name of the SFS server. (e.g. `/.:/encina/sfs/someserver`)

LOG_VOLUME

The name of the SFS server's log volume (as it appears in the output of the "`tkadmin list lvols`" command).

NUM_BACKUPS_TO_RETAIN

The number of complete backups that the Run Script should instruct this SFS server to retain.

RUNS_PER_BACKUP

The Run Script will attempt to break up a complete backup for the SFS server's data volumes into this many runs.

MONITOR_INTERVAL_SEC

The number of seconds that this SFS server's Backup Monitor should sleep in between checks on the SFS server's log archive and data volume backup filesystems.

TAPE_ADVANCE_DAYS

If the Run Script has been writing on the same tape in a copy's tape pool for at least this many days, it will advance to the next tape in the pool.

COMPRESS_LA_FILES

A boolean value indicating whether or not any uncompressed log archive files (except the newest) for this SFS server should be compressed.

COMPRESS_TRB_FILES

A boolean value indicating whether or not any uncompressed data volume backup files for this SFS server should be compressed.

HPSS_LOGGING

A boolean value indicating whether or not messages about completion of backup runs and purges should be logged to the HPSS logging subsystem. Note that the `HPSSMSG` option in the `Makefile.options` must have been enabled before the make was done so that the hpssmsg utility will be built.

SFS_CONFIG_FILES

A list of files to be backed up on each backup run. This list is intended to contain any files necessary to restore the SFS configuration.

ADMIN_DEFINED_FILES

A list of files to be backed up on each backup run. This is for any other files that are to be backed up on each run.

DCE_PRINCIPAL

The name of the principal under which any commands requiring DCE privileges are to be run for this SFS server.

KEYTAB_FILE

The pathname of the keytab file which is to be used to authenticate the identity of the `DCE_PRINCIPAL` to DCE.

LA_DIR

The full pathname of the directory containing this SFS server's log archive files.

LA_PURGE_START_THRESHOLD

The percentage of usage in the filesystem containing the log archive files for this SFS server at which its Backup Monitor should begin purging.

LA_PURGE_STOP_THRESHOLD

The percentage of usage in the filesystem containing the log archive files for this SFS server at which its Backup Monitor can stop purging.

TRB_DIR

The full pathname of the directory containing the data volume backup files for this SFS server.

TRB_PURGE_START_THRESHOLD

The percentage of usage in the filesystem containing the data volume backup files for this SFS server at which its Backup Monitor should begin purging.

TRB_PURGE_STOP_THRESHOLD

The percentage of usage in the filesystem containing the data volume backup files for this SFS server at which its Backup Monitor can stop purging.

MIN_TRB_SIZE_KB

The minimum size, in kilobytes, to which to constrain the data volume backup files that are generated for this SFS server.

MAX_TRB_SIZE_KB

The maximum size, in kilobytes, to which to constrain the data volume backup files that are generated for this SFS server.

SLEEP_BETWEEN_TRBS_SEC

The number of seconds that the Run Script should wait between generating each data volume backup file for this SFS server.

NO_NEW_TRB_THRESHOLD

Percent of space usage in the filesystem containing the TRB directory at which the run script will no longer generate any new TRB files.

Optional Fields

PURGE_RELEVANT_LA_FILES

A Boolean value indicating whether or not log archive files that are still relevant (i.e. are needed for a restoration) should be purged by the monitor when the LA_PURGE_START_THRESHOLD is reached.

Copy Configuration File

The Copy Configuration File is named copy.cfg and resides under the server-level directory. This file consists of a number of stanzas, each of which specifies one copy of the SFS server's data that is to be made. Each copy can be made to either a directory or to a pool of tapes. The parameters specified in the stanza differ depending on whether the destination is a directory or a tape pool.

Mandatory Fields (if copy's destination is a directory):

DIRECTORY

The full pathname of the directory to which this backup copy should be backed up.

Mandatory Fields (if copy's destination is a tape pool):

DRIVES

A list of names of drives to use for backing up this copy. Each name must be one of the stanza names from the Drive Configuration File. The drives will be tried in the order listed until one is available to make the copy. All the drives must be associated with the same library as the copy's tapes.

TAPES

A space-separated list of the tapes to be used to make this backup copy. Each item in the list must specify the name of one of the tapes defined in the Tape Configuration File. The order of the list is significant: the Run Script will attempt to cycle through the tapes in the order they are given.

The Utilities will attempt to detect when more than one server (or more than one copy within a server's configuration) is using the same tape and log an error to that effect. However, the Utilities have no way of detecting the case where the same tape is being used by two different servers defined in different configuration directories (such as on different machines), so care must be taken to ensure that multiple servers are not trying to use the same tape.

FREE_TAPE_THRESHOLD

When the number of free tapes in this copy's tape pool falls below this number, a warning is issued to the administrator.

6.11.6 Hook Program Interface

Whenever a library is configured as type OPERATOR (whether it really does require that its drives' tapes be operator-mounted or it is just a device which is not explicitly supported by the Backup Utilities) three programs must be specified for the parameters OPERATOR_MOUNT_HOOK, OPERATOR_UNMOUNT_HOOK, and OPERATOR_QUERY_HOOK.

Each one of these programs should return an exit status of 0 for success or nonzero for failure.

In addition, when the OPERATOR_QUERY_HOOK program executes successfully, it should print out simply the name of the tape mounted or "No volume mounted" if no tape is mounted.

Each program should expect to receive the following arguments (whether or not it uses them):

-l <library device>

Where <library device> is the device pathname of the library as specified in the library's DEVPATH configuration parameter.

-d <drive device>

Where <drive device> is the device pathname of the drive involved in the request as specified in the drive's DEVPATH configuration parameter.

-a <drive address>

Where <drive address> is the library address of the drive involved in the request as specified in the drive's ROBOT_ADDRESS parameter.

-t <tape label>

Where <tape label> is the external label of the tape (i.e. its configuration name) of the tape to mount. This parameter is only passed for a call to the OPERATOR_MOUNT_HOOK.

-m <media type>

Where <media type> is the MEDIA_TYPE parameter of the tape to be mounted. This parameter is only passed for a call to the OPERATOR_MOUNT_HOOK.

6.11.7 Recovering SFS Data from Backup Utility Archives

In the event that an SFS server's volumes need to be restored from backed up data, it will be necessary to retrieve that data to online storage in order to perform the recovery procedure. This section discusses how to retrieve archived data from tape media to online storage. For instructions on how to actually perform the recovery procedure for an SFS server, please see the appropriate Transarc documentation.

As mentioned already, there are two types of backup copies: those made to a directory in the filesystem, and those made to a pool of tapes. Presumably, if a copy is being made to a directory, the administrator arranges for the files there to be backed up to offline storage by some external means (in order to keep the filesystem from filling up). The procedures to recover files from a copy made to a directory will vary, so administrators who choose this backup option should plan in advance how they will recover that data.

In the case of a copy made to a pool of tapes by the Backup Utilities, it is necessary to know how to recover the data stored on those tapes. A Backup Utility tape has the following format:

```
<Beginning of Tape>
<Internal Label>
Tape Mark
<tar format archive>
Tape Mark
<tar format archive>
Tape Mark
.
.
.
<tar format archive>
Tape Mark
<End of Tape>
```

There may be zero or more occurrences of the "<tar format archive>, Tape Mark" sequence. This is the normal format of a Backup Utility tape. In some circumstances, (specifically, if the CAPACITY_KB parameter of a tape is overestimated), the Backup Utilities may write a tape all the way out to the end of tape. In that case, the format is the same, but the tape mark after the last tar archive will not be present. Special considerations must be taken into account when extracting files from such an archive; they are described in detail below.

The Backup Utilities record the contents of each backup tape, plus some extra information, in files called tape contents logs. These are text files with the same name as the external label of the tape to which they correspond. These files are located in the directory *<configuration dir>*/servers/*<server name>*/destinations/tapes. These are all text files and can be viewed with any text editor, but they shouldn't be manually edited, since a mistake in editing these files could result in the Backup Utilities being unable to determine the state of the backups from them.

Tape contents log files can either be empty if a tape has never been used, or they can have the following format:

```
OVERWRITES <number>
```

FIRST_WRITE_DATE YYYY/MM/DD

then zero or more occurrences of:

TARFILE <number>
<filename> <size in bytes> <date written> <time written> <type code>
<filename> <size in bytes> <date written> <time written> <type code>
…

the file may end with:

EOT

Here is an explanation of each field:

OVERWRITES <number>

Specifies the number of times this tape has been overwritten from the beginning (including the first time it is written). This is used to facilitate retiring tapes after a certain number of overwrites, using the MAX_OVERWRITES parameter of the Tape Configuration File.

FIRST_WRITE_DATE YYYY / MM / DD

This is the date that the tape was written on the last time that it was overwritten. This is used to facilitate advancing to the next tape in the tape pool after a certain number of days using the TAPE_ADVANCE_DAYS parameter of the Server Configuration File.

TARFILE <number>

This specifies that all the file specifications that follow, up until the next TARFILE line or the end of the file, belong to a single tar archive. The number specifies the ordinal position of the tar archive on the tape (starting at 1).

<filename> <size in bytes> <date written> <time written> <type code>

This specifies a single file that is part of a tar archive. All the fields should be self-explanatory except for the type code, which may have one of the following values:

T – the file is a TRB file

L – the file is a complete LA file; i.e. it was not the newest LA file at the time it was archived

I – the file is a possibly-incomplete LA file; i.e. it was the newest LA file at the time it was archived

O – the file does not belong to any of the above categories

EOT

If the file ends with EOT, that means that the Backup Utilities reached the end of tape while writing the last tar archive on the tape.

The first step in recovering data from the Utilities' backup tapes is to determine what files need to be brought back online. At the end of every backup run, the Run Script logs (and emails, if so

configured) the state of the backup as reported by the `tkadmin query logvol` and `tkadmin query backup` commands. The Utilities also contain a tool to report the locations of various backup files on the tapes comprising a tape copy. This tool, **sfs_backup_report**, can be run in one of two ways.

For restoring a log volume (and determining the locations where various LA files are stored) – this is the default if neither –l nor –d is specified:

sfs_backup_report –s server –c tape_copy -l

-s server

This option specifies the configuration name of the server for which the report is being performed.

-c tape_copy

This option specifies the configuration name of a tape copy associated with the specified server.

When querying about the restoration information for a log volume, the SFS server does not have to be running at all. The tool uses the backup logs' record of what LA files are needed to restore the log volume (updated on each backup run).

For restoring a data volume (and determining the locations where various TRB files are stored):

sfs_backup_report –s server –c tape_copy –d data_vol [-n which_backup]

-s server

This option specifies the configuration name of the server for which the report is being performed.

-c tape_copy

This option specifies the configuration name of a tape copy associated with the specified server.

-d data_vol

This option specifies the name of the data volume to be restored.

-n which_backup

If the value specified for the `NUM_BACKUPS_TO_RETAIN` parameter in the Server Configuration File is specified to be greater than 1, the Utilities will instruct the SFS server to retain more than one complete backup of each data volume. This option can be used to specify that the tool should report the locations of TRB files associated with the n'th newest backup (1 being the newest and the default).

In either usage, the –L option can be used to cause the tool to print a "long format" listing (see below). Also, if there is only one possibility for the server or copy, there is no need to specify those through parameters.

When querying about the restoration information for a data volume, the SFS server does need to be running so that the tkadmin query backup command may be issued to determine what TRB files are needed. Please consult official Transarc documentation to determine what state the SFS server should be put in while restoring a data volume.

The output of the tool indicates each tape containing needed files, each tarfile containing those files, and a list of the files in the tarfile. For example:

```
$ sfs_backup_report –s sfsServer1 –c tapeCopy1 –d datavol8
Server: sfsServer1
Copy: tapeCopy1
Using backup 1 of 2 (1 is newest): TRBS 00128-00138
Data volume: datavol8
TAPI A00100
TARFILE 11
00128-00130
TAPE A00101
TARFILE 1
00131-00136
TARFILE 5
00137-00138
```

Here is an example of the type of report generated with the –L option:

```
$ sfs_backup_report –L –s sfsServer1 –c tapeCopy1 –d datavol8
Server: sfsServer1
Copy: tapeCopy1
Using backup 1 of 2 (1 is newest): TRBS 00128-00138
Data volume: datavol8
TAPE A00100
TARFILE 11
/backup/TRB/sfs1/datavol8.00128.Z 158102 1999/06/18 11:15:27 T
/backup/TRB/sfs1/datavol8.00129.Z 158102 1999/06/18 11:17:20 T
/backup/TRB/sfs1/datavol8.00130.Z 158102 1999/06/18 11:21:08 T
TAPE A00101
TARFILE 1
/backup/TRB/sfs1/datavol8.00131.Z 158102 1999/06/18 11:24:27 T
/backup/TRB/sfs1/datavol8.00132.Z 158102 1999/06/18 11:28:20 T
/backup/TRB/sfs1/datavol8.00133.Z 158102 1999/06/18 11:32:08 T
/backup/TRB/sfs1/datavol8.00134.Z 158102 1999/06/18 11:36:27 T
/backup/TRB/sfs1/datavol8.00135.Z 158102 1999/06/18 11:41:20 T
/backup/TRB/sfs1/datavol8.00136.Z 158102 1999/06/18 11:46:08 T
TARFILE 5
/backup/TRB/sfs1/datavol8.00137.Z 158102 1999/06/18 17:15:27 T
/backup/TRB/sfs1/datavol8.00138.Z 158102 1999/06/18 17:17:20 T
```

This indicates that the TRB files that are needed to restore `datavol8` from the tapes in copy `tapeCopy1` can be found on the 11th tarfile on tape `A00100` and on the 1st and 5th tarfiles on tape `A00101`. The output from this utility is helpful for tracking the status of the backups. A good idea may be to set up a **cron** job to run this utility and mail its output to the administrator on a regular basis, perhaps daily.

Once the locations of the needed files on various tapes has been obtained, the files can be extracted from tape using the following steps:

- Mount the tape containing the files.
- Rewind the tape.

- Seek forward past a number of tape marks equal to the ordinal number of the tar archive containing the files. Thus, if the file is in `TARFILE 1` and the tape is mounted in the drive with device pathname /dev/rmt0, then the command

$ mt –f /dev/rmt0 fsf 1

would advance to the beginning of the tar archive containing the file. Note that this is because the tape's internal label always occupies the area before the first tape mark on the tape.

At this point, a utility such as **tar** or the POSIX archive tool **pax** can be used to extract the files from the archive.

If the tarfile being extracted from is the last on the tape, and the tape log contains the `EOT` line as its last line, then the last tarfile is not "complete". Any files that the tape log says are contained in the last tarfile may be extracted from it. However, at some point the tar command will reach the end of the tape and may ask for the "next tape" to be mounted. The tar command should be killed at this point. The Backup Utilities do not span tarfiles cleanly from tape to tape. To avoid this annoying situation, take care not to overestimate the `CAPACITY_KB` parameter in the Tape Configuration File.

As a practical consideration, when retrieving files from tar archives, it makes sense to group files by tape and tar archive and to use a single **tar** or **pax** command to extract all files in that archive at once. This minimizes the amount of mounting and dismounting of tapes, as well as rewinding and seeking.

6.11.8 A Note About SFS Volume Configuration

The Utilities are intended to be used primarily on a stable SFS volume configuration. It is highly recommended that the volume configuration for the SFS be settled upon before the Utilities are used to back up that SFS. The only change to a volume configuration that the Utilities can handle well without reconfiguration is the addition of one or more data volumes. Before adding any data volumes to an SFS being backed up by the Utilities, all backup utility scripts for that server should be terminated (and precautions taken so that they do not start up while the reconfiguration of the volumes is going on). After the addition of the data volumes, the backup scripts can be re-enabled with no changes to the Backup Utilities' configuration. The new data volumes will be backed up along with the old.

If an SFS's volume configuration is changed in any way other than simply adding data volumes, this situation should be treated as though the backup process is being started for a whole new SFS server. A new configuration directory should be created and prepared. If the volume reconfiguration causes the SFS to restart its sequence numbering of LA files, any old LA files that are online should be deleted (since they are now useless). These old LA files could be mistaken for current ones and backed up.

6.11.9 Recovery from Failed Backup Runs

Under most circumstances, little, if any, intervention should be required to keep the Backup Utilities functioning properly. The Backup Utilities can fail in two major ways. First, some condition could be detected that keeps them from continuing, such as an error in a configuration file or running out of some resource (like the pool of tapes for a copy filling up). Second, a catastrophic system failure can cause the Utilities to fail without having a chance to shut down

properly. In the first case, even if the backup activity is not performed, the log information about the state of what is backed up should be consistent. The email addresses defined in the ADMIN_EMAIL parameter should receive a message summarizing what went wrong. This message and the master log can be examined to determine the cause of the failure. The problem can then be corrected, and, if desired, the run script can be invoked in "no new TRBs" mode (-n option) to back up any outstanding files, though this is not absolutely necessary. The run script will always try to back up outstanding files even when invoked in normal mode.

In the second case, that of a system failure, the backup log information may indeed not accurately reflect the state of the backup process at the instant of failure, but no permanent harm should be done. At worst, on the next run, some backup activity that was performed on the last run may be performed again. For example, when the run script is about to overwrite a tape, it first truncates that tape's contents log to ensure that if a failure occurs, the log will not still list the old files being on the tape when they have, in reality, been overwritten. So on the next run, the same tape will be used, and possibly some of the write activity that was performed on the last run will be performed again.

6.11.10 Communicating With the Utilities While They are Running

Preventing the Utilities from Running

Sometimes it is desirable to have the Utilities not run during a specific period of time. For example, a tape drive normally used for backups may be temporarily needed for some other purpose. There is a mechanism to prevent the Utilities associated with a particular SFS server from running: create the file *<configuration dir>*/servers/*<server name>*/STOP. When the Run Script is started, if this file is present, it will log a message and exit without performing any backup activity. When the Monitor is ready to make a purge sweep, if this file is present, it will log a message and go back to sleep until the next purge sweep without compressing or purging any files. To re-enable backup activity, just delete the stop file. Note that creating this file will not stop a backup run or purge sweep that is already in progress.

Signaling the Monitor to Reread the Configuration

It may also be desired to make the Monitor reread the configuration without having to kill it and restart it. This can be accomplished by sending the SIGHUP signal to the Perl interpreter running the Monitor. If the Monitor is in the midst of a purge sweep, before it begins its next purge sweep, it will reread the configuration. If the Monitor is sleeping between purge sweeps, this signal will cause it to wake up, reread the configuration, and immediately begin the next purge sweep. Note that if there are errors in the configuration, when the Monitor tries to reread the configuration, it will terminate and log the errors it found. It is therefore a good idea to use the Configuration Checker on the configuration directory before signaling the Monitor.

6.11.11 Miscellaneous Caveats

Manipulating the Backup Directories While the Scripts are Running

It is inadvisable to manipulate the files in either the LA or TRB directories while the backup scripts are running. First, compressing or uncompressing TRB or LA files may interfere with the backup scripts doing the same thing, possibly confusing them. Also, keep in mind that the Monitor's purging activity could delete older backup files inadvertently. Consider the following scenario: for whatever reason, you want to bring some older TRB files online in the TRB directory. If these TRB files are older than the current set being tracked by the SFS server, the Monitor quite possibly could delete them if space usage in the filesystem rises high enough.

Why Aren't the Scripts Running?

If the scripts seem to not be running when they ought to be, make sure that the STOP file is not present. It's easy to forget to remove it. If this happens, however, the Run Script should send an email message when it terminates. At any rate, examine the master.log file for messages about the STOP file's being present.

If the Run Script seems not to be running and no email messages are being sent, it is probably due to a configuration error. The scripts cannot send email messages using the ADMIN_EMAIL addresses until the configuration is read successfully. Check the master.log file for messages about configuration errors. Also, use the –m flag to the scripts to specify fallback email addresses to use in the case of configuration errors.

Why Don't My Remote Devices Work?

See the section on using remote devices to make sure that you have them configured correctly. Check your REMOTE_COMMAND parameter(s) manually on the command line, using the same user ID as that under which the Utilities run. Remember that the REMOTE_COMMAND parameter must be able to work without prompting for a password. Also check that the REMOTE_BINDIR parameter(s) are correct and that the programs contained in them are executable by the Utilities' user ID.

Only the Monitor Handles SIGHUP

Keep in mind that only the Monitor handles the HUP signal. This signal (and almost any other) will cause the Run Script to terminate ungracefully.

Sharing Resources Between Multiple SFS Servers' Backup Scripts

Beware of resource contention between the backup scripts for different SFS servers. If multiple servers use the same configuration directory, the utilities can detect when two servers (or even two copies within the same server) are configured to use the same tapes. When this happens, an error will be logged and the script will terminate. The problem comes when the same tape is allocated to two different servers using different configuration directories. The scripts for the two servers have no way of knowing that the tape is already being used, so they will both try to use it, resulting in needed data being overwritten. This could quite possibly not be noticed until a restoration involving that tape is attempted. The backup utilities do not support sharing of tapes between multiple servers. It is the administrator's responsibility to ensure that a tape belongs exactly to one copy of one server.

Tape drives are also not simultaneously sharable between different servers, in general. If two servers with different configuration directories attempt to use the same drives at the same time, they will contend for the drives, likely resulting in failed backup runs for both. Two servers that are using the same configuration directory behave slightly more intelligently. At least one will recognize that the drive is being used by another server and is unavailable. However, it is still a race that determines which server actually locks the drive first and gets to use it. The other server will only use the drive after the first if the timeout on the drive is long enough (see the MAX_GRAB_TIMEOUTS and GRAB_TIMEOUT_SECS parameters in the Configuration section). The moral of the story is that the second server's backup run should be started at a time that gives the backup run of the first a reasonable chance of finishing with the drive.

It may be desirable to have the backups for two servers that run on different hosts share a configuration directory by placing that directory on a filesystem mounted via NFS. There are a couple of issues with doing this. First, the TMP_DIR parameter of the master.cfg must not indicate a directory that is NFS mounted, since the scripts store temporary files here whose names are constructed from process ID numbers, which may not be unique between different hosts. For example, if /tmp is used, there should be a separate /tmp on a different filesystem on each machine (which is usually the case anyway). The other issue is that the utilities use file locking on lock files contained in the configuration directory to coordinate certain activities. This is true even between the Run Script and Monitor for a single server. If the configuration directory is stored on an NFS mounted partition, the NFS implementation must support file locking over the network.

Using the "Retain Drives" Option of the Run Script

The –r option to the Run Script can be useful in cases where it is desired to have the Run Script run several times in succession without the overhead of dismounting and remounting tapes. Using this flag also causes any HPSS-controlled drives to remain locked until the Run Script executes a backup run where the –r flag is not specified. Take care when using the –r flag. Consider the following undesirable scenarios:

An SFS server's backup activity is broken into several invocations of the Run Script each day. Each invocation runs soon after the previous one, and all but the last use the –r option. When everything goes as planned, the HPSS drives used by the backups will remain disabled from the time the first invocation begins its backup activity until the last invocation ends its. One day, the administrator needs to use one of the drives just before the final invocation begins, so she creates the STOP file and goes about her business. While she is working, the last invocation of the Run Script starts up, sees the STOP file, and terminates without doing anything to its drives. The disabled drives will now remain disabled until the end of the run cycle the next day – that's bad. The drives will not only be unavailable to HPSS; they will be unavailable to other SFS backup runs – even if they use the same configuration directory. The administrator could have safely re-enabled the drives using SSM or the provided `hpss_drive_ctl` utility between backup runs after she has finished using the drive.

Consider the above scenario where the admin has re-enabled the drives so that they are usable to HPSS after she finishes her work. The following day, just before the first invocation of the Run Script (the STOP file has been removed by now), the admin re-disables one of the drives and begins some other script she has that does something with the drive. When the Run Script starts, it has a record that it disabled the drive on a previous run. The Run Script will see that the drive is disabled, assume that the drive is still disabled from its last run, and attempt to use it, possibly interfering with the admin's script. The admin should use the STOP file while she is running her script, or run it at a time that she knows will not conflict with the backup runs. Alternatively, she can explicitly

make it so that that server will not think it has that drive reserved when its Run Script starts – see the next section.

Releasing a Drive from Being Reserved by a Particular Server's Backups

As explained in the scenarios described in the previous section, using the –r option of the Run Script causes the script to keep a record of the fact that a particular drive is "reserved" by its server for purposes of SFS backup runs. If it is ever necessary to ensure that a particular drive is not reserved by a particular server, follow these steps:

Make certain no backup Run Script is currently running

Edit the file *<config dir>*/lock/reserved_drives. This file consists of a series of entries, one per line, of the form:

drive server

Edit the line for the drive in question. Either delete the line in order to ensure that the drive is not reserved to any server, or replace the current server name with the configuration name of another server, in order to reserve the drive for that server.

If the file is edited to make the drive not reserved for any server and the drive is an HPSS drive, use SSM or the hpss_drive_ctl utility to re-enable the drive.

6.12 Managing HPSS Users

After the HPSS system is up and running, the administrator must create the necessary accounts for the HPSS users. For a new HPSS user, a DCE account and an FTP account must exist before the user can access HPSS via FTP. In addition, if the HPSS user needs to use SSM, a SSM account must also be created before the user can use SSM. The SSM ID should be created only for the HPSS administrators and operators.

The HPSS User Management Utility (**hpssuser**) provided with HPSS can be used by the administrator to add, delete, and list the HPSS user accounts. The utility must run as **root** to acquire the necessary authority to create new AIX, FTP, and SSM IDs. The **cell_admin** password is also required to add a DCE ID. Refer to Appendix I for more information on how to invoke the **hpssuser** utility.

6.12.1 Adding HPSS Users

The **hpssuser** utility can be used by the administrator to add an AIX User ID, a DCE User ID, an FTP User ID, and an SSM User ID for an HPSS user if these IDs do not already exist. The **hpssuser** utility can be invoked to add all four types of User IDs for a user or to add an individual User ID.

Add all User IDs

The utility creates an AIX User ID, a DCE User ID, an FTP User ID, and an SSM User ID for an HPSS user.

Invoke the **hpssuser** utility as follows to add the required User ID for an HPSS user:

hpssuser -add <user> -all

The utility will prompt the user for the required data. Following is an example of adding all User IDs:

$ hpssuser -add user1 -all

DCE: Adding DCE User 'user1' ...

Enter cell_admin password :

Acquiring cell_admin credentials ...

Enter User's Full Name [John Smith]: User1 Test ID
Enter User's Password [user1]:
Enter Home Directory [/home/user1]:
Enter UID [888]: 785 Enter Group [hpss]:
Enter Organization [hpss]:

DCE: Principal 'user1' added
DCE: Account information for 'user1' added
DCE: User 'user1' (User1 Test ID) added to DCE Registry

FTP: Adding FTP User 'user1' ...
FTP: Creating directory for user ... Please wait

scrub> scrub> Umask set to 022
scrub> scrub> Destroying cell_admin credentials ...
FTP: User 'user1' added.

AIX: Adding AIX User 'user1' ...
Enter Group [hpss]:
AIX: User 'user1' added.

SSM: Adding SSM User 'user1' ...
Enter Hostname (where SAMMI Runtime resides) [hpss]:

Select SAMMI Security Level :
1. User
2. Privileged User
3. Operator
4. Admin

Enter Security Level [4]:

SSM: User 'user1' added.
SSM: The Data Server needs to be recycled for the changes to take effect.

Ensure that the Name Server and the Bitfile Server are up and running before adding the FTP or DCE User ID. The hpssuser utility will not be able to create the user's home directory if the Name Server service is not available.

Add an AIX User ID

The utility invokes the AIX **mkuser** utility to create the User ID and Password.

Invoke the **hpssuser** utility as follows to add an AIX User ID:

hpssuser -add <user> -aix

The utility will prompt the user for the required data. Following is an example of adding an AIX User ID:

```
$ hpssuser -add user1 -aix

AIX: Adding AIX User 'user1' ...

Enter User's Full Name [John Smith]: User1 Test ID
Enter User's Password [user1]:
Enter Home Directory [/home/user1]:
Enter UID [888]: 785
Enter Group [hpss]:

AIX: User 'user1' added.
```

Add a DCE User ID

The **hpssuser** utility invokes the DCE **rgy_edit** utility to create the DCE principal and account.

Invoke the **hpssuser** utility as follows to add a DCE User ID:

hpssuser -add <user> -dce

The utility will prompt the user for the required data. Following is an example of adding a DCE User ID:

```
$ hpssuser -add user1 -dce

DCE: Adding DCE User 'user1' ...
Enter cell_admin password :
Acquiring cell_admin credentials ...

Enter User's Full Name [John Smith]: User1 Test ID
Enter User's Password [user1]:
Enter Home Directory [/home/user1]:
Enter UID [888]: 785
Enter Group [hpss]:
Enter Organization [hpss]:

DCE: Principal 'user1' added
DCE: Account information for 'user1' added
DCE: User 'user1' (User1 Test ID) added to DCE Registry
```

If Site-style accounting is used, add the -aaid=<id> option to the above invocation.

The hpssuser utility set the DCE **pwdvalid** flag to "no". DCE user must changed their DCE password upon logging onto DCE for the first time.

Ensure that the Name Server and the Bitfile Server are up and running before adding the FTP or DCE User ID. The hpssuser utility will not be able to create the user's home directory if the Name Server service is not available.

Add an FTP User ID

The **hpssuser** utility adds a password entry in the FTP Password file and create the user's home directory in HPSS.

Invoke the **hpssuser** utility as follows to create an FTP User ID:

hpssuser -add <user> -ftp

The utility will prompt the user for the required data. Following is an example of adding an FTP User ID:

$ hpssuser -add user1 -ftp

FTP: Adding FTP User 'user1' ...

Enter User's Full Name [John Smith]: User1 Test ID
Enter User's Password [user1]:
Enter Home Directory [/home/user1]:
Enter UID [888]: 785
Enter Group [hpss]:

FTP: Creating directory for user ... Please wait
scrub> scrub> Umask set to 022
FTP: User 'user1' added.

Ensure that the Name Server and the Bitfile Server are up and running before adding the FTP User ID. The hpssuser utility will not be able to create the user's home directory if the Name Server service is not available.

Add an SSM User ID

The **hpssuser** utility creates an SSM Configuration directory for the user, creates the SSM User Configuration file (**ssm_console.dat**), adds the user's ID and security level in the SSM Authorization file (**user_authorization.dat**) and adds the user session's communication information into the **api_config.dat** file. Refer to [Section H.5] for more complete details on how the **hpssuser** utility sets up an SSM user. Refer to Section H.7 for more information on the SSM user security.

Invoke the **hpssuser** utility as follows to add an SSM User ID:

hpssuser -add <user> -ssm

The utility will prompt the user for the required data. Following is an example of adding an SSM User ID:

```
$ hpssuser -add user1 -ssm

SSM: Adding SSM User 'user1' ...

Enter User's Full Name [John Smith]: User1 Test ID
Enter Group [hpss]:
Enter Hostname (where SAMMI Runtime resides) [hpss]:

Select SAMMI Security Level :
  1. User
  2. Privileged User
  3. Operator
  4. Admin

Enter Security Level [4]:

SSM: User 'user1' added.
SSM: The Data Server needs to be recycled for the changes to take effect.
```

6.12.2 Deleting HPSS Users

The **hpssuser** utility can be used by the administrator to delete existing User IDs for an HPSS user. The utility can be invoked to delete all User IDs for the user or to delete an individual ID.

The utility will prompt the user for the required data. Following is an example of deleting the User IDs for an HPSS user:

```
$ hpssuser -del user1 -all

Enter cell_admin password :
Acquiring cell_admin credentials ...

DCE: User 'user1' deleted.
Destroying cell_admin credentials ...

FTP: User 'user1' deleted.

AIX: User 'user1' deleted.

SSM: User 'user1' deleted.
SSM: The Data Server needs to be recycled for the changes to take effect.
```

6.12.3 Listing HPSS Users

The **hpssuser** utility can be used by the administrator to list all existing HPSS User IDs. The utility can be invoked to list all HPSS User IDs or a particular type of User ID.

Following is an example of listing the User IDs for an HPSS user:

```
$ hpssuser -list user1 -all
```

```
[ DCE User Info ]
user1 [hpss hpss]:*:785:210:User1 Test ID:/home/user1::

[ FTP User Info ]
user1:!:785:0:User1 Test ID:/home/user1:/bin/ksh

[ AIX User Info ]
user1 id=785 pgrp=hpss groups=hpss,staff home=/home/user1 shell=/bin/ksh gecos=User1
Test ID

[ SSM User Info ]
# User: user1 (User1 Test ID)
logical_server s2_evtsvr 0x20000120 18 hpss 180 200
logical_server s2_mstalm 0x20000121 18 hpss 180 200
logical_server s2_stream 0x20000122 18 hpss 180 200
```

6.13 Using HPSS Utilities

In addition to using the SSM windows to manage HPSS, the following HPSS utilities are also available to help aid the administrator in managing the HPSS system. These utilities can be initiated from the command line on an AIX window or can be incorporated into **cron** jobs. Refer to Appendix I for more information on how to use the HPSS utilities.

archivecomp
archivedel
archivedump
archivelist
archiverec
auth_manager
backman
chacl
create_fset
create_fsys
crt_junction
del_junction
dump_sspvs
dumpbf
dumphpssfs
dumppv_pvl
dumppv_pvr
gen_reclaim_list
getdmattr
hdm_admin
hpss.clean
hdmdump
hpssuser
insif
loadhpssdmid
loadhpssfs
loadhpssid
loadtree
lsacl
lsfilesets
lshpss
lsjunctions
lsrb
lsvol
managesfs
metadata_info
mps_reporter
multinoded
nfsmap
nsde
pftp_client
pftpd
plu
reclaim
reclaim.ksh
recover
remove
repack
retire
scrub
setdmattr
settapestats
sfsbackup
shelf_tape

Chapter 7

DFS Configuration and Management

7.1 Introduction

Release 4.1.1 of HPSS provides distributed file system services by optionally allowing HPSS to interface with The Open Group's DFS™. DFS is a scalable distributed file system that provides a uniform view of file data to all users through a global name space. DFS supports directory hierarchies, and an administrative entity, called filesets. DFS also supports ACLs on both directories and files to allow granting different permissions to different users and groups accessing an object. DFS uses the DCE concept of cells to allow data access and authorization between clients and servers in different cells. DFS uses the Episode™ physical file system, although it can use other native file systems, such as UFS.

A standard interface is used to couple DFS with HPSS. The integrated HPSS/DFS system provides transparent, automatic archiving and caching of data between DFS and HPSS. It supports partially-resident file data in both DFS and HPSS, and allows changes made to a file or directory through DFS interfaces to be visible through HPSS interfaces and vice versa.

The standard selected to couple DFS with HPSS is The Open Group's Data Storage Management standard, XDSM. It was previously known as DMAPI and originated in the mid-nineties from the Data Management Interfaces Group (DMIG). XDSM is a low-level interface to the physical file system. It provides the Data Management application (DMAP) with the ability to store important attribute information with a file and allows for the generation of notifications to the DMAP on occurrence of various file system operations. XDSM enables the DMAP to control disk storage by allowing the DMAP to move disk-resident file data to tertiary storage systems and vice-versa.

7.1.1 Filesets

DFS supports logical collections of files called *filesets*. A fileset is a directory subtree, administered as a unit, that can be mounted in the global name space. Multiple DFS filesets may reside on an *aggregate*, which is analogous to a disk partition. Filesets may be moved between DFS aggregates, either on the same or different servers to achieve load balancing.

For the HPSS/DFS interface, two data management configuration options, archived filesets and mirrored filesets, are available which differ in the way the fileset is managed by HPSS.

7.1.1.1 Archived Filesets

HPSS is used strictly as an archive facility for DFS. Access to the name and data space is provided only through DFS interfaces. Filesets managed with this option are called archived filesets. The files in an archived fileset contain copies of file data from any DFS files that have migrated to HPSS. Path names to these HPSS files are generated by the HPSS Data Management Application (HDM) based on configuration policies. HPSS root may access objects in these filesets, but all client access is via DFS.

7.1.1.2 Mirrored Filesets

Consistency between HPSS and DFS name and data spaces is maintained. DFS data is archived to HPSS. Access to the name and data space is available through both DFS and HPSS interfaces, with updates made through the DFS interface being visible through the HPSS interface and vice versa. Filesets managed with this option are called mirrored filesets. Objects in mirrored filesets have corresponding entries in both DFS and HPSS with identical names and attributes. A user may access data through DFS, at standard DFS rates, or when high performance I/O rates are important, use the HPSS interface.

7.1.2 Architectural Overview

To interface DFS with HPSS, modifications to both DFS and HPSS were required. The architecture used to integrate HPSS with the Episode file system is shown in Figure 7-1. A brief description of the modifications and new components are discussed in the following sections.

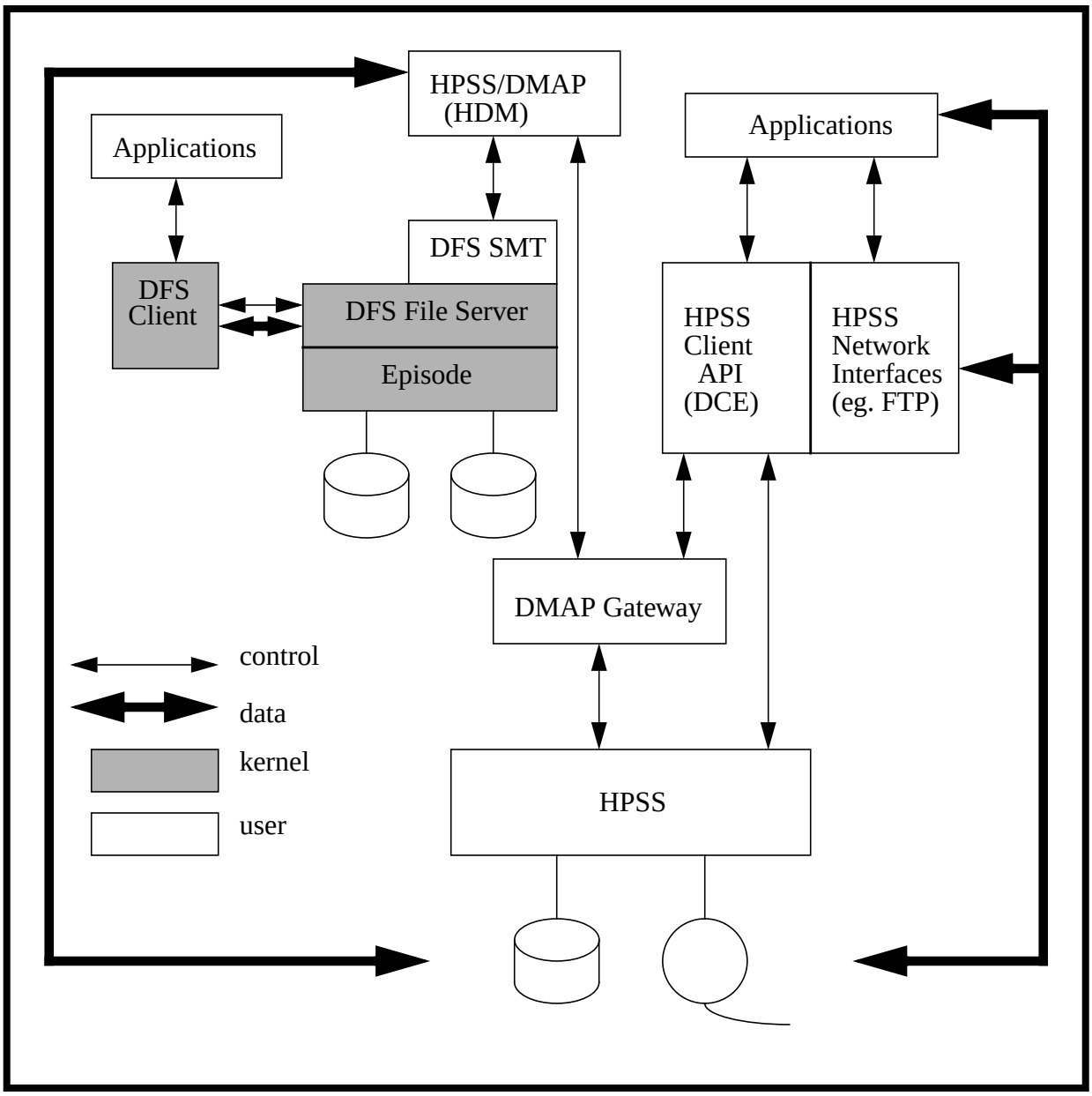


Figure 7-1 DFS / HPSS DMAPI Architecture

7.1.2.1 XDSM Implementation for DFS

An XDSM implementation for DFS was required. The XDSM implementation, developed by Transarc, is called the DFS Storage Management Toolkit (DFS SMT), and is fully compliant with the corresponding standard XDSM specification. In addition, it provides optional features: persistent opaque Data Management (DM) attributes, persistent event masks, persistent managed regions, non-blocking lock upgrades and the ability to scan for objects with a particular DM attribute.

The bulk of DFS SMT is implemented in the DFS file server, but there is also a user space shared library that implements all APIs in the XDSM specification. The kernel component maintains XDSM sessions, XDSM tokens, event queues, and the metadata which describes the events for which various file systems have registered. The kernel component is also responsible for receiving events and dispatching them to the DMAP.

The DFS File Server was augmented to fetch and store DM attributes, provide persistent managed regions and events, perform invisible I/O, purge data from files, and verify file residency. This component determines if a DMAP has registered to receive a notification for an event related to a particular operation, and then generates the event. If the event is synchronous, it causes the file system operation to wait for a response to the event before proceeding. It also provides for interlocking between DFS SMT requests and file system calls.

To support persistent DM related metadata, an extended attribute facility was provided in Episode. DM attributes, event masks, managed regions, and attribute change times (dtime values) are stored as extended attributes. These extended attributes are treated as file metadata. Changes to extended attributes are logged.

Episode was also modified to support files that become sparse by punching holes that release disk resources. With a conventional sparse file, reading from a hole returns zeroes. To assume these same semantics for a hole that exists because the DMAP migrated the data to tertiary storage would be incorrect. In this case, the DMAP must retrieve the data from tertiary storage. Hence, a facility is provided in Episode to mark file blocks as being off-line (in tertiary store) instead of as a hole. This allows the file server to handle partially resident files.

To prevent blocking file server (kernel) threads while waiting for a response to an event, a mechanism to notify the client to retry after a specified interval of time was also added to Episode.

The DFS fileset dump and restore capability was augmented to include extended attributes and migrated regions. Migrated data is not recalled when a dump is taken, producing an abbreviated dump. Checks were added to DFS and retrofitted into prior releases to prevent restoring an abbreviated dump to a file system that may not be able to interpret its contents.

The current release of Episode that supports XDSM does not support cloning. Consequently some fileset commands, such as, fts clone, fts move, etc., will not work for XDSM enabled DFS. As a result fileset backups that rely on cloning will have to be done with fts dump.

7.1.2.2 HPSS Modifications and Additions to Support DFS

Two new components, an HPSS Data Management Application (HPSS/DMAP or HDM) and a DMAP Gateway, were developed by the HPSS Collaboration.

HPSS/DMAP (HDM) Server

HDM is responsible for initiating and coordinating name space and data interactions between Episode and HPSS. It catches and processes desired name and data space events generated by Episode sent through the DFS SMT, migrates file data from Episode to HPSS, purges unneeded Episode file data after that data has been migrated to HPSS, and processes requests originating in HPSS interfaces that require either Episode name or data resources. HDM resides on the same machine where the Episode file system is running. HDM communicates with HPSS components via XDR over TCP sockets.

HDM registers to receive name and data space events originating in DFS. After catching a name space event involving a mirrored fileset, the appropriate requests are made to HPSS to keep the name spaces synchronized. However, for archived filesets, only create and destroy name space events are processed, but no HPSS resources are utilized until the file is migrated to HPSS. HDM receives data space events when a file is read, written, or truncated and the data region involved is registered with the DFS SMT. HDM is responsible for caching data to Episode that is not present; invalidating HPSS data, if necessary; and manipulating data regions to minimize the occurrence of events involving the same region. HDM can cache partial files.

HDM provides an interface for HPSS to request that an action occur in DFS to keep the HPSS and DFS name and data spaces synchronized. This mechanism is only used with mirrored filesets when an HPSS client requests to create, delete or modify a name space object. This interface is also used by HPSS to migrate or purge data from Episode disks. Before file data can be altered through HPSS interfaces, the data must first be purged from Episode disks. The capability to forward DCE credentials is provided, enabling HDM to make DFS requests on behalf of the user.

HDM migrates file data from Episode to HPSS. To free Episode disk resources, it also purges file data from Episode. Only the data that has been modified on Episode is migrated to HPSS, thus, a minor modification to a very large file will not result in re-migrating the entire file. Because policies for migrating and purging data are separately configurable, file data migrated from Episode is not automatically purged. In many cases, data for a given file will be present in both Episode and HPSS and that data can be read from either interface without any data movement between Episode and HPSS.

DMAP Gateway Server

The DMAP Gateway is a conduit and a translator between HDM and HPSS. HPSS servers use DCE/RPCs to communicate, however the DMAP Gateway encodes requests using XDR and sends these requests via sockets to HDM. In addition, it translates XDR from the HDM to DCE/TRPC/Encina calls to the appropriate HPSS servers. When a connection between the HDM and Gateway is made, mutual authentication occurs.

The DMAP Gateway keeps a record of all the DFS and HPSS filesets it manages. For scalability, multiple DMAP Gateways are supported. However, a given DMAP Gateway will only operate on the filesets it manages. At this time, the DMAP Gateway only supports filesets managed by DFS.

In addition to translating requests between HDM and HPSS servers, the DMAP Gateway keeps fileset request statistics and internal DMAP Gateway resource utilization statistics. Thus, heavily used filesets can be identified and management of these filesets can be distributed across multiple DMAP Gateways to improve performance.

7.1.2.3 HPSS/DFS Usage Guide and Limitations

This section of the document will help system administrators determine the best way to use the HPSS/DFS interface. Both DFS and HPSS have limitations that will be visible through the HPSS/DFS interface. Proper configuration of the HPSS/DFS interface will help with both DFS and HPSS performance. Many of these limitations are caused by implementation decisions made in both HPSS and DFS. Others are hardware dependent. This section will address known limitations, why they exist, when and if the limitation will change, and how to best configure your system to help avoid problems caused by the limitations.

Fileset Anode Limitations

The current implementation of Episode allows a maximum of 2GB of space per fileset for anode and file attribute storage. This space includes anodes (file headers), space allocation, ACL's, and space used to keep track of DMAPI attributes (regions and extended attributes). When not using HPSS/DFS this allows storage space for approximately 4 million files in a single fileset. This limit is greatly reduced when a fileset is backing data into HPSS.

The information stored to keep track of migration information will generally use an extra fragment of anode space. Since we suggest that the fragment size be equivalent to the block size (8K), each file migrated into HPSS will use a little more than 8K (assuming no extra ACL space). This puts an upper limit on the number of files per fileset at about 250,000.

This limit may change in the future. If the fragment size can be lowered to 1K then the number of files increases to about 1.5 million. The 1K fragment size is not currently suggested for two reasons:

The internal fragmentation is not reflected properly for the aggregate. It is quite possible for an aggregate to look like it has lots of space but the space is all tied up in fragments which can not be used for files greater than the block size. This leads to some real problems when trying to determine how much data needs to be purged to free up DFS space.

Fragments can not be purged. Only files using blocks can be purged.

Meetings with Transarc and IBM Austin have taken place to discuss the issue. The above restriction may be fixed in the future.

Migration and Purge Algorithms

Currently, the HDM will read through all the anodes in an aggregate to determine migration and purge candidates. Using tests run at customer sites and during testing cycles in Houston we have been able to determine that the HDM can read approximately 70 entries per second (this is disk hardware dependent). The desirable number of objects stored on an aggregate can be determined by using the following parameters:

The amount of time to read through all the aggregate objects.

The frequency of migration runs (purge runs when space is needed).

The amount of time it normally takes to move files from DFS to HPSS.

If the migration runs once per day, and the data can be moved in 4 hours, the current suggestion is to allow approximately 500,000 objects on an aggregate. This would mean that migration will take approximately 6 hours (7142 seconds = ~2 hours for header rumble + 4 hours for data movement). Obviously, these numbers vary dramatically and are dependent on file size and the number of files that need to be migrated. Remember, only new files and files whose data has been altered need to move data into HPSS. Files that have already had their data moved into HPSS, but have been read do have to update some DMAPI attribute information, but this can be done fairly quickly (approximately 60 per second).

Modifications for Episode XDSM(DMAPI) changes have been discussed between HPSS and Transarc. It is believed that with some changes to the Episode XDSM(DMAPI) speed to read all objects in an aggregate can be increased to approximately 400 per second. This would greatly

reduce the time it takes to determine migration and purge candidates and would raise the suggested limit above 2 million objects.

HPSS currently has no implementation plan for keeping track of migration and purge candidates other than using DMAPI extended attributes. This has been discussed and it is believed that the bookkeeping would be a significant performance problem. HPSS bookkeeping may be revisited if changes to DMAPI do not provide the desired performance gains.

Fileset Quotas

The Episode fileset quota is currently stored in a 32-bit field (actually only 2^31 – 1 can be stored). The quota is kept track in 1K allocations. So, the current maximum quota per fileset is about 2TB. The Episode quota reflects the amount of data stored in the fileset plus the amount of anode space. The quota can not be increased beyond 2TB. When the quota limit is reached the only way to store more files is to remove old files.

When the HDM purges data from Episode the quota is not affected. The quota reflects all data ever stored through the DFS interface. If a file is written to a mirrored fileset through the HPSS interfaces other than DFS, the quota is not altered (except for anode space) until the data is actually read through the DFS interface. So, it is possible to write more than 2TB to a mirrored fileset. But only if the data is written through a non-DFS interface and never read through the DFS interface (This may occur if large data dumps are made directly to HPSS, but the client wishes to use DFS to managed the name space).

There are no current plans to change the Episode fileset quota implementation.

Mirrored Fileset Recovery Speed

Mirrored fileset recovery time must be considered when configuring the system. Mirrored fileset recovery is fairly tedious and slow. The DFS fileset is recovered by using HPSS metadata, but this requires many SFS accesses.

There are three time consuming steps used to recover mirrored filesets:

Dumping the HPSS fileset information. This step proceeds at approximately 60 entries per second.

Restoring the fileset to Episode. This step proceeds at approximately 60 entries per second.

Loading Episode DMAPI handles into the HPSS metadata. This is by far the most time consuming step. This step is highly dependent on SFS speeds and during tests this step can be run at approximately 12 per second.

The above numbers can be used to approximate recovery times for mirrored filesets. The recovery time for a mirrored fileset with 100,000 objects will take approximately 3.2 hours (1666 seconds for metadata dump + 1666 seconds for Episode fileset recovery + 8333 seconds for HPSS metadata updates = 11665 seconds = ~3.2 hours).

So, it can be seen that recovery of a lost Episode fileset can take a very long time. If multiple filesets need to be recovered, many of these steps can be done in parallel, but disk and SFS limitations will quickly outweigh any amount of parallelism.

There is currently no easy fix for this problem.

HPSS/DFS Activity

The system administrator must consider what level of HPSS / DFS activity the core HPSS servers can handle. HPSS speed is limited by hardware, SFS, disk and tape speeds. If HPSS / DFS activity causes millions of HPSS file creations and file migrations into HPSS, then the system may have some real problems.

Users of HPSS / DFS filesets should be made aware of HPSS / DFS limitations and instructed in their proper usage. Though the HPSS / DFS interface allows for normal Unix type file activity, it can not be forgotten that these interfaces cause activity to the archive. The HDM is implemented to migrate individual files into HPSS, so there is a potential that each file access may incur a tape delay when reading back files. If an HPSS / DFS user stores related data into larger files, tape accesses can be decreased, and the user's performance will increase. Also, the load on HPSS will also be decreased.

Users and administrators should be educated in the use of HPSS / DFS filesets, so that they may make informed decisions on how to use HPSS / DFS filesets. Remember, a single fileset type does not optimally handle all types of file usage patterns. Normal DFS filesets run out of disk space. HPSS / DFS archived filesets incur delays for data retrieval if the data has to be staged back to Episode disk. HPSS / DFS mirrored filesets incur delays for data retrieval and also are much slower for name space updates, but they allow access to the data and name spaces through all HPSS interfaces.

Cloning, Replication, and Fileset Backup

Aggregates managed by the Episode XDSM (DMAPI) do not support cloning. Fileset replication and movement can not be done because they require cloning. DFS backups can be done more efficiently with cloning. The normal DFS fileset backup mechanisms can not be used.

HPSS / DFS filesets that are mirrored do not have to be backed up since they can be restored from HPSS metadata. The restored mirrored fileset will only be able to restore what HPSS knows about. Files deleted by mistake may not be recovered on mirrored filesets (Users should be made aware of this fact and the proper precautions taken).

HPSS / DFS archived filesets must be backed up, because not all data is migrated into HPSS and the name space information is not recoverable from HPSS metadata. The only way to back up archived filesets is with the DFS utility '*fts dump*'. The problem is that during the dump the fileset is locked for updates. Filesets that contain many objects and a lot of data may be locked from updates for a long period of time.

This problem can not be addressed by HPSS. Transarc must make any advances in this area.

7.2 DFS Configuration

For AIX systems, it is assumed that the system on which the DFS Server is running has been configured with AIX DCE 2.2 and at least PTF SET 6.

The following steps are required to set up the DFS/HPSS interface:

1. Configure DFS SMT Kernel Extensions (AIX)

2. Configure DCE DFS

3. Configure HDM

Configuring DCE DFS involves editing and creating some new AIX scripts:

/opt/dcelocal/tcl/user_cmd.tcl

/var/hpss/hdm/pre_start_dfs

/var/hpss/hdm/pre_stop_dfs

/var/hpss/hdm/post_start_dfs

/var/hpss/hdm/post_stop_dfs

or, on the Sun Solaris platform, adding the script

/etc/dce_modules/local_envf

which may contain dfs pre/post start/stop functions.

These scripts ensure that the DFS SMT Kernel extensions are loaded correctly at startup time, that the HDM is started before DFS file systems are exported, and that the system is shut down gracefully.

Configuring the HDM involves creating a number of configuration files. One set of files is required for each HDM that is to be run on a machine, and different HDMs must not use the same files. The files include a configuration file (config.dat) that determines the operating parameters of an HDM; a file system configuration file (filesys.dat) that specifies what aggregates and filesets are managed by that HDM; a gateway configuration file (gateways.dat) that tells which DMAP Gateways are permitted to talk to that HDM; a policy file (policy.dat) that defines the HDM's migration and purge policies; and a security file (security.dat) that controls cross-cell security for mirrored files managed by that HDM.

The following sections describe how to configure the SMT, DFS, and HDM in more detail.

7.2.1 Configuring DFS SMT Kernel Extensions (AIX)

The DFS SMT kernel extension software should be configured to use the shortest possible timeout parameter for **delay**. This parameter determines the interval at which the kernel backs off when HDM response is slow, which can happen if a file is being staged to Episode from HPSS and the data is not yet available on Episode. The timeout parameter is set with **-delay**. The default value should not be used, since it can cause the system to take up to 17 minutes to process a single end-user request. It is recommended that **delay** be set to 1. The parameter is expressed as an exponent that is applied to a base of 4 seconds. Hence, setting the parameter to 1 causes a 4 second delay. Configuring this parameter is described in the next section.

7.2.2 Configuring DCE DFS

7.2.2.1 Configuring DCE and DFS on AIX

To avoid data and name space inconsistencies, the HDM should be started before the aggregates it manages are exported. The safest way to ensure this is to modify the AIX startup scripts. For more information on how these scripts work, refer to the DCE 2.2 document *Quick Beginnings* in the chapter "Configuring DCE 2.2 for AIX Servers and Clients" in the section "User-Supplied Commands".

The first thing to do is to create the following new scripts:

/opt/dcelocal/tcl/user_cmd.tcl

/var/hpss/hdm/pre_start_dfs

/var/hpss/hdm/pre_stop_dfs

/var/hpss/hdm/post_stop_dfs

The new scripts take care of loading the DFS SMT Kernel extensions and starting the HDM. For purposes of discussion, the last three scripts are assumed to be in **/var/hpss/hdm**; but any suitable directory can be used as long as **user_cmd.tcl** has been set up to point to that directory.

The example scripts below assume that a site runs more than one HDM on a machine, and that these HDMs obey certain conventions. The scripts will have to be modified if different conventions are used, and can be simplified on machines that have only one HDM. The following convention makes it particularly easy to write the scripts and to use **hdm_admin** to administer the resulting system. In this convention, an HDM whose **ServerID** parameter is **<N>** works with data files in the directory named **/var/hpss/hdm/hdm<N>** and uses a shared memory key given by **3788+<N>**.

For more information on the **ServerID** and shared memory key concepts, refer to **Section 7.2.3**. For more information on hdm_admin, refer to **Appendix I.20**.

Here is a sample **user_cmd.tcl**:

```
#!/bin/ksh
set pre_start_dfs   "/var/hpss/hdm/pre_start_dfs"
set pre_start_dfs_fail_on_error $TRUE
set post_start_dfs  "/var/hpss/hdm/post_start_dfs"
set pre_stop_dfs    "/var/hpss/hdm/pre_stop_dfs"
set post_stop_dfs   "/var/hpss/hdm/post_stop_dfs"
```

AIX will run the **pre_start_dfs** Korn shell script before trying to start DFS and export DFS files. The script ensures that the SMT kernel extensions are loaded and that the HDMs are all running. If there is any problem doing that, AIX will ensure that DFS is not started, giving the system administrator a chance to fix the problem.

If this script needs to be modified, it is important to make sure that incidental messages that would normally be written to stderr get rerouted to stdout or **/dev/null**. Otherwise the TCL procedures that call **pre_start_dfs** will assume the script has had an error, even if the script eventually calls exit 0. It is also important to redirect all output from the **hpss_hdm** command; otherwise TCL will wait for the HDM to stop before going ahead with the startup, with the result that DFS will never start.

In this script, be sure to set the delay parameter on **cfgdmepi** to 1. This parameter controls the maximum delay time between client retries after an operation fails. In general, a delay parameter of N causes a maximum delay interval of 4 raised to the Nth power. If N is zero, retries will be done once a second which may cause the system to thrash. If N is omitted, the default value of 5 will be used, which can result in delay times as long 1024 seconds (roughly 17 minutes).

Here is a sample for **pre_start_dfs**:

```
#!/bin/ksh
export HPSS_PATH_BIN=/usr/lpp/hpss/bin
echo "   loading dfscore, dfssvr, and dcelfs"
/usr/sbin/cfgdfs -a /usr/lib/drivers/dfscore.ext
/usr/sbin/cfgdfs -a /usr/lib/drivers/dfssvr.ext
/usr/sbin/cfgdfs -a /usr/lib/drivers/dcelfs.ext
echo "   configuring dmepi"
/usr/sbin/cfgdmepi -delay 1 -a /usr/lib/drivers/dmlfs.ext
if [ $? != 0 ]; then
  exit 1
fi
# Start the servers (two of them in this example):
for id in 0 1; do
  key=`expr 3788 + $id`
  var=/var/hpss/hdm/hdm$id
  $HPSS_PATH_BIN/hdm_admin -k $key -s $id -v $var ps >/dev/null 2>&1
  if [ $? != 0 ]; then
    echo "   starting hdm$id"
    rm -f $var/hdm.out
    $HPSS_PATH_BIN/hpss_hdm $var/config.dat $id > $var/hdm.out 2>&1
    status=$?
    if [ $status != 0 ]; then
      echo "   could not start hdm$id, status = $status"
      exit $status
    fi
  else
    echo "   hdm$id is already running"
  fi
done
echo "   all hdm servers are running"
exit 0
```

AIX will run the **post_start_dfs** Korn shell script after the DFS aggregates have been exported. Normally there is nothing that needs to be done at this time, so the script can be left blank. Alternatively, the file can be eliminated altogether, in which case **user_cmd.tcl** should be edited to remove the line that refers to **post_start_dfs**.

Here is an example of what the **post_start_dfs** script might look like:

```
#!/bin/ksh
echo "   startup complete"
# This is a good place to start a diagnostic program or
# set dfstrace options.
exit 0
```

AIX runs the **pre_stop_dfs** script just before shutting down DFS. As part of the shutdown procedure, AIX will try to unexport or "detach" all DFS aggregates. This ensures that they are left in a consistent state and don't need to be salvaged before they can be used again. In order to detach an aggregate, it is important that the HDMs be running first, so this script tends to that. The script also ensures that no filesets are locally mounted, which would prevent the aggregates from being detached. As with the other scripts, incidental messages to stderr should be rerouted to stdout.

Here is sample for **pre_stop_dfs**:

```
#!/bin/ksh
export HPSS_PATH_BIN=/usr/lpp/hpss/bin
for id in 0 1; do
 key=`expr 3788 + $id`
 var=/var/hpss/hdm/hdm$id
 $HPSS_PATH_BIN/hdm_admin -k $key -s $id -v $var ps >/dev/null 2>&1
 if [ $? != 0 ]; then
  echo "   starting hdm$id for cleanup"
  $HPSS_PATH_BIN/hpss_hdm $var/config.dat $id > $var/hdm.out 2>&1
 fi
 echo "   unmounting locally mounted filesets for hdm$id"
 $HPSS_PATH_BIN/hdm_admin -k $key -s $id -v $var tcp disable >/dev/null 2>&1
done
exit 0
```

AIX runs the **post_stop_dfs** script once it has completed detaching the DFS aggregates and shutting down DFS. The script stops any HDMs that are still running.

Here is a sample for **post_stop_dfs**:

```
#!/bin/ksh
export HPSS_PATH_BIN=/usr/lpp/hpss/bin
# Stop any hdm's that are still running
for id in 0 1; do
 key=`expr 3788 + $id`
 var=/var/hpss/hdm/hdm$id
 $HPSS_PATH_BIN/hdm_admin -k $key -s $id -v $var ps >/dev/null 2>&1
 if [ $? = 0 ]; then
  echo "   stopping hdm$id"
  $HPSS_PATH_BIN/hdm_admin -k $key -s $id -v $var -y stop >/dev/null 2>&1
 fi
done
```

```
exit 0
```

To start and stop DFS, be sure to use the commands start.dfs and stop.dfs. In particular, do not stop DFS using dfs.clean, since it does not handle HDM correctly.

7.2.2.2 Configuring DCE and DFS on Sun Solaris

To avoid data and name space inconsistencies in the Solaris environment, the HDM should be started before the aggregates it manages are exported. An automatic way to do this is to create the script "local_envf" and place it in the directory "/etc/dce_modules". This script may contain pre-start, post-start,pre-stop and post-stop functions. When the dfs start or stop script is executed by the system at startup or shutdown, any pre/post start/stop functions in local_envf will be executed at the appropriate phase.

Here is a sample **local_envf** file:

```
#
# dce/dfs_pre/post_execute functions for the Sun Solaris Platform.
#

#  Processed prior to DCE Initiation
dce_pre_execute()
{
   return 0;
}

#  Processed after DCE Initiation
dce_post_execute()
{
   return 0;
}

#  Processed prior to DFS Initiation
dfs_pre_execute()
{
  HPSS_PATH_BIN=/usr/lpp/hpss/bin

# Currently configured for 1 HDM
 for id in 1
 do
  var=/var/hpss/hdm/hdm$id

  case $1 in
#
#   pre START dfs execution
#
     "dfs_start")
          $HPSS_PATH_BIN/hdm_admin -s $id  ps >/dev/null 2>&1
          if [ $? != 0 ]; then
            echo "   starting hdm$id"
            $HPSS_PATH_BIN/hpss_hdm -f $var/config.dat $id >> $var/hdm.out 2>&1
            sleep 5
```

```
            echo HDM Started.
            status=$?
            if [ $status != 0 ]; then
             echo "   could not start hdm$id, status = $status"
             exit $status
            fi
           else
             echo "   hdm$id is already running"
           fi

      ;;
#
#   pre STOP dfs execution
#
"dfs_stop")
         $HPSS_PATH_BIN/hdm_admin  -s $id  ps >/dev/null 2>&1
         if [ $? != 0 ]; then
           echo "   starting hdm$id for cleanup"
           $HPSS_PATH_BIN/hpss_hdm $var/config.dat $id > $var/hdm.out 2>&1
         fi
         echo "   unmounting locally mounted filesets for hdm$id"
    $HPSS_PATH_BIN/hdm_admin -s $id -v $var tcp disable >/dev/null 2>&1
      ;;

     *)
       exit 0
      ;;
 esac
 done

   return 0
}

# Processed after DFS Initiation
dfs_post_execute()
{
 HPSS_PATH_BIN=/usr/lpp/hpss/bin

 for id in  1
 do
  var=/var/hpss/hdm/hdm$id

 case $1 in
#
#   post START dfs execution
#
    "dfs_start")
      ;;
#
#   post STOP dfs execution
#
    "dfs_stop")
         $HPSS_PATH_BIN/hdm_admin -s $id  ps >/dev/null 2>&1
         if [ $? = 0 ]; then
```

```
            echo "   stopping hdm$id"
            $HPSS_PATH_BIN/hdm_admin -s $id -v $var -y stop >/dev/null 2>&1
           fi
       ;;
      *)
        exit 0
       ;;
  esac
 done
 exit 0;
}
```

7.2.3 Configuring HDM Server

The HDM server is configured with five configuration files, usually kept in **/var/hpss/hdm/<hdm-id>.** Although any directory can be used, all five files must be kept together. By convention, the event and message logs are also kept in the same directory. The names of the configuration files are:

config.dat	basic configuration file
filesys.dat	description of file systems and filesets
gateways.dat	gateways permitted to access HDM
policy.dat	migration and purge policies
security.dat	configuration for Security Server

Four of the files (**config.dat**, **gateways.dat**, **policy.dat**, and **security.dat**) may be created using a text editor. The format of these files is described below. The files must be present before HDM can be started. If any one of these files is edited, HDM must be restarted before the change will take effect.

The fifth file (**filesys.dat**) is automatically updated by HDM as new aggregates and filesets are created. Therefore this file should not ordinary be edited by the administrator. HDM cannot be started if this file is missing or does not contain correct information. Before starting HDM for the first time, a special version of **filesys.dat** must be created so that HDM will recognize that the file is correct.

It is possible to run multiple instances of HDM on one machine. For example, one HDM might be used to handle mirrored aggregates, while another might be used to handle archived aggregates. To set up a system in this manner, be sure to keep the associated configuration and log files in separate directories. For example, **/var/hpss/hdm/hdm1** and **/var/hpss/hdm/hdm2**.

Since logs necessary for HDM are heavily used and vital, it is recommended that configuration directories and the directories where the logs are stored be placed on AIX file systems that are mirrored and have low latencies.

7.2.3.1 config.dat File

The basic configuration file, **config.dat**, is a text file that defines the configuration of an HDM server. It is recommended that the file be kept in the directory **/var/hpss/hdm/<hdm-id>**.

The file consists of a series of lines, where each line defines one parameter. The first field on the line specifies the name of the parameter and the second field is the value for that parameter. The first line in the file must define **ServerID**. The following lines define the rest of the parameters. **ServerID** must begin in column one, while the other parameters must be indented by at least one tab character. The file may contain comments that start with a `#' character and continue until the end of the line.

ServerID is an arbitrary number used to distinguish different HDM servers defined in the configuration file. Once **ServerID** has been established, it should not be changed because it is used during event recovery whenever HDM is restarted. A good choice for **ServerID** is 1, since this is the default used by **hdm_admin**. HDM can be started with the command:

hdm_hdm /var/hpss/hdm/<hdm-id>/config.dat <ServerID>

where **ServerID** is a number identifying which part of the configuration file HDM should read.

Following is an extract from a typical configuration file:

An example HDM configuration file

```
ServerID        1
                DescNameProduction configuration
                RegisterBitMap0.0 #A comment
                [...]
```

For a complete example of a configuration file, refer to the template file, **/usr/lpp/hpss/config/templates/hdm_config.dat.template**.

The following paragraphs discuss each parameter found in the file. Except as noted, each parameter must be specified. HDM will **not** start if a mandatory parameter is omitted. The configuration parameters can be specified in any order. The keywords must be spelled correctly, using the specified upper and lower case letters. For example, DescName, not descName or descname.

AclLogName specifies the name of the file used for the ACL log. Typically, this will be **/var/hpss/hdm/<hdm-id>/hdm_acl_log**. This file contains a record of pending ACL change requests made through the HPSS interface. The file must exist before HDM is started, but can be empty. The size of the file is unbounded, but typically will be small. This log is very important and should be stored on a reliable disk.

DestroyLogName specifies the name of the file used for the destroy log. Typically, this will be **/var/hpss/hdm/<hdm-id>/hdm_destroy_log**. This file contains a record of all files on mirrored filesets that need to be destroyed. The file must exist before HDM is started, but can be empty. If **DestroyLogSize** is changed, HDM automatically adjusts the size of the file. This log is very important and should be stored on a reliable disk.

DestroyLogSize specifies the total number of files in mirrored filesets that are waiting to be destroyed, thus determining the size of the log file. Because of limitations in DFS SMT, it is not possible to destroy HPSS files immediately when a destroy event is received. For example, after a recursive remove, the number of pending file destroys can become quite large. Once the log is full,

attempts to remove file names and delete the file data will be delayed until the destroy process clears the log. Another consideration occurs when a user's program creates a file, opens it, and then unlinks it, expecting that when the program exits, the file will go away. HDM must keep an entry for this file in the destroy log until the user's process exits. This ties up destroy log entries. For these reasons, a fairly large **DestroyLogSize** should be used. On the other hand, avoid using an excessively large value because that causes more overhead when deleting files. Also, if the system has a fairly heavily loaded archived file system, it may take a while for HDM to get around to destroying files on the mirrored file systems. Using a smaller **DestroyLogSize** tends to fix this problem. A good starting value for **DestroyLogSize** is 200. The name of this log file is specified by **DestroyLogName**.

If necessary, **DestroyLogSize** can be decreased by editing **config.dat** and restarting HDM. However, this only works if the new value is large enough to accommodate all of the outstanding entries in the old destroy log.

EventQueueSize specifies the maximum number of events HDM can queue for processing. Ideally, the number should be the sum of **NumDataProcesses**, **NumNamespProcesses**, and **NumAdminProcesses**, but it may be a good idea to use a slightly larger number. If the value is too small, some subprocesses could lie idle. For example, if the queue happens to fill with data events, then name space processes will lie idle until some of the data events have finished processing. A value in the range 20-50 is a good starting point.

ExecPath specifies the path name of the directory where HDM executables are located. Typically, this will be **/usr/lpp/hpss/bin**.

Flags defines special flags that control the operation of HDM. The parameter is specified as a series of keywords, separated by white space. Currently there are two keywords defined: "**permissiveMount**" and "**stdout**".

When a DFS aggregate is exported and "**permissiveMount**" is specified, HDM will check its tables to see if it manages that aggregate. If not, it assumes that some other HDM will manage the aggregate and so relays the event forward. If in fact no other HDMs are prepared to manage the aggregate, it will get mounted but will not be kept synchronized with HPSS. This flag is required when several HDM servers are run on one machine, but leads to a possibility that an aggregate will get overlooked and not kept synchronized.

On the other hand, if "**permissiveMount**" is not specified, then HDM will abort mount events for aggregates that it does not manage. While this is safer, it cannot be used on a machine where multiple copies of HDM are running.

The "**permissiveMount**" flag must be specified if several HDM servers are to be run on one machine, and should never be specified when only one is to be run.

If the "**stdout**" keyword is present on the Flags line, HDM will write all messages to standard output. While this option can help with debugging, its use is generally discouraged.

HPSSDMAPHostName specifies the host name for the machine where the HDM will be run. This should be a fully qualified host name. For example, **tardis.ca.sandia.gov**. Be sure to use the host name where HDM will be run, not the host name where the DMAP Gateway will be run.

HPSSDMAPTCPPort specifies the TCP port used by the HDM to listen for requests from the DMAP Gateway. Conventionally, this will be 6002. Check **/etc/services** to ensure that this port is not already being used by another program. Do not confuse this port number with the port used by the

DMAP Gateway to receive requests from HDM, whose value is conventionally, 7001. Make sure both port numbers mentioned here are consistent between HDM and DMAP Gateway configurations.

When multiple HDM servers are configured , each HDM must have a unique TCP port.

LogRecordMask specifies the type of messages recorded in the message log. A series of keywords, separated by white space, are used to define the messages that are to be recorded. For example:

LogRecordMask alarm event trace debug

Legal keywords and their meanings are:

accounting	accounting information; not currently used
alarm	error conditions of interest to the administrator
debug	low level error message for troubleshooting
event	informational messages (e. g., system starting)
request	request-specific messages; not currently used
security	security related events; not currently used
status	status messages; not currently used
trace	program flow and other low level messages

In normal operation, only alarm and event messages need to be enabled. Trace and debug messages should be enabled when it is necessary to track down the root cause of a problem. Logging too many different types of messages will impact HDM performance.

MainLogName specifies the name of the file used for the main event log. Typically, this will be **/var/hpss/hdm/<hdm-id>/hdm_main_log**. This file contains a record of all name space events for which processing has not yet completed. The file must exist before the HDM is started, but can be empty. If **MainLogSize** is changed, HDM automatically adjusts the size of the file as needed. This log is very important and should be stored on a reliable disk. Also, since the file is frequently updated by HDM, it should be stored on a low latency disk.

MainLogSize specifies the total number of name space events that can be handled concurrently. While an event is being processed, information about the event is stored in a log file. When HDM restarts after a crash, it reads the log to determine what must be done to synchronize the DFS and HPSS name spaces. If too small a value is chosen for the log size, HDM may occasionally have to wait for a log entry to become available before it can process a name space event. There is no reason to select a value much larger than **NumNamespProcesses**. A value between 10-25 is a good starting point. If necessary, the value of this parameter can be decreased by editing **config.dat** and restarting HDM. However, this only works if the new value of **MainLogSize** is large enough to accommodate all of the outstanding events in the old event log. The name of this log file is specified by **MainLogName**.

If both the event log and event queue become full, event processing can not be done, and HDM will return an error to the DFS SMT.

MaxFilesets specifies the maximum number of filesets HDM can support. The number must be at least as large as the number of filesets on the **dmlfs** aggregates at the site. A larger value can be used if adding new filesets is anticipated. However, using a value that is too large adds overhead to event processing.

MaxFileSystems specifies the maximum number of DFS aggregates (or loosely speaking, "file systems") HDM can support. This number must be at least as large as the number of **dmlfs** aggregates on the DFS File Server. A larger value can be used if adding new aggregates is anticipated. If the value must be changed, HDM must be restarted before the change takes effect.

MaxMsgFileSize specifies the maximum size in bytes of the message log files. When writing to a log file will exceed this size, the message logger automatically switches to the other log file. If the value for **MaxMsgFileSize** is too small, potentially useful information will be lost as older messages are overwritten with new ones. A good starting value is 5000000.

MaxPolicies specifies the maximum number of migration and purge policies that can be handled by HDM. The value should be large enough to accommodate all of the purge and migration policies defined in **policy.dat**. (There is no need to make this value larger than the sum of these policies.)

MaxStages specifies the maximum number of data event processes that can concurrently stage files from HPSS to Episode. When this limit is reached, further transfers from HPSS are deferred until one of the stages completes. This value must be less than **NumDataProcesses**. A value in the range 1-3 is a good starting point.

MaxTcpConnects specifics the maximum number of simultaneous requests to mirrored filesets managed by this HDM allowed through the HPSS interface. If the value for this parameter is too small, an HPSS user request may occasionally fail. A good starting value is 32.

MinArchiveMigrationSize defines the minimum file size that will be migrated into an archived fileset. If the migration process is taking too much CPU time, use this parameter to prevent small files from being migrated.

MsgFileDirectory specifies the path name of the directory where HDM error message files are located. Typically, this will be **/var/hpss/hdm/<hdm-id>**. The log files will be named **hdm_message.0**, **hdm_message.1**, etc. The number of these files is controlled by **NumMessageFiles**. If **MsgFileDirectory** is left blank, the HDM will not write messages to the message files. In this situation, the **stdout** flag should be defined so that messages appear somewhere. But remember that use of the **stdout** flag is discouraged, so it is better to provide a value for **MsgFileDirectory**.

NumAdminProcesses specifies the number of HDM subprocesses that will be assigned to handle administration events, such as mounting and unmounting file systems. The value, 1, is a good starting point.

NumDataProcesses specifies the number of HDM subprocesses that will be assigned to handle events involving data requests, mainly read and write events. The value selected must be large enough to allow a reasonable amount of I/O overlap. Remember that, at any given time, some number of these processes may be busy staging data from HPSS (**MaxStages**). The value, 8, is a good starting point.

NumMessageFiles specifies the number of files that will be used for recording messages generated by the HDM. These files will be stored in the directory given by **MsgFileDirectory**, and will be

named **hdm_message.0**, **hdm_message.1**, etc. This parameter is optional; if it is omitted, two message files will be used

NumNamespProcesses specifies the number of HDM subprocesses that will be assigned to handle events involving changes to the name space. These events include file creates, deletes, renames, and permission changes. The value, 8, is a good starting point.

RegisterBitmap is a 64-bit number, expressed in the form "<high>.<low>", where <high> and <low> are integers providing the high and low 32 bits of the number. This field is not currently used, and should be set to 0.0.

SharedMemoryKey is an integer that defines a key used by HDM subprocesses to attach to the shared memory segment set up by the main process. The key is also used to identify the HDM semaphore set. If a value of zero is listed in the file, a default key with the value, 3789, is substituted. If only one HDM is running on a machine, use zero for this parameter; this ensures that HDM and **hdm_admin** use the same key. The default setting can be changed with the environment variable, HPSS_HDM_SHMEM_KEY.

When multiple HDM servers are to be run on the same machine, each HDM must have a unique SharedMemoryKey. HDM servers cannot share memory or logs without serious problems.

ZapLogName specifies the name of the file used for the zap log. Typically, this will be **/var/hpss/hdm/<hdm-id>/hdm_zap_log**. This file contains a record of the archived fileset files that need to be destroyed. The file must exist before HDM is started, but can be empty. The size of the file is unbounded, but typically will be small. This log is very important and should be stored on a reliable disk.

7.2.3.2 filesys.dat File

The filesystem configuration file, filesys.dat, is a text file that defines each aggregate and fileset HDM manages. If HDM receives an event related to an aggregate or fileset not listed in the file, the event will be aborted and the end user will receive an error. The file must be located in the same directory as config.dat, typically /var/hpss/hdm/<hdm-id>.

This file is maintained by HDM and should not need to be edited by an administrator. When HDM modifies the file, a copy of the original file is saved using a name like filesys.YYMMDDhhmmss, where YYMMDDhhmmss gives the date and time that the original file was saved (not the date when the original file was first created.) If many changes are made in a short period of time, HDM will not save a copy of each file, but rather will save the most recent file in filesys.bak, overwriting the previous backups in the process. For example, if the system administrator creates a large number of filesets at 10:00 AM on 4/1/99, when the operation is complete, three files will be created: a file named filesys.990401100000 containing a copy of filesys.dat that was in effect before the new filesets were created; filesys.bak describing all but the last fileset; and filesys.dat describing all of the filesets.

When HDM is first started, filesys.dat must be present. It is not possible to start HDM if filesys.dat is empty or does not have the right format. To make a suitable file for starting HDM the first time, use a text editor to create a file whose first and last lines are identical and contain the string "# HDM filesys.dat version 1". Alternatively, create the file by copying a template file using commands such as these:

```
cd /usr/lpp/hpss/config/templates
```

cp hdm_filesys.dat.template /var/hpss/hdm/<hdm-id>/filesys.dat

where **<hdm-id>** is the the directory where filesys.dat is to placed.

Although HDM is responsible for maintaining **filesys.dat**, it may occasionally be necessary for the administrator to edit the file manually to fix problems. This section describes the format of the file.

It is not safe to edit the file any time XDSM managed filesets or aggregates are being created or deleted. Once the file has been edited, HDM must be restarted for the changes to take effect. Remember, until HDM is restarted, it can overwrite the edits.

Comments may appear in this file, but they will not be preserved when HDM updates the file. Comments begin with a '#' character and continue to the end of the line. HDM requires that the first and last lines in the file contain the comment "# HDM filesys.dat version 1". If these lines are absent or incorrect, HDM assumes the file is corrupt; so be careful not to edit these lines. With the exception of these two lines, HDM ignores comments.

This file consists of a number of lines that describe the aggregates and the filesets that reside on that aggregate. Each aggregate is described by a line that begins in the first column. After the line for each aggregate are the lines that describe the filesets that reside on that aggregate. Fileset lines begin with a TAB character. Any line that begins in column one is treated as the definition for an aggregate.

An aggregate configuration line contains the following information:

path media fsid option migratePolicy purgePolicy stageType

and the format for a fileset configuration line is:

<TAB> ftname global local ftid gateway port

The following is an example **filesys.dat**:

```
######################################################################
# HDM filesys.dat version 1
# Sample filesys.dat
/opt/dcelocal/var/dfs/aggrs/bkup1 /dev/bkup1 2 archive/delete wait run partial
        new.bkup ? ? 0.292 ? 0 # We haven't specified this yet
        hpss.bkup NO_MOUNT_POINT NO_MOUNT_POINT 0.232 tardis
7001
/opt/dcelocal/var/dfs/aggrs/mirror1 /dev/mirror1 4 mirrored run wait partial
        new.mirror ? ? 0.295 ? 0
        hpss.mirror /:/hpss/mirror /var/hpss/hdm/hdm1/aggr/mirror1 /
hpss.mirror 0.361 tardis 7001
# HDM filesys.dat version 1
######################################################################
```

Following is a description of the parameters for an aggregate:

Path specifies the path name where DFS mounts the aggregate. Typically, this is **/opt/dcelocal/var/dfs/aggrs/aggrname**, where **aggrname** is the name of the aggregate.

Media specifies the device file for the aggregate. Typically, this is **/dev/aggrname**, where **aggrname** is the name of the aggregate.

Fsid specifies the file system id for this aggregate. The value is defined by **dfstab**.

Option specifies how the filesets on the aggregate will be managed by HPSS. The parameter may be either **archive/delete**, **archive/rename**, or **mirror**. If **mirror** is selected, the name and data space will be mirrored by HPSS, and the end user can access the name and data space from either DFS or HPSS. Otherwise, only files are archived by HPSS and the files can only be accessed from DFS. If **archive/delete** is selected, any files deleted from DFS will also be deleted from HPSS. If **archive/rename** is selected, any file deleted from DFS is not deleted from HPSS, but renamed instead. This allows for the possibility of restoring a file later, if it was accidentally deleted.

MigratePolicy and **PurgePolicy** specify the name of the policies that will be used to migrate and purge files on this aggregate. The name of the migrate policy must appear in the **MigratePolicy** section of **policy.dat**, and the name for the purge policy must appear in the **PurgePolicy** section of the file. In the example above, the names of the policies are **wait** and **run**, but these names have no special meaning to HDM. Section 7.2.3.4 discusses **policy.dat** in detail.

StageType specifies the type of staging HDM will use when it is necessary to stage a file from HPSS to DFS. Legal choices are **whole** and **partial**. If **whole** is specified, the whole file will be staged; otherwise only that part of the file necessary to satisfy a request will be staged. With **partial**, the amount of data staged will be at least the size of the data access, or if the data access is small and the file is large, a 16MB chunk of the file surrounding the data being accessed will be staged.

Following is a description of the parameters for a fileset:

Ftname specifies the name of the fileset. The name should be the same as the name for the DFS fileset, which is also the name of the HPSS fileset.

Global specifies the global mount point for the fileset. This name will be a DFS style path name. For example, /:/hpss/mirror.

Local specifies the local mount point for the fileset. This name will be a UNIX style path name. Typically, the mount point will be in the directory, /**var/hpss/hdm/<hdm-id>/aggr/<aggr name>**, where **aggr name** will be the same as the fileset name. For example, hpss.mirror.

Ftid specifies the fileset Id. The parameter is specified in the form <high>.<low>, where <high> and <low> are numbers representing the high and low 32 bits for the fileset Id. This Id should be the same as the DFS fileset Id, which is also the fileset Id of the HPSS fileset.

If a fileset uses one of the archive options, the global and local mount points will typically not be set, and filesys.dat will show NO_MOUNT_POINT for these parameters. If a mirrored fileset will only be accessed by users local to the cell, the global mount point can also be specified as NO_MOUNT_POINT.

When a fileset is only partially configured, the global and local mount points will be shown as `?'. While this condition exists, DFS users cannot access the fileset. Typically, this happens only for a short period of time while the administrator is setting up the HPSS fileset. To complete the configuration, an administrator will use SSM to create the HPSS fileset.

An administrator must not edit filesys.dat to "fix" the "?"s! This data will be assigned when the HPSS fileset is created.

Gateway specifies the fully qualified name of the host where the DMAP Gateway that will manage this fileset runs. To keep the example above short, **Gateway** is shown as tardis, but in practice, the name should be tardis.ca.sandia.gov. If the fileset is partially configured, the host name will be

shown as `?'. That prevents end users from accessing the DFS fileset. To complete the configuration, an administrator will use SSM to create the HPSS fileset.

Port specifies the TCP port the DMAP Gateway uses to listen for requests from HDM. Conventionally, this is 7001. Do not confuse this port number with the port number used by HDM to listen for requests from the DMAP Gateway (6002). Until the fileset is fully configured, **Port** will be shown as zero.

When multiple HDM Servers and DMAP Gateways are running, they must use different TCP ports.

7.2.3.3 gateways.dat File

The gateway configuration file, **gateways.dat**, is a text file identifying DMAP gateways that will communicate with HDM. The file must be located in the same directory as **config.dat**, typically **/var/hpss/hdm/<hdm-id>**.

The file consists of a number of entries, each containing a host name, port, and encryption key. The host name is the name of the machine where a gateway runs or the name of a host where a system administrator will run sensitive commands such as **create_fsys**. The host name should be a fully qualified name, which includes the domain name. However, two entries may be necessary in some cases, one using the fully qualified name and one using the abbreviated name. Be sure to make entries in this file for every DMAP Gateway that HDM uses. For security reasons, do not name hosts that do not communicate with HDM.

Be sure to use the DMAP Gateway port (typically 7001) and not HDM port (typically 6002). If the machine is an administrative machine, use 0 for the port.

The encryption key is used to secure communications between HDM and DMAP Gateway. The value is expressed as a 16 digit hexadecimal number. The number must agree with the number entered on the DMAP Gateway's server specific configuration screen. The **gateways.dat** file should be protected to prevent unauthorized users from discovering the key.

For backward compatibility, the port and encryption key fields are optional. If the encryption key is missing, HDM will use an encryption key of zero. If the port is missing, a value of zero will be used. In this case, the named machine will not be able to act as a gateway, but will still be able to act as an administrative machine.

The following is a sample **gateways.dat** file:

```
######################################################################
# gateways.dat: sample gateway configuration file
#
# Normal entry has full name
tardis.ca.sandia.gov    7001  0123456789abcdef
# Short name may be needed
tardis                  7001  0123456789abcdef
# An administrative machine
admin.daleks.com        0     fedcba9876543210
# Port = 0; key = 0
k9.master.com
######################################################################
```

7.2.3.4 policy.dat File

The policy configuration file, **policy.dat**, is a text file that describes the parameters used to control the migration and purge processes. The file must be located in the same directory as **config.dat**, typically, **/var/hpss/hdm/<hdm-id>**.

The file consists of a number of sections, where each section defines a migration or purge policy. Each section begins with a line identifying the type of policy being defined (a migration or purge policy), and giving it a name. Comments can appear in the file, starting with a `#' character and continuing to the end of the line. Following is an example policy file that will be used in the rest of the discussion:

```
#####################################################
# policy.dat: sample migration and purge policy definition file

# Migration policies

MigratePolicy wait
        MigrationDelayTime0
        LastAccessTimeBeforeMigration3000

MigratePolicy run
        MigrationDelayTime86400
        LastAccessTimeBeforeMigration3000

# Purge policies

PurgePolicy   wait
        PurgeDelayTime0
        LastAccessTimeBeforePurge4000
        UpperBound80
        LowerBound60

PurgePolicy   run
        PurgeDelayTime300
        LastAccessTimeBeforePurge4000
        UpperBound80
        LowerBound60
#####################################################
```

A migration policy is specified by a line that begins with **MigratePolicy**, and a purge policy is specified by a line that begins with **PurgePolicy**. The keyword is followed by the name of the policy being defined, such as, **wait** and **run**, in the example. The choice of policy names has no significance to HDM. If desired, the same name could be used to describe a migrate and a purge policy; HDM does not assume these policies are related. The configuration parameters that define a policy immediately follow the line that names the policy. The parameters can be in any order, as long as they are all provided. Each parameter line contains a keyword, preceded by a TAB character.

After the **MigratePolicy** line, there must be two lines defining the following parameters:

LastAccessTimeBeforeMigration specifies the number of seconds that must elapse after a file is accessed before the file can become eligible for migration.

MigrationDelayTime specifies the time, in seconds, that the migration process waits between passes in which it looks for files to migrate. If the time is set to zero, HDM waits an infinite amount of time, meaning the migration process waits for a signal before looking for files to migrate. **hdm_admin** can be used to send the signal.

After the **PurgePolicy** line, there must be four lines defining the following parameters:

LastAccessTimeBeforePurge specifies the number of seconds that must elapse after a file is accessed before the file can become eligible for purging.

PurgeDelayTime is the time, in seconds, that the purge process waits between passes in which it looks for files to purge. If this time is set to zero, HDM waits an infinite amount of time, meaning the purge process waits for a signal before looking for files to purge. **hdm_admin** can be used to send the signal.

UpperBound and **LowerBound** are integers between 0 to 100, representing percentages. The purge process initiates a purge cycle when the percentage of space used in the aggregate exceeds **UpperBound**, and stops the cycle when the percentage drops below **LowerBound**. Needless to say, **LowerBound** should be less than **UpperBound**.

The choice of values for policy parameters is site specific. Some sites may want an administrator to determine when to initiate migration and purge cycles. In this case, the infinite wait policies mentioned above would be appropriate. Other sites may want the migration and purge cycles to run automatically, and for those sites, the run policies, also described above, would be better. In the second **MigratePolicy** in the above example, the migration process is set to run once every 24 hours, which may work well for a lightly loaded site. For other sites, it may be necessary to run the migration process more frequently to keep up with the load. The purge process should be run often enough to ensure that free space on the system is available. In the example above, **UpperBound** and **LowerBound** were chosen to keep between 60% and 80% of DFS on-line. Using too low of a value for **LowerBound** could lead to thrashing. Using too high of a value for **UpperBound** could cause end users to wait whenever the system needs to free space.

7.2.3.5 security.dat File

The security configuration file, security.dat, is a text file used by the DFS Security Server component of HDM for cross cell activity through HPSS interfaces on mirrored filesets. The file must be located in the same directory as config.dat, typically, /var/hpss/hdm/<hdm-id>.

The format of this file is similar to config.dat. Like, config.dat, the first line in security.dat contains the keyword ServerID, starting in column one, and provides its value. The lines for the remaining parameters immediately follow, and begin with a TAB character. All of the parameters must be defined. Following is an extract from a sample security.dat file:

```
#####################################################
# security.dat: sample security configuration file
ServerID        1
                ServerName      /.:/hpss/hdm
                Principal       hpss_hdm
                KeyTabFil       /krb5/hpss.keytabs
                RecoveryFile    /var/hpss/hdm/hdm1/pag.dat
                ObjectID        7622b5a2-226e-11d2-9cdd-08005a4726ef
```

```
##################################################
```

ServerID is an arbitrary number used to distinguish different HDM servers defined in the security configuration file. Once **ServerID** has been established, it should not be changed because it is used during event recovery whenever HDM is restarted. Both **config.dat** and s**ecurity.dat** must contain a section for each **ServerID** that is defined. (Recall that this is one of the parameters entered on the **hdm_admin** execute line.)

KeyTabFile specifies the name of a UNIX file containing a copy of the DCE key for the HDM Security Server component. The file must exist, and must contain an entry for the given **Principal**.

ObjectID specifies the DCE object UUID for an HDM. ObjectID is used by the endpoint mapper to distinguish between different instantiations of HDM servers, so a unique value must be used. **uuidgen** can be used to generate a unique UUID.

Principal specifies the name of the DCE principal that the HDM Security Server component will use.

RecoveryFile specifies the name of a UNIX file containing a copy of the Process Activation Group (PAG) records that the HDM Security Server maintains. The file must exist, but can be empty.

ServerName specifies the name of a CDS entry HDM uses to register its bindings.

7.3 Managing HDM

HDM does not run on the machines where the HPSS are run, so it is not managed by SSM. Therefore, the procedures for managing it are quite different from the procedures for managing other HPSS servers. Administration of HDM is done with command line utilities such as, **hdm_admin**. Many of the utilities require an administrator to log in as root, but do not require the administrator to log into a DCE account.

The **hdm_admin** utility accesses HDM shared memory. Shared memory can be inspected even after HDM has stopped. This is a mixed blessing. In most cases, an administrator can figure out what HDM was doing the last time it was running. However, it can also give an administrator a false sense of security if the administrator believes the state represents the current condition of HDM, when, in fact, HDM is not running at all. **hdm_admin** continually refreshes its view of shared memory, so there is no danger of the utility reporting data for an old instance of HDM after a new instance has been started.

7.3.1 Configuring HDM

Before HDM can be run, the configuration files must be created using a text editor. There are no SSM screens to configure HDM, and there are only a few utilities that allow the configuration to be changed dynamically. In general, if the configuration of HDM needs to be changed, an administrator must edit the appropriate configuration file, then restart HDM. The only notable exceptions are that HDM does not need to be restarted to create aggregates and filesets, or to change the message logging policy.

Read Sections 7.2.3.1 through 7.2.3.5 for a detailed description of how to set up the configuration files.

To make a DFS aggregate known by HDM, use the **create_fsys** utility, described in Section 7.4.2.7. To create a fileset, use the appropriate SSM screen, described in Section 6.5.1.1. HDM must be running before using either utility.

7.3.2 Starting HDM

There are three ways to start HDM. It can be started directly by issuing the **hpss_hdm** command; it can be started indirectly by using the **hdm_admin** utility's **start** subcommand; or it can be started by **rc.dce** or **start.dfs** whenever the system is booted. (Some of the scripts that **rc.dce** calls must be modified to ensure that HDM is started before DFS is started. See section 7.2.2 for details.)

To start HDM with **hpss_hdm**, the **hpss_hdm** command is issued directly from the shell. Assuming the configuration files are in **/var/hpss/hdm** and that **ServerID** is 1, the command line for starting HDM is:

/usr/lpp/hpss/bin/hpss_hdm /var/hpss/hdm/<hdm-id>/config.dat 1

When HDM is started, it immediately verifies that it is not already running. If it is, a few diagnostic messages will be issued, and HDM will shut down.

Using **hdm_admin** to start HDM has the advantage that it checks to ensure that HDM is not already running. See Appendix I for the **hdm_admin** man page. Assuming the usual defaults, the command for starting HDM with **hdm_admin** is:

hdm_admin start

If HDM is to be started from rc.dfs, it is vital to have the script verify that HDM starts successfully before allowing the managed aggregates to be exported. If this precaution is not observed, users will be able to change DFS files without having the corresponding changes made to HPSS. If that happens, it will be quite difficult to re-synchronize the DFS and HPSS file systems.

Also, when starting HDM from rc.dfs, make sure the Flags configuration parameter is set to disable output to stdout. Standard output can also be redirected to /dev/null, but this will make HDM perform more slowly.

See Section 7.2.2 for details on how to start HDM automatically using **rc.dfs**.

Generally, it is not possible to start HDM when it is already running. However, there may be times when it is impossible to kill an earlier instance of HDM. If this happens, it may be possible to work around this problem by force starting HDM using the **-f** switch with **hpss_hdm**, or the **-force** option with **hdm_admin start**. For example:

/usr/lpp/hpss/bin/hpss_hdm -f /var/hpss/hdm/hdm2/config.dat 1

hdm_admin start -force

7.3.3 Handling multiple HDM servers

By default, HDM will find its configuration files in **/var/hpss/hdm/hdm1,** and will also write its log files to this directory. By default, HDM will search for **Server ID** with the value of 1 in **config.dat**, and use a default shared memory key. Different **ServerID**s must be used if multiple HDM servers are to be run on the same machine. Also, when multiple HDM servers are to be run, different directories should be set up for the configuration, and each HDM must use a different shared memory key.

To change **hdm_admin** to use different defaults, set the following environment variables:

HPSS_HDM_SHMEM_KEY	shared memory key (default 3789)
HPSS_HDM_SERVER_ID	server id (default 1)
HPSS_PATH_HDM	path to files (default /var/hpss/hdm)

When running running several HDMs on one machine, it is important that the "**permissiveMount**" flag be set in each HDM's **config.dat**. If this is not done, it will be impossible to use one of HDM's aggregates. If a single HDM is being used, the "**permissiveMount**" field should not be set.

7.3.4 Restarting HDM

HDM must be restarted after the configuration files have been changed, or when HDM is not working correctly.

Configuration files may be edited while HDM is running, but the change will not take effect until after HDM has been restarted.

HDM can be restarted by manually stopping and then starting HDM, but it is more convenient to use the **restart** option with **hdm_admin**. The utility will stop the previous instance of HDM, wait for HDM to finish shutting down, then start the new instance. The command to restart HDM is:

hdm_admin restart

7.3.5 Stopping HDM

The preferred mechanism to stop HDM is with the **stop** option to **hdm_admin**:

hdm_admin stop

This command ensures that HDM is shut down in an orderly fashion.

7.3.6 Using hdm_admin

The status of HDM can be monitored using various **hdm_admin** commands. For example, the **hdm_admin** command **ps** can be used to determine which processes are running, and **fsstat** and **ftstat** can be used to determine the status of file systems and filesets, respectively. This section describes some of these commands in detail. See Appendix I for more information on the **hdm_admin** commands.

To determine which processes are running and what they are doing, the **hdm_admin** command, **ps**, can be used. It will produce the following type of output:

PROCESS	PID	STATUS	FILESYSTEM
main	33716	running	all
tcp	40888	wait_req	all
destroy	36022	suspended	all
data	43706	idle	all

namespace	35516	idle	all
admin	25790	idle	all
dispatch	35776	running	bkup1
migrate	30402	suspended	bkup1
purge	50628	suspended	bkup1
dispatch	34246	running	mirror1
migrate	11720	suspended	mirror1
purge	49866	suspended	mirror1

Information in the **STATUS** column can be used to determine which processes are working on events and which processes are idle. The information can also be used to determine if the destroy, migrate, and purge processes are running or suspended.

The **hdm_admin** command, **fsstat**, can be used to determine the status of the file systems that are managed by HDM. An example of output from this command is:

MEDIA	FSID	STATUS	TCP
/dev/bkup1	4	up	enabled
/dev/mirror1	5	down	disabled

MEDIA lists the name of the aggregates. **FSID** lists the Ids of the aggregate. **TCP** tells whether the tcp process has been set up to enable access to the file system from the HPSS side. **STATUS** lists the status of the aggregates. Possible values for **STATUS** are:

STATUS	Interpretation
down	Aggregate has not been exported and is unavailable to users.
disabled	Aggregate has been exported but is currently not available to users.
coming_up	Aggregate has been exported, but HDM has not yet finished setting it up.
up	Aggregate has been exported, HDM is managing it, and it is available to users.
coming_down	Aggregate has been detached, HDM is in the process of bringing it down, and it is not available to users.

The **hdm_admin** command, **fsstat**, can also be used to determine detailed information about a particular file system. For example, to get information about a file system named **tardis_mirror1**, use the command **fsstat tardis_mirror1**. An example of the output from this command is:

file system id **4**
file system status **up**
tcp status **enabled**

migrate status	**suspended**
start time	**09/29 09:01:58**
stop time	**09/29 09:01:58**
bytes moved	**0 (0)**
recent file	**0 (0)**
files considered	**0**
files migrated	**0**
busy and new files	**0**
errors	**0**
purge status	**suspended**
start time	**09/29 09:01:58**
stop time	**09/29 09:01:58**
bytes purged	**0 (0)**
this cycle	**0 (0)**
target	**0 (0)**
files purged	**0**
purge errors	**0**

Note that the start time shows when the last migration or purge started, and the stop time shows when it stopped. If a migration or purge cycle is underway, the stop time will be replaced by the word **running**. The bytes moved value refers to the total number of bytes moved in the current cycle. In principal, the recent file value ought to help determine whether progress is being made on the file currently being purged. In practice, this value will usually be zero because of the way the migration process works. The busy and new files field indicates files that were skipped because they were in use or were too new to be eligible for migration.

The purge target refers to the number of bytes that must be purged before HDM will stop purging files.

The **hdm_admin** command **ftstat** can be used to determine which filesets are managed by HDM. An example of output from this command is:

NAME	FSID	STATUS
bogus.bkup	4	needs_info
hpss.bkup	4	ready
bogus.mirror	5	needs_info
hpss.mirror	5	ready
hpss.mirror.fileset	5	ready

NAME lists the name of the filesets. **FSID** lists the Ids of the aggregate on which the fileset resides. **STATUS** lists the status of the fileset. Possible values for **STATUS** are:

STATUS	**Interpretation**
needs_info	HDM needs additional information about the fileset before it can be made available to users. This probably means the HPSS fileset has not yet been created.
missing	Fileset appears in filesys.dat but does not exist on DFS. This probably means the DFS fileset has been deleted, but HDM has

	not yet updated its tables. It can also mean HDM encountered a problem enumerating the DFS filesets.
unknown	HDM could not determine the status of the fileset.
ready	Fileset is ready for use.

The **hdm_admin** command **stats** command can be used to determine which operations have been performed by HDM, and how many operations have had errors. An example of output from the command is:

	SYNCPOST		SYNCPOST		
TYPE	EVENTS	ERRORS	EVENTS	ERRORS	PENDING
create	12	0	12	0	0
remove	3	0	3	0	0
symlink	1	1	0	0	0
link	1	0	1	0	0
perm	1	0	1	0	0
destroy	3	0			
read	4	0			
write	0	0			
trunc	0	0			
mount	0	0			
preunmount	0	0			
unmount	0	0			
nospace	0	0			

A large numbers of errors could indicate a problem. If the pending counts stay large for very long, it suggests that HDM is having trouble processing events.

Several **hdm_admin** commands can be used to monitor HDM activity, among these are **queue**, **mlog**, dlog, and **zlog**.

The **queue** command lists the status of events that have been received by HDM. An example of output from this command is:

TYPE	STATUS	AGE	NAME(S)
symlink	pre	26	file1 file2
create	pre	2	big_data_file.dat
create	queued	2	doit.c

create	queued	2	Makefile
create	queued	2	chris.0
remove	waiting	2	copy.0
create	queued	1	surae
destroy	queued	0	

TYPE specifies the type of event. **NAME(S)** specifies the name of the file or files involved in that operation. **AGE** specifies the number of seconds that have elapsed since the event was received. The presence of entries with large values for **AGE** suggests that the system could be hung.

STATUS specifies the status of event processing. As events are processed, **STATUS** values are updated as they step through the following values in order:

STATUS	Interpretation
queued	The event has not yet been assigned to an event handler.
alloc	The event handler has allocated event resources, but no processing has begun.
pre	An event handler has begun preliminary processing for the event.
waiting	The event handler has requested the DMAP Gateway to perform some operation for this event, and is now waiting for DFS to finish its part of the job. If the event stays in this state more than a few seconds, it could mean DFS is hung.
post	The event handler has requested the DMAP Gateway to commit the transaction. If the event stays in this state more than a few seconds, it could mean the gateway is hung.
dmattr	The event handler is updating DMAP attributes. Because this activity is quick, this state will not be seen often.

The **mlog** command lists the contents of the main event log, which contains information about events that will affect the name space (creates, removes, etc.). An example of output from this command is:

TYPE	*STATUS*	*AGE*	*NAME(S)*
remove	alloc	0	copy.0
create	waiting	0	wendy.new

TYPE, **STATUS**, **AGE** and **NAME** have the same meanings as described in the output from the **queue** command.

The **dlog** command lists the contents of the destroy log, which contains a record of each file in a mirrored file system that can be destroyed. An example of output from this command is:

STATE	AGE	REF	HANDLE

```
destroyable     2       0       00.00.00.02.[...]
destroyable     2       0       00.00.00.02.[...]
destroyable     2       0       00.00.00.02.[...]
destroyable     1       0       00.00.00.02.[...]
destroyable     1       0       00.00.00.02.[...]
destroyable     1       0       00.00.00.02.[...]
----
6 destroy candidates found.
```

STATE specifies what state the file is in. **STATE** values and their meaning are:

STATE	Interpretation
pending	The decision to destroy the file has been delayed. This typically only happens at startup time before the destroy process has been started.
destroyable	The file can safely be destroyed, and this will be done as soon as the destroy process gets to it. Most entries will be in this state.
destroying	The destroy process is trying to destroy this file. Entries will be in this state only as long as it takes the destroy process to notify HPSS to be delete the file.
destroyed	The file has been destroyed, but the destroy log entry cannot be deleted until other processes are done referring to it.
limbo	This file has been unlinked by the end user program, but the file is still open. The destroy process will delete the file when the program closes the file. In the mean time, an entry for this file will remain in the destroy log. Files can stay in this state for long periods of time.

AGE specifies the length of time in seconds an entry for this file has been in the destroy log. Files can remain limbo for long periods of time, perhaps days. If the system is heavily loaded, files can remain destroyable for a fairly long time, perhaps minutes. Files should only be in the other states for only a few seconds.

REF is a reference count that indicates how many HDM processes are accessing this destroy log entry. The number will usually be zero. If any other value appears for very long, this could mean something is hung or that HDM has lost track of the file's status. It may be possible to clean up the log by restarting the HDM.

HANDLE is the hexadecimal representation of the file's DM handle. DM handles contain 20 hex digits, but this example only shows a few of them so the example will fit on one line. The handle can be used as input to other command-line utilities to find other information such as the bitfile id of the file.

The **zlog** command lists the contents of the zap log, which contains a record of each file in an archived file system that can be destroyed. An example of output from this command is:

```
AGE        HANDLE

2          00.00.00.02.[...]

1          00.00.00.02.[...]

1          00.00.00.02.[...]

1          00.00.00.02.[...]

1          00.00.00.02.[...]

1          00.00.00.02.[...]

1          00.00.00.02.[...]

1          00.00.00.02.[...]

1          00.00.00.02.[...]

1          00.00.00.02.[...]

----

10 zap candidates found.
```

Files listed in this log will unconditionally be destroyed as soon as the destroy process finds them. **AGE** specifies the length of time in seconds that the entry for this file has been in the log; be on the lookout for large values of **AGE**.

7.3.7 Manually Starting Migration and Purge

Normally the administrator does not need to do anything to start file migrations and purges. Rather, these activities will occur on the schedule defined in policy.dat. The one exception to this is when **MigrationDelayTime** or **PurgeDelayTime** has been set to zero, which HDM interprets as an infinite time. In that case, the administrator must initiate migrations and purges manually. On occasion, it may also be necessary to initiate a migration or purge to clear some space on an aggregate that is full.

To initiate a file migration, use the **hdm_admin** command **migrate**. For example:

```
hdm_admin migrate mirror1
```

This command returns as soon as the migration process has been notified to begin a migrate cycle. To determine whether the process has been able to migrate anything or not, inspect the HDM message log. Migrating files does not free space in DFS; the files must also be purged from DFS.

To initiate a file purge, use the **hdm_admin** command **purg**e. For example:

```
hdm_admin purge mirror1
```

As with **migrate**, this command simply notifies the purge process to begin a purge cycle, but does not wait for the process to do anything. Look in the message log to determine the progress of the purge. Attempting to purge files does no good if the files have not first been migrated.

7.3.8 Using the HDM Message Log

The HDM message logs are stored in the directory named by the **MsgFileDirectory** configuration parameter. This directory is usually named **/var/hpss/hdm/<hdm-id>**. The message log files are named **hdm_message.0** and **hdm_message.1**. When one file is full, HDM automatically switches to the other file.

The log files are plain text files, so they can be examined using normal UNIX utilities. For example, to see new messages written to the log, use **tail**:

tail -f /var/hpss/hdm/hdm1/hdm_message.0

New alarm messages can be singled out by using **tail** and **grep**:

tail -f /var/hpss/hdm/hdm1/hdm_message.1 | grep ALM

Although HDM can be configured to write messages to stdout, it usually is not possible to feed the output into a pipe and get satisfactory results.

The following is an example line from the message log:

98.07.10-08:20:07 EVT NONE mai initServer 0,0 263:0 the HPSS/DMAP is starting

The first field lists the date and time when the log entry was made. In this example, the entry was made on July 10th, 1998, 7 seconds after 8:20 AM.

The second field describes the type of log message. In this example, **EVT** indicates that the message is an event message. Following is a table describing the message types:

TYPE	Interpretation
ALM	indicates an alarm message.
EVT	indicates an event message.
DBG	indicates a debug message.
TRC	indicates a trace message.

Depending on how **LogRecordMask** has been set, some of these types of messages may not appear in the file. To change **LogRecordMask** dynamically, use the **hdm_admin** command **report**. For example, to include all types of messages, use:

hdm_admin report alarm debug event trace

To only include alarm messages, use:

hdm_admin report alarm

The third field in the log entry specifies the severity level for the message. In this example, **NONE** indicates that the message has no severity level. Following is a table describing the severity levels.

SEVERITY	Interpretation
CRIT	indicates the message pertains to a critical problem.
NONE	indicates the message has no severity level.

MAJ	indicates the message pertains to a major problem.
MIN	indicates the message pertains to a minor problem.
WARN	indicates the message is a warning.

The fourth field specifies the type of HDM process that generated the message. In this example, the notation "mai" indicates that the messages was generated by the main HDM overseer process. This field can take these values:

dst	destroy process
evt	event dispatcher
han	event handler
mai	main overseer process
mig	migration process
pur	purge process
tcp	TCP process

The fifth field specifies the name of the C function where the message was generated. In this example, the function was **initServer**. This information can be used, along with the text of the error message, to better understand the problem.

The sixth field 0,0 gives a 64-bit request id. In this case, the request id is zero. Use this field to track the progress of a single file creation event for example.

The seventh field 263:0 gives respectively the message number and error code associated with this message. In this case, 263 is the number of the message that reads "the HPSS/DMAP is starting". Use this number to look up errors in the Error Manual to find out more about what the message means.

The last field is the log message explains why the message was written. In the example, HDM was being started.

In some cases, messages may take the form:

Error binding to DMAP TCP Server port: Address already in use

Here the text before the colon describes the situation and the text following the colon explains why it happened. The second half of the message typically indicates the value of errno returned from a system call, or the value of a status returned from some other function.

7.4 Fileset Configuration and Management

7.4.1 Configuring Filesets

An important decision to make when configuring a DFS fileset is the type of the fileset: archived or mirrored. This decision depends on the interface that clients intend to use to access objects in the fileset. If the objects in the fileset will be accessed through both DFS and HPSS interfaces, then the

type of the fileset must be mirrored. If the objects will only be accessed through DFS, then the type of the fileset should be archived.

Another point to consider when selecting the type of a fileset is performance. Be aware that client access to the high-speed I/O interface is only available for files in mirrored filesets. Archived and mirrored filesets exist in both HPSS and DFS name spaces, and some performance overhead is incurred as a result. Whenever files are created and deleted in archived and mirrored filesets, XDSM events are generated and processed. XDSM events are also generated and processed in mirrored filesets for any name space activity that alters the name space, such as, unlinks, renames, and owner or permissions changes, but not in archived filesets. In both archived and mirrored filesets, the overhead of using XDSM is insignificant for DFS client I/O requests, unless the file data must be cached to DFS. When using a mirrored fileset, an HPSS client may also experience delays in I/O requests if file data must be migrated or purged from Episode (the physical file system on which DFS is built). In either case, the length of the delay will depend on the amount of data to be moved, network and media speed, and the load currently on the DFS and HPSS systems. Data and name space activity in HPSS-only filesets will perform at the same rates as previous HPSS releases. The cost incurred for HPSS to support filesets is insignificant.

Finally, because Transarc's implementation of XDSM generates events on a per aggregate basis, **all** DFS/HPSS filesets on a given aggregate **must** be of the same fileset type.

7.4.1.1 Mirrored Name Spaces

No special set up is required for archived filesets, but some initial set up in HPSS is required if the DFS and HPSS name spaces of mirrored filesets are to look identical. HPSS directory entries corresponding to DFS directory entries that are used by DFS clients to reference objects relative to the local cell and the global name space need to be created.

First, create a directory called "..." in the HPSS root directory. In DFS, this directory is used to reference objects by their global name. Then create a directory in "..." and name it with the local cell name (e. g., tardis.sandia.gov). Next, create a directory called "fs" in the local cell directory. For example, if the name of the local cell is "tardis.sandia.gov", the following directory tree will exist after completing the previous steps:

/...

/.../tardis.sandia.gov

/.../tardis.sandia.gov/fs

Finally, in the HPSS root, create symbolic links to the local cell directory and the file system root. Name these symlinks with the same names used by DFS, ".:" refers to the local cell root, and ":" refers to the root of DFS. Using the above example, the symlinks that would be created are:

/.: -> /.../tardis.sandia.gov

/: -> /.../tardis.sandia.gov/fs

7.4.2 Creating HPSS/DFS Filesets

Before a DFS/HPSS fileset can be created, the DFS aggregate on which the fileset will reside must exist, and that aggregate must be registered with HDM. Sections 7.4.2.1 through 7.4.2.6 describe the

steps required to create an aggregate and get it set up so DFS/HPSS filesets may be created on it. Sections 7.4.2.7 through 7.4.2.12 describe the steps required to create a DFS/HPSS fileset. The order in which these steps are performed is important, and should not be altered.

Archived and mirrored filesets must be created in both HPSS and DFS. For many of the steps, the only difference between creating a mirrored or archived fileset is specifying the appropriate fileset type. Any differences, other than this, will be noted. Basically, the steps are:

1. Create DFS Aggregate

2. Mark Aggregate as Managed by HDM

3. Edit dfstab

4. Create HDM File system Entry

5. Export Episode Aggregate

6. Create Directory for Local UNIX Mount Point to DFS Fileset

7. Create DFS Fileset

8. Set DFS Fileset Quota

9. Create HPSS Fileset

10. Create HPSS Junction to Fileset [Optional]

11. Create DFS Fileset Mount Point

12. Set Permissions and Owner for the DFS Fileset

The platforms supporting XDSM may have different commands for certain steps, the procedure outlined is for IBM AIX systems.

7.4.2.1 Creating DFS Aggregate

Before creating a fileset, the aggregate on which the fileset will be created must exist and be known about by HDM. The following steps explain how to create the aggregate and register it with HDM.

First, a logical volume (disk partition) for the aggregate must be created. SMIT can be used to accomplish this task. The aggregate can be created with the **newaggr** command, which must be run as root. (For more information about the **newaggr** command, see *DFS Administration Guide and Reference*.) For our purposes, many of the **newaggr** defaults can be used. The command line for **newaggr** is:

newaggr <aggregate> <block size> <fragment size>

where

aggregate — device name of the aggregate (e.g., /dev/aggr1).

block size	number of bytes in the blocks, must be a of power 2 between 4096 and 65,536 (8192 is recommended).
fragment size	number of bytes in a fragment, and must be a power of 2 between 1024 and the value specified for block size (1024 is recommended).

In the following example an aggregate, that will be used to contain mirrored filesets, called MirroredAggregate1, will be created.

newaggr /dev/MirroredAggrgate1 8192 1024

7.4.2.2 Marking the Aggregate as HDM Managed

The AIX utility, **dmaggr**, which must be run as root, marks the aggregate to indicate that XDSM events and processing are active. The utility, **dmaggr**, can be found in **/usr/lpp/dce/bin**. The command line for **dmaggr** is:

dmaggr -on <name>

where

name	device name of the aggregate for which XDSM is to be activated.

In the following example, the aggregate created in the previous section, MirroredAggregate1, will be marked as HDM managed.

dmaggr -on /dev/MirroredAggregate1

7.4.2.3 Editing dfstab

The **dfstab** file, which contains information about aggregates and partitions that can be exported to the DCE name space, must be edited. (For more information about dfstab, see *DFS Administration Guide and Reference*.) Write permission is required to edit this file. The file can be found in the directory **/opt/dcelocal/var/dfs**. Each entry in the file must appear on its own line, and have the following fields, separated by at least one space or tab, in the order indicated:

device name	device name of aggregate (e.g., /dev/aggr_name).
aggregate name	name of the aggregate (e.g., aggr_name).
file system type	this must be **dmlfs** for XDSM support.
aggregate id	any positive integer unique within this dfstab file.
file system id	this field must be left empty.

At this point, the aggregate exists in DFS, and is marked to generate and process XDSM events.

Using the same example, **dfstab** will be edited and the following entry for MirroredAggregate1 will be added.

/dev/MirroredAggregate1 Mirrored Aggregate1 dmlfs 4

7.4.2.4 Creating the HDM File System Entry

This step requires that the HDM be running, and that the administrator has write permission to the HDM configuration file, **filesys.dat**. To verify that HDM is running use **hdm_admin**.

Since HDM must run on the platform where XDSM is installed, and this platform typically has no HPSS components, no SSM screens are available to help configure HDM. The utility, **create_fsys**, is used to register the aggregate with HDM. It can be found in the directory, **/usr/lpp/hpss/tools/dmapi/dmapitools/create_fsys**. The command line for **create_fsys** is:

create_fsys <host> <port> <fsid> <path> <media>

<option> <mig><purge><stage><key>

where

host	host name where HDM and DFS are running, we recommend including the domain name (e. g., tardis.sandia.gov).
port	port on the host that HPSS will use to communicate with HDM, (6002).
fsid	file system id of the aggregate; must match the aggregate id in dfstab.
path	mount path for aggregate (e.g., /opt/dcelocal/var/dfs/aggrs/aggr1).
media	device name of aggregate (e.g., /dev/dsk/c0t2d0s4 or /dev/aggr1).
option	fileset type; either archive/rename, archive/delete, or mirrored.
mig	name of migrate policy for this file system.
purge	name of purge policy for this file system.
stage	partial or whole.
key	encryption key used to secure communications between HDM andDMAP Gateway, expressed as a 16-digit hexadecimal number.

Option specifies whether the DFS fileset is to be archived or mirrored in HPSS. If the fileset is archived there are two options for managing any files that are deleted from DFS: *rename* - the HPSS object is renamed by prefixing the string "DEL."; and *delete* - the HPSS file is deleted. (See Section 7.4.4.1 for more details about these delete options.) The names of the migrate and purge policies used as input to **create_fsys** must be the same names defined in the HDM configuration file, **policy.dat**.

create_fsys causes HDM to write a new line to **filesys.dat** describing the aggregate. The aggregate must be listed in this file in order for the HDM to process any XDSM events received for this aggregate. The format for the aggregate entry is:

<path> <media> <fsid> <option> <mig> <purge> <stage>

If necessary, **filesys.dat** may be edited directly, but HDM must be restarted for the change to take effect. (As a rule, editing **filesys.dat** directly is not recommended.)

In the following example, the HDM file system entry for MirroredAggregate will be created. Assume that the name of the host is tardis.sandia.gov, and that MigPolicy1 and PurgPolicy4 have been defined in **policy.dat**.

create_fsys tardis.sandia.gov 6002 4 /opt/dcelocal/var/dfs/aggrs/MirroredAggregate1 / dev/MirroredAggregate1 mirrored MigPolicy1 PurgePolicy4 Whole

7.4.2.5 Exporting the Episode Aggregate

The final step in preparing an aggregate for DFS / HPSS filesets is to make the aggregate visible to the DFS name space. This step is accomplished with **dfsexport**, a DFS command, which must be run as root. (For more information about **dfsexport**, see *DFS Administration Guide and Reference*.) The simplified **dfsexport** command line we may use is:

dfsexport <name>

where

name	device name of the aggregate to export or the aggregate name specified in the newaggr command.

In the following example, the aggregate, MirroredAggregate1, used in the previous examples will be exported.

dfsexport /dev/MirroredAggregate1

7.4.2.6 Creating the Local UNIX Mount Point

The fileset must also be mounted in the local UNIX file system, and HDM will do this the first time client activity in this fileset occurs. HDM will also mount the fileset in the local UNIX file system any time it detects that the mount point does not exist. However, HDM does not create the directory where HDM will mount the fileset. Because HDM runs as root, the directory should be owned by root, and grant **rwx** permissions to root.

We suggest that the following directory structure for aggregates and filesets be used:

/var/hpss/hdm/<hdm-id>/aggr/<aggregate name>/<fileset name>

Continuing with the example in the previous sections, the following command would be used to create the local UNIX mount point for the fileset, MirroredFilesetA, which will reside on the aggregate, MirroredAggregate1:

mkdir /var/hpss/hdm/hdm1/aggr/MirroredAggregate1/MirroredFilesetA

7.4.2.7 Creating the DFS Fileset

The previous 6 steps must all be completed before the DFS and HPSS filesets can be created. The DFS command **fts lsaggr** (see *DFS Administration Guide and Reference*) can be used to verify that the aggregate has been created, exported, and is of type **dmlfs**.

The DFS command, **fts create**, can be used to create the DFS fileset. (For more information about **fts create**, see *DFS Administration Guide and Reference*.) For this command to succeed, the administrator must be listed in the **admin.ft** file on the DFS server machine where the aggregate resides. Additionally, the administrator must be listed in the **admin.fl** file on all fileset database machines or own the server entry for the machine where the aggregate resides. The command line for **fts create** is:

fts create <fileset> <machine> <aggregate>

where

fileset	name of the DFS fileset; the name **must be unique** within the DCE cell.
machine	name of host machine where the aggregate resides.
aggregate	name of aggregate where the fileset is to be created.

The output information from this command includes the fileset id, and the fileset id must be remembered for subsequent steps.

Creating the DFS fileset causes an XDSM event to be generated. The HDM processes this event by modifying **filesys.dat** to indicate the presence of the new fileset and marking the fileset as partially configured since it does not yet exist in HPSS.

Continuing with our example, the following command would be used to create the fileset, MirroredFilesetA:

fts create MirroredFilesetA tardis.sandia.gov MirroredAggregate1

7.4.2.8 Setting the DFS Fileset Quota

When a fileset is created, 5000 kbytes are allocated to it. Depending on the intended use of the fileset, this may be inadequate. The DFS command, **fts setquota**, can be used to reset the quota of the fileset. (For more information about **fts setquota**, see *DFS Administration Guide and Reference*.) For this command to succeed, the administrator must be listed in the **admin.ft** file on the machine where the fileset resides. The command line for **fts setquota** is:

fts setquota -fileset <name or id> -size <kbytes>

where

name or id	the name or id of the fileset (e. g., bob.archivefs or 0,,32).
kbytes	maximum amount of disk space in kbytes that all files and directories in this fileset can occupy (e. g., 100000).

Continuing with our example, the following command would be used to set the quota of MirroredFilesetA to 200000 kbytes:

fts setquota -fileset MirroredFilesetA -size 200000

7.4.2.9 Creating the HPSS Fileset

SSM may be used to create the HPSS counterpart of the DFS fileset. On the HPSS Health and Status screen, select "Operations". From that menu, select "Create DFS/HPSS Fileset". The SSM screen, titled "Create DFS/HPSS Fileset", will appear. See Section 6.5.1.1 for more details about this SSM screen.

Following is a list of the first five fields, which must be filled in, and a description of data that should be supplied for each field:

Fileset ID	id of the DFS fileset; use the fileset id produced as output from fts create.
Filesystem ID	id of the aggregate; use value supplied for fsid in create_fsys.
Filesystem Name	path name for device where aggregate resides; use value supplied for media with create_fsys.
HPSS/DMAP TCP Port	port used to contact HDM; use same value supplied port with create_fsys.
HPSS/DMAP TCP Hostname	name of the machine where HDM is running.

The next three fields, User ID, Group ID, and Permissions, need not be changed.

The next two fields, Global Mount Point and Local Mount Points do not need to be filled in for archived filesets. However, these fields must be filled in for mirrored filesets if requests made through the HPSS interface are to succeed. (The HDM writes this information to **filesys.dat,** and later uses this information to construct DFS path names to process requests made by the DMAP Gateway on behalf of HPSS clients.) Following is a description of the data that should be supplied for these two fields:

Global Mount Point	DFS path name where the fileset will be mounted; use same path name that will be supplied as <directory> in **fts crmount**, (e. g., /:/dfs-hpss.filesets/FilesetA).
Local Mount Point	path name on the local UNIX file system where the fileset will be mounted (e. g.,) /var/hpss/hdm/hdm1/aggr/aggr1/hilary).

The next four fields, DMAP Gateway, File Family, Class of Service, and Mount Point Name Server, all have sub-menus from which values may be selected. Upon selecting the DMAP Gateway field, a menu listing the DMAP Gateways in the DCE cell will appear. If there is only one DMAP Gateway in the cell, that DMAP Gateway will automatically be selected. If there are multiple DMAP Gateways, select the one that will be managing this fileset. The DMAP Gateway selected must be configured to contact the HDM where the DFS fileset resides. Select the desired File Family and Class of Service. Only one Name Server will appear on the menu. Select that Name Server. After reviewing the settings on the screen and ensuring correctness, select the "Create" button at the bottom left of the screen. SSM will send a request to the Name Server to create a fileset. Upon

completion, a message will appear at the bottom of the screen. If the create is successful, the message will contain the fileset id of the newly created fileset, otherwise an error message will be displayed in a separate window.

7.4.2.10 Creating the HPSS Junction

This step is optional, depending on how a site administers filesets. The HPSS tools provided to administer archived filesets identify the fileset by name, so it is not essential to create HPSS junctions to archived filesets. However, if an administrator has need to use other interfaces to access objects in an archived fileset, the archived fileset will need a junction. Therefore, we recommend that a directory be created that will contain junctions to archived filesets. We suggest that this directory be created in the HPSS root directory. This directory should be given a name that clearly identifies its purpose, e. g., archived.fs, arch_fs, arch.fs, Archived_Filesets, etc.

As part of keeping the DFS and HPSS name spaces synchronized, creating a DFS mount point (**fts crmount**) may cause an HPSS junction to be created. Specifically, if a DFS fileset will be mounted within an existing mirrored fileset, the DFS mount request will generate an event which HDM will process by creating the equivalent HPSS junction. However, to make it easier to manage and administer mirrored filesets, we also recommend that a directory be created in the HPSS root directory which should be used to contain junctions to mirrored filesets. This directory should be given a name that clearly identifies its purpose, e. g., mirrored.fs, Mirrored_Filesets, mirrored.filesets, Mirrored.Filesets, etc. If a DFS fileset is to be mounted at the DFS file system level (**/.:/fs, /:**), no HDM processing will take place. If the DFS and HPSS name spaces are to remain synchronized, a junction in the HPSS "**/.:/fs**" directory must also be created to the fileset with the same name that will be given to the DFS mount point.

The command line utility, **crtjunction**, can be used to create junctions. See Appendix I for details about using **crtjunction**.

SSM may be used to create junctions. See Section 6.5.1.3 for details. On the HPSS Health and Status screen, select "**Operations**". From that menu, select "**Create Junction**". The SSM screen, titled "Create Junction", will appear. (Care must be taken when choosing the path name of a junction to a mirrored fileset. The path name to the HPSS fileset must be identical to the DFS mount point. For example, if a mirrored DFS fileset is to be called FilesetA and mounted in **/:/mirrored_filesets**, the path name used to create the HPSS junction should be **/:/mirrored_filesets/FilesetA**, where **/:/mirrored_filesets** is a directory in the HPSS root.)

On the "Create Junction" screen either the fileset ID, which was given in the message after the fileset was created, or the fileset name should be entered into the appropriate field on the screen. The other field that must be filled in is **Junction Path Name**. The path name specified for the junction is relative to the root directory. After reviewing the values for these fields and ensuring correctness, select the "**Create**" button at the bottom of the screen. The SSM will send a request to the Name Server to create a junction. If the request completes successfully, the SSM will provide HDM with information about the DMAP Gateway and mount points. Also, a message indicating success will appear at the bottom of the SSM screen. If an error occurs, a separate window will appear stating the error.

In the following example, a junction to the mirrored fileset **MirroredFilesetA** is about to be created. Notice that the name of the junction is also MirroredFilesetA, and that the junction is to be created in **/fs/mirrored_filesets**.

7.4.2.11 Creating the DFS Fileset Mount Point

For DFS clients to access the DFS fileset it must be mounted in the DFS name space. The **fts crmount** command can be used to mount the fileset. This command is processed entirely within DFS, it has no effect on HPSS. For this command to succeed the administrator must have write, execute, control and insert permissions on the directory where the fileset will be mounted. (For more information about **fts crmount**, see *DFS Administration Guide and Reference*.) The command line for **fts crmount** is:

fts crmount <directory> <fileset>

where

directory	the DFS path name where the root directory of the fileset will be mounted; use the path name entered for Global Mount Point on the Create DFS / HPSS fileset SSM screen.
fileset	the name or id of the fileset (e. g., FilesetA or 0,,28).

Continuing with our example, to create a DFS mount point for MirroredFilesetA, the following command would be used:

fts crmount /:/hpss_fs/MirroredFilesetA MirroredFilesetA

where hpss is an existing DFS (and HPSS) directory.

7.4.2.12 Assigning the Fileset Owner and Permissions

Special privileges are required to create a DFS fileset. (The caller of **fts create** must be listed in **admin.ft** or **admin.fl**. See Section 7.4.2.7 for details.) The default settings on the "Create DFS / HPSS Fileset" SSM screens for UID and GID are 0. The system administrator must use the UNIX **chown** or its equivalent through the DFS interface to assign the fileset to the user that will own the fileset. Similarly, the UNIX **chgrp** or its equivalent must be used to change the owning group. As part of processing **chown** and **chgrp** for mirrored filesets, XDSM events are generated. HDM handles the event by requesting HPSS to perform the indicated UID / GID change to the HPSS fileset. If **chown** or **chgrp** completes without error, the UID / GID will have been changed for both the DFS and HPSS filesets.

Also, filesets created in DFS have no DCE ACLs. The fileset is protected only with the UNIX mode bits assigned to it, which defaults to 0700. That is, the owner has **r**, **w**, and **x** mode bits, and group and other have no permissions. By default, the owner also has **c**, **i** and **d**, if the permissions are displayed with **acl_edit**. When a new owner is assigned to the fileset, the permissions will not change. Although the permissions granted to the owner are adequate, it may be desirable to change the permissions that apply to group or other. The UNIX **chmod** or its equivalent may be used through the DFS interface to change the permissions. Once again, for mirrored filesets, an XDSM event will be generated that is processed by HDM resulting in the equivalent permission change for the HPSS fileset.

7.4.3 Deleting Filesets

An HPSS fileset must be completely empty before it can be deleted from HPSS. If a complete fileset deletion is attempted for a fileset that is not empty an error will occur. The DMG's fileset metadata can be deleted but the HPSS objects within that fileset (managed by the Name Server) will be unaffected.

7.4.3.1 Deleting the Junction

Once the fileset is empty, the steps to delete it are the inverse of the steps taken when the fileset was created. First, the junction to the fileset should be deleted, then the fileset should be deleted.

SSM may be used to delete a junction. On the HPSS Health and Status screen, select "Operations". From that menu, select "Delete Junction". The SSM screen, titled "Delete Junction", will appear.

Enter the fully qualified path name of the junction that is to be deleted. After reviewing the path name and ensuring correctness, select the "Delete" button at the bottom of the screen. The SSM will send a request to the Name Server to delete the junction. If the request completes successfully, a message so indicating will appear at the bottom of the screen. If an error occurs, a separate window will appear stating the error.

Junctions can also be deleted by using the command line tool, **deljunction**. See Appendix I for details about **deljunction**.

7.4.3.2 Deleting DFS/HPSS Filesets

When a fileset is deleted from Episode, it need not be empty, as is required to delete an HPSS fileset. Since a fileset may contain a large number of objects, deleting them all could take some time. Therefore, we recommend deleting the Episode fileset, and changing the type of the fileset from archived or mirrored to HPSS-Only. The objects in the HPSS-Only fileset can then be removed at any time, even as a background activity. After all the objects have been removed from the HPSS-Only fileset, the HPSS junctions to the fileset and the fileset itself may be deleted.

7.4.3.3 Deleting the DFS Mount Points

A fileset must not be in use when it is deleted from Episode. Therefore, the first step in deleting an archived or mirrored fileset is to remove the fileset from the DFS name space so clients can no longer access files. It may not be possible to remove all DFS mount points since any user may have created them in the past, but all known mount points should be deleted. (Subsequent references through a DFS mount point where the fileset has been removed will result in an error.) The **fts delmount** command can be used for this purpose. (For more information about **fts delmount**, see *DFS Administration Guide and Reference*.) For this command to succeed, the administrator must have write, execute and read permissions to the directory containing the mount point. The command line for **fts delmount** is:

fts delmount <path name>

where

path name — fully qualified DFS path name of the mount point to be deleted.

7.4.3.4 Deleting the DFS Fileset

A DFS fileset can not be deleted if there are any local UNIX mount points to it. Check **filesys.dat** for the specified fileset entry to determine the local mount point used by HDM, and remove it with the UNIX **unmount** command. To delete the Episode fileset the **fts delete** command may be used. (For more information about **fts delete**, see *DFS Administration Guide and Reference*.) For this command to succeed, the administrator must be listed in **admin.ft** on the specified host machine. Additionally, the administrator must be listed in **admin.fl** on all fileset database machines or own the server entry for the machine where the fileset resides. The command line for **fts delete** is:

fts delete <fileset> <host> <aggregate>

where

fileset	name or fileset id of the fileset to delete (e. g., bobs.archfs; 0,,22).
host	host name of machine where DFS is running (e. g., tardis).
aggregate	device name, aggregate name or aggregate id where the fileset resides. Other values for this field can be found in the first, second and fourth fields of the entry for the aggregate in **dfstab**).

When a DFS fileset is deleted, the HPSS file need not be removed. Removing the HPSS fileset can be done at a later time, or never at all. For example, a project may run out of funding and the DFS fileset could be removed. However, the HPSS fileset still exists, and the data can be recovered at a later time, if funding is resumed.

7.4.3.5 Changing Fileset to HPSS-Only

If the HPSS counterpart of a DFS / HPSS fileset is to be deleted at a later time, the type of the fileset should be changed to HPSS-Only. SSM may be used to change the fileset from archived or mirrored to HPSS-Only. (See Section 6.5.1.4 for details.) On the Health and Status screen, select "**Monitor**". From that menu select "**HPSS Objects**", which will display a sub-menu. Select "Filesets". The SSM screen, titled "Identify A Fileset", will appear. Enter either the fileset id or the fileset name, then select the "**Get NS Info**" button on the upper right side of the screen. Another SSM screen, titled "Name Server Fileset Information", will appear. Verify that this is the correct fileset. Select the Fileset Type button, and a menu will appear. Select "**HPSS-Onl**y" as the new value for this field. The type of the fileset has now been changed.

7.4.3.6 Deleting the DMAP Gateway Fileset

SSM may be used to delete the resources kept by the DMAP Gateway about a fileset. On the HPSS Health and Status screen, select "**Operations**". From that menu, select "**Delete Fileset**". (See Section 6.5.1.4 for details.) The SSM screen, titled "Identify A Fileset", will appear.

Either the fileset ID or the fileset name may be entered into the appropriate field on the screen. After reviewing the fileset ID or fileset name and ensuring correctness, select the "**Get DMG Info**" on the right side of the screen. A new SSM screen, titled, "DMAP Gateway Fileset Information", will appear. Verify from the information on the screen that this is the fileset to be deleted. If so, select the "**Delete Filese**t" button at the bottom of the screen. An SSM Confirmation screen will appear with a menu from which "**DMG-Only Delete**" should be selected. The SSM will send a request to the DMAP Gateway to delete the fileset, and the Name Server fileset information will be left alone. (If there are no objects in the HPSS fileset, "**FULL DELETE**" may be selected from the SSM

Confirmation screen instead of "**DMG-Only**", and this will cause both the DMAP Gateway and Name Server filesets to be deleted.) If the request completes successfully, a message so indicating will appear at the bottom of the screen. If an error occurs, a separate window will appear stating the error.

At any later time, the objects in the HPSS fileset can be removed. After the fileset is empty, the fileset can be deleted following the procedure outlined for deleting HPSS-Only filesets. If the "**FULL DELETE**" option described above was used, be sure to delete any HPSS junction(s) to the fileset, as previously described.

7.4.4 Archived Fileset Management

When HPSS and DFS are functioning normally, HDM automatically backs up the files in an archive fileset from Episode to HPSS, unfortunately, system and hardware failures occasionally occur. These failures may result in losing XDSM create and delete events. When a create event is lost, the file will exist in Episode, but may not be marked for migration to HPSS. Consequently, the file will never migrate from Episode to HPSS. Conversely, when a delete event is lost, the file may continue to exist in HPSS well after it has been deleted from Episode.

To detect and correct these problems, management tools and procedures have been provided. We recommend using these procedures at least once a month for each archived fileset. However, a site may choose to run these procedures more frequently, if a high number of system shutdowns or crashes occur on the machine where Episode is running. While executing these management tools, the HDM migration process responsible for the fileset being administered should be suspended.

7.4.4.1 Archived Fileset Usage and Management

Archived filesets should be used when normal DFS aggregates do not contain enough space for all the files in the filesets on the aggregate. Archived filesets do not maintain name space consistency with HPSS, thus files can only be viewed through DFS interfaces. As the disks fill, file data will be moved into HPSS and freed from DFS. This will give the DFS client the illusion of an infinite disk store. If the file being accessed has been archived to HPSS and the file data is no longer on the DFS disks then the client will see a delay as the data is staged from HPSS back to the DFS disks.

Since archived filesets do not maintain name space consistency with HPSS they have very little name space activity overhead. In fact, name space activity should have little or no delay, except for file creates and deletes, which will have slightly more overhead than normal DFS aggregates (approximately 20-30% depending on the hardware).

Archived filesets should only be used when there is not enough DFS disk space to handle the client's data and the clients only plan to access their files through DFS interfaces. If DFS-only access is desired and enough disk space is available, a better choice would be a normal DFS fileset.

Since HPSS does not manage the name space of an archived fileset it is impossible to recover from DFS disk losses using the metadata and the file data stored in HPSS. To recover from such losses, administrators will need to make system and disk backups. Since XDSM supported aggregates (**dmlfs**) do not support cloning, the only way to make fileset backups is to use **fts dump** and the only way to restore a fileset is to use **fts restore**. Please refer to the AIX documentation describing DFS administration in the document *Distributed File Service Administration Guide and Reference*.

There are two types of archived filesets, **archive/delete** and **archive/rename**, which differ in the way files deleted from DFS are handled.

In **archive/delete** filesets the HPSS file is immediately deleted following the DFS request to delete the file. Because of this, it may not be possible to fully recover with a DFS backup if there are DFS disk losses on an aggregate of this type. If an **archive/delete** aggregate is recovered using **fts restore**, DFS may contain references to file data that it believes is stored in HPSS but which has actually been deleted. There is no recovery for this situation, the file data is lost. However, since the file has been deleted in DFS, simply deleting it again may be all that is necessary.

In **archive/rename** filesets the HPSS file will be renamed in HPSS when it is deleted from DFS. The renamed files will not be removed from HPSS until the administrative tool, **archivedel**, is used to remove the old files. See Section 7.4.4.6 for more details about using **archivedel**. If a DFS disk with filesets is lost, **fts restore** can be used to restore the fileset back to the state of the last dump and, then the tool **archiverec** (described in Section 7.4.4.8) can be run to recover any files deleted since the last full dump. Recovery for this type of file set is described in more detail in the following sections.

7.4.4.2 Archived Fileset Tools

A number of tools have been provided to administer archived filesets. Although Appendix I provides details on each of these tools, this section briefly describes the purpose of these tools and how the tools can be used with one another to accomplish an administrative task. The tools described are:

1. archivelist
2. archivedump
3. archivedel
4. setdmattr
5. archivecmp
6. archiverec

7.4.4.3 archivelist

The tool, **archivelist**, traverses an HPSS fileset, and produces a list of HPSS path names to every file in the fileset. The number of entries in the output file will be the number of files contained in the HPSS archived fileset. The output from **archivelist** can be sorted and then compared with the sorted output from the utility **archivedump** (which describes the DFS file structure) to determine if the archived fileset has any inconsistencies.

When running this tool make sure there is enough space in the local file system to handle the output file.

archivelist can be found in **/usr/lpp/hpss/tools/dmapi/archivelist**. For this tool to run, the environment variable, HPSS_LS_NAME, must be set to identify the HPSS Location Server, typically, **/.:/subsys/hpss/ls/group**. In addition, clients running this tool must be logged into DCE as the DMG principal. The command for **archivelist** is:

archivelist <name of archived fileset> > <output file>

After running **archivelist**, the output should be sorted. Sorting does not need any special user privileges. For example:

sort <output file> > <archive sortfile>

7.4.4.4 archivedump

The tool, **archivedump**, traverses the DFS fileset, and produces information about each file in the fileset. The number of entries in the output file will be the number of files contained in the DFS archived fileset, this number may be quite a bit larger than the output produced by **archivelist**, since it is possible to configure the HDM to only migrate files from DFS into HPSS that are larger than a specified size. The output from **archivedump** can be sorted and then compared with the sorted output from **archivelist** (which describes the HPSS file structure) to determine if the archive fileset has any inconsistencies. When running this tool make sure there is enough space in the local UNIX file system to handle the output file. The information provided in the output file is:

<hpss name> <access age> <file state> <Episode name>

where

hpss name	path name of the file archived to the HPSS fileset.
access age	time the file was last accessed or written.
file state	one of the following four states: HPSS_BACKED_DATA_VALID - file archived to HPSS, and Episode believes HPSS has all the valid data. HPSS_BACKED_DATA_INVALID - archived to HPSS at some point in the past, but file has since been altered in Episode. NOT_YET_MIGRATED - file has never been migrated to HPSS, but is marked for migration. ERROR_NO_DMATTRS - occurs when create events are lost by HDM. Mark files in this state as migratable with the tool, **setdmattr** (Section 7.4.4.7.
Episode name	path name of the Episode file.

archivedump can be found in **/usr/lpp/hpss/tools/dmapi/dmapitools/archivedump**. This tool must be run as root on the platform where Episode is running. In addition, the archived fileset must be mounted in the local UNIX file system on this same platform. The syntax for the **archivedump** command is:

archivedump <fileset id> <fileset name> <mount point> <min size> > <output file>

where

fileset id	fileset ID in the format [num].[num] (e. g., 0.4 or 0.22).
fileset name	ASCII name of the DFS fileset (e. g., bob.fileset or fileset.tinytim).

mount point	UNIX path name where fileset is mounted locally.
min size	The minimum size of a file that will migrate from DFS into an archived fileset.

The values for fileset id and fileset name can be found in **filesys.dat**. HDM does not require a local mount point for archived filesets, and the information in **filesys.dat** will determine if a local mount point exists for this fileset or not. If not, use the UNIX **mount** command to create the local mount point. For example:

mount -v dmlfs -o aggregate=<aggr name> -n <host> <fileset name> <local mount>

where

aggr name	name of the DFS aggregate (e. g., ArchivedAggregate1).
host	host name of machine where aggregate (e. g., tardis.sandia.gov).
fileset name	name of DFS fileset (e. g., ArchivedFilesetA)
local mount	path name where fileset is mounted in local UNIX file system (e. g., /var/hpss/hdm/hdm1/aggr/ArchivedAggregate1/ArchivedFilesetA).

After running **archivedump**, the output should be sorted. Sorting does not need any special user privileges. For example:

sort <output file> > <sorted.archivedump>

7.4.4.5 archivecmp

The tool, **archivecmp**, compares the information produced by **archivelist** and **archivedump**, and produces an output file containing a list of files with inconsistencies and suggestions for correcting the inconsistency. Following are the inconsistencies that can be detected and the corrective actions:

- Files in the "ERROR_NO_DMATTRS" state have not yet been migrated to HPSS. Files in this state should have the DM attribute, "MIGRATE", set, allowing future migration of this file to HPSS. Use **setdmattr** to set the DM "MIGRATE" attribute.
- Files in either the "NOT_YET_MIGRATED" or "HPSS_BACKED_DATA_INVALID" states and which have not been accessed for over a day suggest that migration is not being run often enough, or that the migration process is not functioning properly. Check the value of "MigrationDelayTime" in the HDM configuration file, **policy.dat**. This value states the total number of seconds the migration process waits before it resumes . To check that status of the HDM migration process, **hdm_admin** may be used.
- Any files found in HPSS that do not exist in Episode suggest the loss of an XDSM delete event. Depending on the option used to manage the HPSS files deleted from an Episode fileset, the HPSS file should either be unlinked or renamed. Recall that HDM has been configured to either delete or rename HPSS files when an Episode file is deleted from an archived fileset. Check the entry for the fileset in **policy.dat** to determine which delete option has been configured for the fileset.

If the **archive/rename** policy has been used for a fileset there may be many files in the HPSS fileset that are not found in the Episode fileset. These are files that have been deleted from Episode and renamed in HPSS. These files are all prefixed with the string, `"DEL.", and are ignored by **archivecmp**.

archivecmp can be found in **/usr/lpp/hpss/tools/dmapi/archivecmp**, and should be run using the sorted output from **archivelist** and **archivedump**. No special user privileges are required to run **archivecmp**. The syntax for the **archivecmp** command is:

archivecmp <sorted archivedump output> <sorted archivelist output> > <output file>

archivecmp can be found in /hpss/build/4v1/tools/dmapi/archivecmp, and should be run using the sorted output from **archivelist** and **archivedump**. No special user privileges are required to run archivecmp. The syntax for the **archivecmp** command is:

archivecmp [sorted archivedump output] [sorted archivelist output] > [output file]

7.4.4.6 archivedel

Two policies are available for handling HPSS files that are deleted from Episode filesets: **archive/delete**, where the HPSS file is deleted, and **archive/rename**, where the HPSS file is simply renamed by prefixing the string "DEL." to the HPSS file name. An advantage to using **archive/rename** is the ability to restore Episode files from the archived HPSS fileset. However, a large number of "DEL." files may accumulate in the HPSS fileset. Depending on the space allocated to the HPSS fileset this may or may not be a problem. However, a site may periodically want remove old "DEL." files. he tool, **archivedel**, found in **/usr/lpp/hpss/tools/dmapi/archivedel**, may be used for this purpose. Clients running this tool must be logged into DCE as the DMG principal. The syntax of the **archivedel** command is:

archivedel <fileset name> <access age in days>

Any "DEL." files in the fileset that are older than "access age in days" will be deleted from the HPSS fileset. The amount of time required for **archivedel** to complete will depend on the total number of files to be deleted and the size of the fileset. The administrator should be careful not the delete any files that might be used to later restore a fileset.

7.4.4.7 setdmattr

To correct an inconsistency found between the Episode and HPSS archived filesets, it may be necessary to set DM attributes for a file. The tool, **setdmattr**, found in **/usr/lpp/hpss/tools/dmapi/dmapitools/setdmattr**, can be used for this purpose. The administrator must be logged in as root to run this tool. The syntax of the **setdmattr** command is:

setdmattr <path name> <DM attribute> <attribute value>

where

path name	DFS path name of the file whose attribute is to be set.
DM attribute	name of the attribute to be set: CAC_VAL, DIR_HAN, FILESET, HPSS_ID, HPSS_VAL, MIGRATE, OPTION, PURGE.
attribute value	value to which the DM attribute will be set.

The attribute that will most likely need to be set to correct inconsistencies between the Episode and HPSS filesets is "MIGRATE". Typically, "MIGRATE" will need to be set to "1", which indicates that the file needs to be migrated to HPSS. For example:

setdmattr /var/hpss/hdm/hdm1/aggr/test.aggr/fileset.0/file.1 MIGRATE 1

7.4.4.8 archiverec

If an archived Episode fileset is restored from a prior Episode dump (taken with **fts dump**) of the fileset, there may be inconsistencies between the restored Episode fileset and the HPSS fileset. Archived filesets of type **archive/rename** may have "DEL." files that should be restored to their original HPSS names. The tool, **archiverec**, found in **/usr/lpp/hpss/tools/dmapi/archiverec**, may be used for that purpose. Clients running this tool must be logged into DCE as the DMG principal. The syntax of the **archiverec** command is:

archiverec <fileset name> <access age in days>

Any `'DEL." files that are younger than "access age in days" will be renamed to their original name, that is, the "DEL." prefix will be removed. The administrator should take care to make sure to recover all deleted files since the **fts dump** used to recover the fileset.

Unfortunately, it may not be possible to fully recover file data from an archive/delete fileset. See Section 7.4.5.1 for more details.

7.4.5 Archived Fileset Recovery

Recovering from a DFS dump will differ, depending on the way files deleted from Episode are managed in HPSS.

7.4.5.1 Archive/Delete Recovery

It may not be possible to fully recover file data from an **archive/delete** fileset with the DFS tool, **fts restore**. Although it is possible to restore inode information for all the files from the Episode backup of the fileset, it may not be possible to restore all the file data for files that have been deleted since the dump was taken. As required by disk space demands, Episode purges file data from files that have migrated to HPSS. As long as a file exists in Episode, any purged data can be obtained from HPSS. However, once the file is deleted from Episode it will no longer exist in the HPSS. Therefore, even though the inode information can be recovered from an Episode backup, any file data that had been purged will be lost. However, since this applies only to files that have been deleted since the dump was taken, simply deleting these files again will restore consistency to the fileset.

7.4.5.2 Archive/Rename Recovery

Filesets of this type can be fully recovered under the following conditions:

- The DFS administrator has used **fts dump** to make full and incremental backups of the filesets. **fts dump** only backs up anodes and file data resident on the DFS disks at the time of the dump. Data resident in HPSS will not be included in the dump.

- The tool, **archivedel** (or anything with equivalent functionality), has not been used to delete files since the full fileset backup (**fts dump**) was taken. If files were deleted after the dump was taken, a recovered DFS fileset may have references to HPSS files that no longer exist.

If these conditions have been met, the fileset can be fully recovered. However, because data migrates into HPSS after a DFS file is updated, it is possible that some files will contain data altered after the last fileset backup. The data will be at least as new as the last incremental backup. Please refer to the AIX document , *DFS Administration Guide and Reference*, for any of the following steps that refer to DFS specific commands (**fts**, **newaggr**, **dmaggr**, **dfsexport**).

7.4.5.3 Restore the DFS Fileset

The fileset should be recovered to DFS using **fts restore**. Examples of using **fts restore** can be found in the AIX document *Distributed File Service Administration Guide and Reference*. **fts restore** only restores the fileset anodes and any data that was resident on the Episode disks at the time the **fts dump** was taken. Any file data stored in HPSS is not restored with this step.

1. **dfsexport -detach** the damaged aggregate, so the HDM disarms any DMAPI events currently being caught on that aggregate.

2. Save a copy of the **filesys.dat** file containing the entry for the aggregate that is to be recovered, and call it **filesys.dat.recover**. This will make it easier to recreate the aggregate and the filesets for the DMAP Gateway and the HDM. Then, edit **filesys.dat** and delete any references to the aggregate to be restored and the filesets that reside on the aggregate.

3. Turn on the **permissiveMount** option in the **Flags** parameter in the file **config.dat**.

4. Stop and restart HDM. This will cause the HDM to stop processing events for the lost aggregate.

5. Build a new aggregate for the lost aggregate and make sure it has the same ID and aggregate name as the lost aggregate.

6. Run **dmaggr -on** for the newly created aggregate so it will generate XDSM events. Modify **/opt/dcelocal/var/dfs/dfstab**, so the aggregate exists and is of type dmlfs.

7. Run **dfsexport** for the aggregate to make it available for DFS use.

8. Run **create_fsys** to register the aggregate with the HDM. Run this as if the aggregate is being created for the first time.

9. Remove the **permissiveMount** option in the **Flags** parameter in the file **config.dat**, if desired. Stop are restart the HDM, if the change was made.

10. For every fileset to be restored on the lost aggregate, use the DFS utility **fts restore** to restore filesets from the backups generated by **fts dump**. Filesets can only be restored from the last full **fts dump** backup and then applying the latest incremental **fts dump** backups for the fileset. Check **filesys.dat** to guarantee that there is an incomplete fileset entry for each fileset that is being restored. (A "?" will be present in the fileset entry. If the incomplete fileset entry does not appear then the **filesys.dat** file should be editted to add them and the HDM stopped and restarted).

11. For every fileset to be restored, delete its DMAP Gateway fileset entry. Make sure to use the DMG-Only option when deleting the fileset from the DMAP Gateway. The SSM may be used or the command line utility, **delete_fset**, may be used to delete the filesets.

12. For every fileset to be restored, recreate the DMG fileset. The DMG-Only create option should be used since the Name Server information for the fileset still exists. The information necessary to recreate a fileset can be found in **filesys.dat.recover** (saved in step 1). The filesets can be created through the SSM or with the utility, **create_fset**. After this has been done, the incomplete fileset entries generated in step 7 should now be complete and contain all necessary information to access files in the fileset.

13. The **archiverec** tool should be run to recover any files that have been deleted since the last full backup. Since this tool will rename any deleted files, the amount of time required to run it will depend on how many files are in the fileset and the number of files in the fileset that have been deleted since the last full backup. Make sure that any deleted files are recovered beyond the start of the last full **fts dump** taken for the fileset. This step must be done for each fileset to be restored and can be done at the same time for each fileset.

14. After completing the previous two steps, the fileset can be made available for client use. However, there may be files on HPSS that are no longer referenced by DFS after the fileset recovery. These files should be found and deleted with the steps described in Section 7.4.4.8.

If the fileset id, the aggregate id, or the aggregate name of the recovered fileset were changed during the **fts restore**, then these items must be changed in **filesys.dat** and in the fileset information kept by the DMAP Gateway and Name Server. It is desirable to recover filesets to an aggregate with the same name as the one that was lost and to keep the fileset Id the same to avoid altering **filesys.dat** and the DMAP Gateway and Name Server metadata. If necessary, use SSM screens to change the DMAP Gateway and Name Server information. See Section 6.5.1.4 for details about changing this metadata.

7.4.5.4 Archived Fileset Import

Previous sections of this document described how to configure filesets and how to create archive DFS / HPSS filesets. However, a different procedure need to be followed when an existing DFS fileset is to be imported to become an archived fileset. This section describes the procedure.

7.4.5.5 Initial Import a DFS fileset into HPSS (Archived Only)

Following is a description of each step needed to import a DFS fileset into HPSS, creating an archived DFS / HPSS fileset. The following assumptions are made: the HDM is up and running, the DFS fileset type is **lfs**, and the platform is AIX.

1. Change the DFS fileset type from **lfs** to **dmlfs**. Run **dmaggr -on** for the aggregate to allow XDSM events to be generated.

2. Use **create_fsys** to register the DFS aggregate with HDM. (For more information on the **create_fsys** tool, see Section 7.4.2.4.) This tool must be executed on the machine where the HDM is running. The tool, **hdm_admin**, which must be run as root, can be used to verify that HDM is running. For this tool to work, the administrator must have write permission to the HDM configuration file, **filesys.dat**.

3. Edit **filesys.dat** to include the new fileset entry.

4. Use **hdm_admin** to stop and restart HDM so that the changes made in step (3) take effect. **hdm_admin** and must be run as root.

5. Use SSM to create the archived fileset in HPSS.

6. Use the AIX **mount** command (using the **-v dmlfs** option) to mount the DFS fileset in the local UNIX file system so the set up required to migrate file data takes place. See Section 7.4.4.4 for an example of using **mount**.

7. Use **hdmdump** to setup the data to be migrated from DFS to HPSS. The **SetMigrate** optional parameter must be used so that all Epsiode files are marked migrateable. This will caused normal HDM migration to to moved DFS file data into HPSS.

Example Initial Import of DFS Fileset to Archived Fileset

In the following example an existing DFS non-archived fileset named `ProjectA' on DFS aggregate `archive.aggr' will be modified to become a DFS/HPSS archived fileset. Following is a list of assumptions:

Aggregate ID	5
Aggregate Mount Path	/opt/dcelocal/var/dfs/aggr/archive.aggr
Aggregate Name	archive.aggr
Archive Management Type	Rename
Block Size of Aggregate	8192
Device Name of Aggregate	/dev/archive.aggr
DMAP Gateway Desc Name	DMAP Gateway
Fileset Class of Service	<not specified>
Fileset Family ID	1 (Project A Family)
Fileset Group	ProjectA (GID 456)
Fileset ID	0,,8
Fileset Name	ProjectA.Fileset
Fileset Owner	FilesetOwner (UID 123)
Fileset Permissions	rwxrwx--- (770)
Fragment Size of Aggregate	8192
Global Mount Point	/:/ProjectA.Fileset
Host's HDM TCP port	6002
Host where DFS & HDM run	ballzooka.llnl.gov
Host where Gateway runs	ripsaw.llnl.gov
Local Mount Point	/var/hpss/hdm/hdm1/aggr/archive.aggr/ ProjectA
Mount Point Name Server	Name Server
Path to HDM config.dat	/var/hpss/hdm/hdm1/config.dat
Path to HDM filesys.dat	/var/hpss/hdm/hdm1/filesys.dat

1. Change the DFS fileset type from **lfs** to **dmlfs**:

   login ballzooka.llnl.gov as root

   dfsexport -detach archive.aggr
   dmaggr -on archive.aggr

edit /opt/dcelocal/var/dfs/dfstab changing the archive.aggr entry's type from lfs to dmlfs

2. Register the DFS aggregate with the HDM:

   login ripsaw.llnl.gov as root

   ```
   create_fsys  ballzooka.llnl.gov  6002  5  \
                    /opt/dcelocal/var/dfs/aggrs/archive.aggr  \
                    /dev/archive.aggr  archive/rename eighthourmig purge.80.70\
                    whole
   dfsexport archive.aggr
   ```

3. Edit filesys.dat to reflect the new fileset entry for the loaded fileset:

   login ballzooka.llnl.gov as root, and after the archive.aggr entry add the line:

   ```
   ProjectA  ?  ?  0.8  ?  0
   ```

4. Shutdown and restart HDM so that the new entry changes in step (3) take effect:

   login ballzooka.llnl.gov as root

   ```
   cd /bin
   ./hdm_admin <hdm shmkey>
   > stop
   > start
   cd /bin
   ./hdm_admin <hdm shmkey>
   > stop
   > start
   ```

5. Use SSM on ripsaw to create the DFS / HPSS fileset: On the HPSS Health and Status screen, select "Operations". From that menu, select "Create DFS / HPSS Fileset". The SSM screen titled "Create DFS / HPSS Fileset" appears. Fill in the fields as follows:

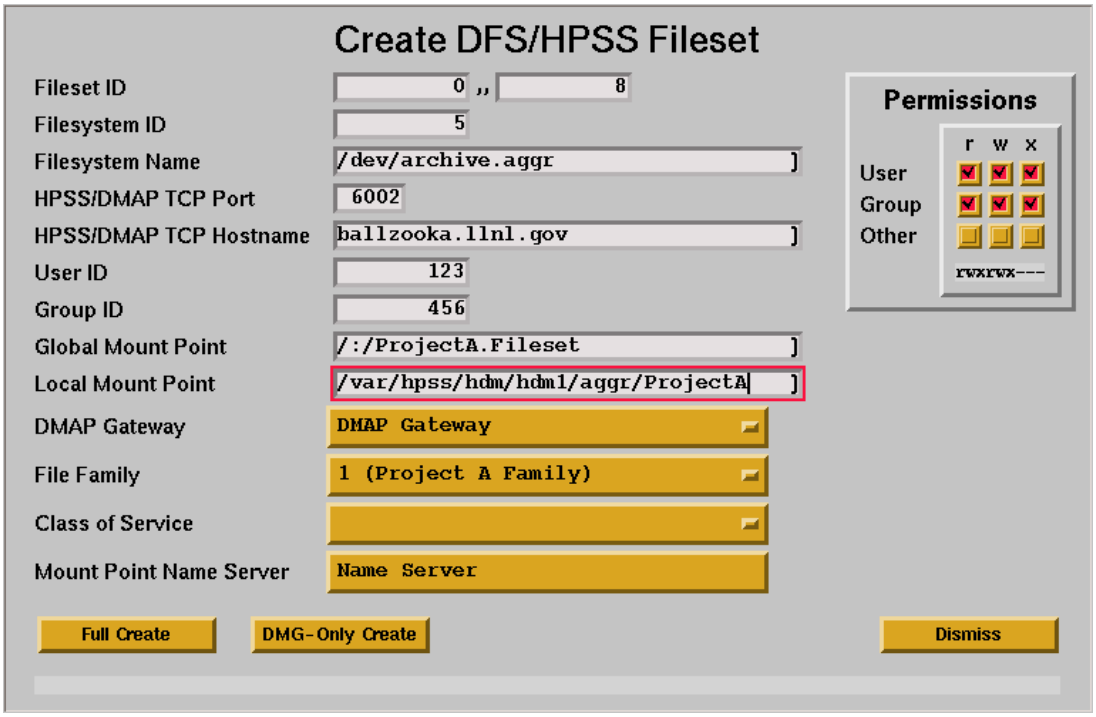


Figure 7-2 Create DFS / HPSS Fileset (Full Create)

After reviewing the settings on the screen and ensuring correctness, select the "Full Create" button at the bottom left of the screen. The SSM will send the request to the specified Name Server to create the fileset. Upon successful completion, a message will appear at the bottom of the screen, otherwise an error message will be displayed in a separate window.

6. Mount the DFS fileset locally:

   login to ballzooka.llnl.gov as root

   ```
   mount -v dmlfs -o aggregate=archive.aggr -n \
                ballzooka.llnl.gov ProjectA \
                /var/hpss/hdm/hdm1/Filesets/ProjectA
   ```

7. Mark the data for migration from DFS to HPSS:

   login to ballzooka.llnl.gov as root

   ```
   /usr/lpp/hpss/tools/dmapi/dmapitools/hdmdump/hdmdump \
                ProjectA /var/hpss/hdm/hdm1/Filesets/ProjectA \
                SetMigrate > /dev/null
   ```

7.4.6 Mirrored Fileset Import and Recovery

Mirrored filesets allow access to the name and data space through **both** DFS and HPSS interfaces, with updates made through the DFS interface being visible through the HPSS interface and vice versa. Thus objects in mirrored filesets have corresponding entries in both DFS and HPSS with identical names and attributes.

Previous sections of this document described how to configure filesets and how to create mirrored DFS / HPSS filesets. However, a different procedure needs to be followed when an existing HPSS or DFS fileset is to be imported to become a mirrored fileset, or to recover from losing DFS metadata in a mirrored DFS / HPSS fileset. This section describes these procedures.

To mirror an existing HPSS fileset, the HPSS name space must be imported into its DFS fileset counterpart. Likewise, to mirror an existing DFS fileset, the DFS name space must be imported into its HPSS fileset counterpart. Similarly, to recover from DFS fileset metadata loss, the HPSS name space must be imported back into its DFS fileset counterpart. A set of instructions and tools are provided which work together to accomplish these tasks. The instructions provided will describe the following:

1. Importing an HPSS Fileset into DFS

   - Initial Import of an HPSS Fileset into a New Aggregate on DFS
   - Recovery from Losing a DFS Aggregate (Creating a New Aggregate)

2. Importing a DFS Fileset into HPSS

Tools provided by HPSS that operate on the DFS filesets are: **hdmdump**, **loadtree**, and **loadhpssid**. These tools are located in **/usr/lpp/hpss/bin**.

Tools provided by HPSS that operate on the HPSS filesets are: **dumphpssfs**, **loadhpssfs**, and **loadhpssdmid**. These tools are located in **/usr/lpp/hpss/bin**.

In addition to the above tools, SSM, DFS Administrative commands, UNIX Administrative commands, and other HPSS XDSM tools, will be used to create and modify the filesets.

Requirements

The DFS and HPSS filesets in a mirrored fileset will need to have matching Fileset IDs, Fileset Names, and Fileset Type of Mirrored.

References

- Information on the individual import / export tools (**dumphpssfs**, **hdmdump**, **loadhpssfs**, **loadhpssid**, **loadtree**, and **loadhpssdmid**) that operate on the DFS and HPSS filesets will be described later and a detailed description is provided in *Appendix I* of the *HPSS Administration Guide*.
- Information on DFS tools and commands is provided in the *DFS Administration Guide and Reference*.
- Information on DFS / HPSS Filesets is provided in the Fileset section of the *HPSS Administration Guide*.

7.4.6.1 Importing an HPSS Fileset into DFS (Mirrored Only)

The following is a list of the logical steps needed to import an existing HPSS fileset into a new DFS fileset, creating a mirrored DFS/HPSS fileset. The specifics of the following steps will be different depending on the platform, if a new aggregate is being created, if an initial import is being performed, or if a recovery is being performed.

- Change the state of the HPSS-Only fileset to prevent users from using the fileset during the import.
- Dump the HPSS fileset.
- Setup the DFS fileset counterpart.
- Load the DFS fileset.
- Load the XDSM IDs into the HPSS fileset.
- Finish DFS/HPSS fileset metadata setup so HDM and DMG know about the new DFS/HPSS mirrored fileset.
- Prepare the new DFS/HPSS fileset to allow access.
- Cleanup extraneous files.

The amount of time required to import an HPSS Fileset into DFS may be significant if there are a large number of objects in the aggregate.

7.4.6.2 Initial Import of an HPSS Fileset into a New Aggregate on DFS (Mirrored Only)

The following is a description of each logical step needed to import an HPSS-Only fileset into a new DFS fileset on a new DFS aggregate, creating a mirrored DFS/HPSS fileset. A couple of assumptions were made: the HDM is up and running and the platform is AIX.

Change the state of the HPSS-Only fileset to prevent users from using the fileset during the import.

1. Use SSM to change the HPSS Fileset State to read only and the Fileset Type to Mirrored. It is extremely important that the fileset not be written to during this process. See Section 6.5.1.4 for details.

2. Use SSM to remove any junctions to the HPSS fileset. See Section 6.5.2 for details. This is an optional step, but is strongly recommended to reduce the possibility of a user modifying the HPSS fileset during the import process.

Dump the HPSS fileset.

3. Use the HPSS tool, **dumphpssfs**, to generate a UNIX file containing an ASCII text representation of the HPSS fileset. This step may take awhile depending on how many objects the fileset contains and the speed of the processor. So, while this step is running, continue with steps (3) through (10). Step (11) requires the completion of this step. The **dumphpssfs** tool creates a checkpoint / restart file in the current working directory (thus requiring write access to the current working directory) and has to be run as DCE principal `hpss_dmg'. This tool can be restarted if it should fail for some unforeseen problem (e.g. power failure). Refer to Appendix I of this manual for a detailed description of this too

Steps 4-10 can be performed at the same time that step 3 is being performed. Step 3 may take a significant amount of time if there are many objects in the fileset.

Setup the DFS fileset counterpart.

4. Before creating the DFS fileset, a logical volume (disk partition) for the aggregate must be created. The aggregate will be created with the **newaggr** DFS command, which must be run as root. (For more information about creating the aggregate and how to use it in the DFS / HPSS environment, see *DFS Administration Guide and Reference* and Section 7.4.2.1.)

5. Use the AIX utility, **dmaggr**, to mark the aggregate to generate and process XDSM events. This utility must be run as root. See Section 7.4.2.2 for details about using **dmaggr**.

6. The **dfstab** file which contains information about aggregates and partitions that can be exported to the DCE name space, must be edited to contain information about the new aggregate. (For more information about dfstab and how to modify it for the DFS / HPSS environment, see *DFS Administration Guide and Reference* and Section 7.4.2.3.) Write permission is required to edit the file. The file can be found in the directory **/opt/dcelocal/var/dfs**. Each entry in the file must appear on its own line. The file system type for the new aggregate must be **dmlfs** for XDSM support.

7. Make the aggregate visible to the DFS name space. SMIT can be used to accomplish this task or the aggregate can be made visible with the **dfsexport** DFS command, which must be run as root. (For more information about **dfsexport** and how to use it in the DFS / HPSS environment, see *DFS Administration Guide and Reference* and Section 7.4.2.5 of this manual titled "Creating a DFS aggregate".) An error may occur during this step if the the HDM is not running with the **permissiveMount** configuration option. If this is the case add the option to the HDM's configuration file and thn stop and restart the HDM (The option will need to be turned off at a later time).

8. Use the DFS command , **fts create**, to create the new DFS fileset. (For more information about **fts create**, see *DFS Administration Guide and Reference*.) For this command to succeed, the administrator must be listed in the **admin.ft** file on the DFS server machine where the aggregate resides. Additionally, the administrator must be listed in the **admin.fl** file on all fileset database machines or own the server entry for the machine where the aggregate resides. The DCE administrative user **cell_admin** has the correct authority to issue **fts create**. The output information from **fts create** includes the Fileset ID; this number must be remembered for subsequent steps.

9. When a fileset is created in DFS, 5000 Kbytes are allocated to it. Depending on the intended use of the fileset, this may be inadequate. The DFS command, **fts setquota**, can be used to reset the quota of the fileset. (For more information about **fts setquota**, see *DFS Administration Guide and Reference*.) For this command to succeed, the administrator must be listed in the **admin.ft** file on the machine where the fileset resides. The quota must be large enough to hold the fileset contents. If the quota is not set large enough, then ann error will occur during step 11.

Load the DFS fileset.

10. Use the AIX **mount** command (using the **-v dmlfs** option) to mount the new DFS fileset in the local UNIX file system so that we can import / load the metadata. (For more information about **mount** and how to use it in the DFS / HPSS environment, see Section 7.4.4.4 of this manual.)

Steps 3 and 10 must be completed before step 11 can be attempted.

11. **cd** to the local mount point and use the HPSS **loadtree** tool to read the UNIX file produced in step (3) and load the DFS fileset. Capture the **loadtree** output (i.e. redirect stdout) to a UNIX text file which will contain mappings between the XDSM ID and HPSS ID, which allow the DFS and HPSS name spaces to be kept in sync.

Load the XDSM IDs into the HPSS fileset.

12. Use SSM to change the HPSS fileset state to allow writing since the next step will need to be able to modify the HPSS fileset metadata objects to contain back-pointers to the DFS fileset metadata objects. If a user attempts to write to the fileset during this time, an error may be returned to the user, depending on whether or not the XDSM ID has been loaded.

13. Use the HPSS **loadhpssdmid** tool to read the UNIX file produced in step (11) and load the XDSM IDs into the HPSS fileset objects. This tool has to be run as DCE principal `hpss_dmg'. Refer to Appendix I of this manual for a description of this tool.

Finish DFS/HPSS fileset metadata setup so HDM and DMG know about the new DFS/HPSS mirrored fileset.

14. Use the HPSS **create_fsys** tool to register the DFS aggregate with the HDM. (For more information **create_fsys**, see Section 7.4.2.4 of this manual.) This tool must be executed on the machine where the HDM is running. For this tool to succeed, the HDM must be running and the administrator must have write permission to the HDM configuration file, **filesys.dat**. **hdm_admin** can be used to verify that HDM is running.

15. Edit **filesys.dat** to include the new fileset entry for the loaded fileset. Since the fileset has already exists, HDM can not modify **filesys.dat** to include the information about the new fileset.

16. Use **hdm_admin** to stop and restart the HDM so the changes made in step (15) take effect.

17. Use SSM to modify the HPSS fileset attributes. The Fileset ID and Fileset Name must match the Fileset ID and Fileset Name from the DFS fileset created in step (8) above.

18. Since the HPSS Name Server already knows about the HPSS fileset, use SSM to create the corresponding DMG-Only fileset using the "**DMG-Only Create**" button on the "Create DFS / HPSS Fileset" screen. Refer to the Section 6.5.1.2] on how to fill in the information.

Verify that the name used to fill in the "HPSS / DMAP TCPHostname" field is in the same exact format as the **HPSSDMAPHostName** parameter in **config.dat**.

Prepare the new DFS/HPSS fileset to allow access.

19. Create an HPSS junction as described in the instructions in the section titled "Creating the HPSS Junction" so that the fileset will be available through the HPSS interface.

20. Mount the DFS fileset as described in Section 7.4.2.6.

21. Assign the fileset owner and permissions as described in Section 7.4.2.12.

Cleanup extraneous files.

22. If no errors are reported, delete the intermediate files generated by **dumphpssfs**, **loadtree**, and **loadhpssdmid**.

Example Initial Import of DFS Fileset to Mirrored Fileset

In the following example, an existing HPSS-Only fileset, named MyHPSSOnlyFileset, will be converted to a mirrored DFS / HPSS fileset named `test.mirror'. A DFS aggregate called `test.aggr' will be created and registered with the HDM on the host ballzooka.llnl.gov. Following is a list of assumptions:

Aggregate ID	3

Aggregate Mount Path	/opt/dcelocal/var/dfs/aggrs/test.aggr
Aggregate Name	test.aggr
Block Size of Aggregate	8192
Device Name of Aggregate	/dev/test.aggr
Directory to store UNIX files	/WorkingDir
Fragment Size of Aggregate	8192
Global Mount Point	:/test.mirror
Host's HDM TCP port	6002
Host where DFS & HDM run	ballzooka.llnl.gov
Host where Gateway runs	ripsaw.llnl.gov
Local Mount Point	/var/hpss/hdm/hdm1/aggr/test.aggr/test.mirror
Migration Policy Name	MigPolicy1
New Fileset Group	FilesetGroup (GID 456)
New Fileset ID	0,,10
New Fileset Name	test.mirror
New Fileset Owner	FilesetOwner (UID 123)
New Fileset Permissions	rwxrwx--- (770)
New Fileset Quota	100,000 kbytes
New Fileset Type	Mirrored
Old HPSS Fileset ID	2178014960,,226000718
Old HPSS Fileset Junction	/JunctionToMyHPSSOnlyFileset
Old HPSS Fileset Name	MyHPSSOnlyFileset
Old HPSS Fileset Type	HPSS-Only
Old & New Fileset File Family	1 (My file family)
Purge Policy Name	PurgePolicy4

1. Use SSM to change the HPSS Fileset State to read only and the Fileset Type to Mirrored: (See Section 6.5.1.2 for details about changing fileset attributes.) On the *HPSS Health and Status* screen, select "**Monitor**". From that menu, select "**HPSS Objects**". From that menu, select "Filesets". The SSM screen titled "Identify A Fileset" appears. Enter the fileset ID or fileset name of the HPSS fileset (e.g. MyHPSSOnlyFileset) and select the "**Get NS Info**" button. The SSM screen, titled "Name Server Fileset Information" appears. Click on "**Read**" so that a check is placed in the box. If a check appears in the "**Write**" or "**Destroyed**" boxes, click on them so that a check does not appear in either box. Also, select "**Mirrored**" from the Fileset Type pop-up list.

2. Use SSM to remove the HPSS junction points to the HPSS fileset: (See Section 6.5.2 about deleting junctions.) On the *HPSS Health and Status* screen, select "**Operations**". From that menu, select "**Delete Junction**". The SSM screen titled "Delete Junction" appears. Enter the Junction Path Name (e.g. /JunctionToMyHPSSOnlyFileset) to the HPSS fileset and select the "**Delete**" button.

3. Dump the HPSS fileset:

   dce_login into ripsaw.llnl.gov as hpss_dmg

   ```
   ksh
   cd /WorkingDir
   . /usr/lpp/hpss/config/hpss_env
   /usr/lpp/hpss/bin/dumphpssfs MyHPSSOnlyFileset Dump.MyHPSSOnlyFileset.out
   exit              # exit dce login
   ```

4. Create the DFS aggregate called `test.aggr' on the host ballzooka.llnl.gov:

login to ballzooka.llnl.gov as root

newaggr -aggregate /dev/test.aggr -blocksize 8192 -fragsize 8192-overwrite

5. Mark the aggregate to generate and process XDSM events:

   login to ballzooka.llnl.gov as root

   dmaggr -on /dev/test.aggr

6. Update the dfstab file with information about the new aggregate:

   login to ballzooka.llnl.gov as root

   edit /opt/dcelocal/var/dfs/dfstab to add the following line
   /dev/test.aggr test.aggr dmlfs 3

7. Make the aggregate visible to the DFS name space. If an error occurs during this step the HDM may have to be stopped and restarted with the **permissiveMount** option set:

   login to ballzooka.llnl.gov as root

   dfsexport test.aggr

8. Create the DFS fileset:

   dce_login to ballzooka as cell_admin

   fts create -ftname test.mirror -server ballzooka.llnl.gov -aggregate test.aggr

   Remember the fileset ID returned from the **fts create** command (e.g. 0,,10)

9. Set the DFS fileset quota:

   login to ballzooka.llnl.gov as root

   fts setquota -fileset test.mirror -size 100000

10. Mount new DFS fileset locally:

    login to ballzooka.llnl.gov as root

    **mount -v dmlfs -o aggregate=test.aggr -n ballzooka.llnl.gov test.mirror **
    /var/hpss/hdm/hdm1/aggr/test.aggr/test.mirror

11. Load the DFS fileset:

login to ballzooka.llnl.gov as root

ftp ripsaw:/WorkingDir/Dump.MyHPSSOnlyFileset.out to ballzooka:/WorkingDir
cd /var/hpss/hdm/hdm1/aggr/test.aggr/test.mirror
**/usr/lpp/hpss/bin/loadtree < /WorkingDir/Dump.MyHPSSOnlyFileset.out **
> /WorkingDir/Load.test.mirror.out

12. Use SSM to modify the HPSS fileset to allow writing: (See Section 6.5.1.4 for details about setting fileset attributes.) On the *HPSS Health and Status* screen, select "**Monitor**". From that menu, select "**HPSS Objects**". From that menu, select "**Filesets**". he SSM screen titled "Identify A Fileset" appears. Enter the fileset ID or fileset name of the HPSS fileset (e.g. MyHPSSOnly-Fileset) and select the "**Get NS Info**" button. The SSM screen, titled "Name Server Fileset Information", will appear. Click on "**Write**" so that a check is placed in the box.

13. Load the XDSM IDs into the HPSS fileset:

    ftp ballzooka:/WorkingDir/Load.test.mirror.out to ripsaw:/WorkingDir

    dce_login to ripsaw.llnl.gov as hpss_dmg

    ksh
    cd /WorkingDir
    . /usr/lpp/hpss/config/hpss_env
    /usr/lpp/hpss/bin/loadhpssdmid < Load.test.mirror.out
    exit #exit dce login

14. Register the DFS aggregate with the HDM:

    login ripsaw.llnl.gov as root

    **create_fsys ballzooka.llnl.gov 6002 3 **
    **/opt/dcelocal/var/dfs/aggrs/test.aggr **
    **/dev/test.aggr mirrored MigPolicy1 **
    PurgePolicy4 whole

15. Edit **filesys.dat** to include the new fileset entry for the loaded fileset:

    login ballzooka.llnl.gov as root

    After the line that describes the aggregate, test.aggr, add the line:

    test.mirror ? ? 0.10 ? 0

    Where 0.10 is the fileset ID returned in step 8 and test.mirror is the name of the fileset.

16. Shutdown and restart HDM so that the changes in step (15) take effect. It may be desireable to turn off the **permissiveMount** option before restarting the HDM.:

    login ballzooka.llnl.gov as root

    cd /usr/lpp/hpss/bin

./hdm_admin <shmkey for the hdm>
> stop
> start

17. Use SSM to modify the HPSS Fileset ID and Fileset Name attributes: (See Section 6.5.1.4 for details about changing fileset attributes.) On the *HPSS Health and Status* screen, select "**Monitor**". From that menu, select "**HPSS Objects**". From that menu, select "**Filesets**". The SSM screen titled "Identify A Fileset" appears. Enter the fileset ID or fileset name (e.g. `MyHPSSOnlyFileset') of the HPSS fileset and select the "**Get NS Info**" button. The SSM screen, titled "Name Server Fileset Information" appears. Enter the new Fileset ID (e.g. 0,,10) returned by the **fts create** command in step (8) into the "**Fileset ID**" field. Both fields of the number must be entered. Click on the **Update** button to the right of the new Fileset ID. Next, enter the new Fileset Name (e.g. `test.mirror') that was used in the **fts create** command in step (8) into the "**Fileset Name**" field.

18. Use SSM to create the corresponding DMG-Only fileset: On the *HPSS Health and Status* screen, select "Operations". From that menu, select "Create DFS / HPSS Fileset". The SSM screen titled "Create DFS / HPSS Fileset" appears. Fill in the fields as follows:

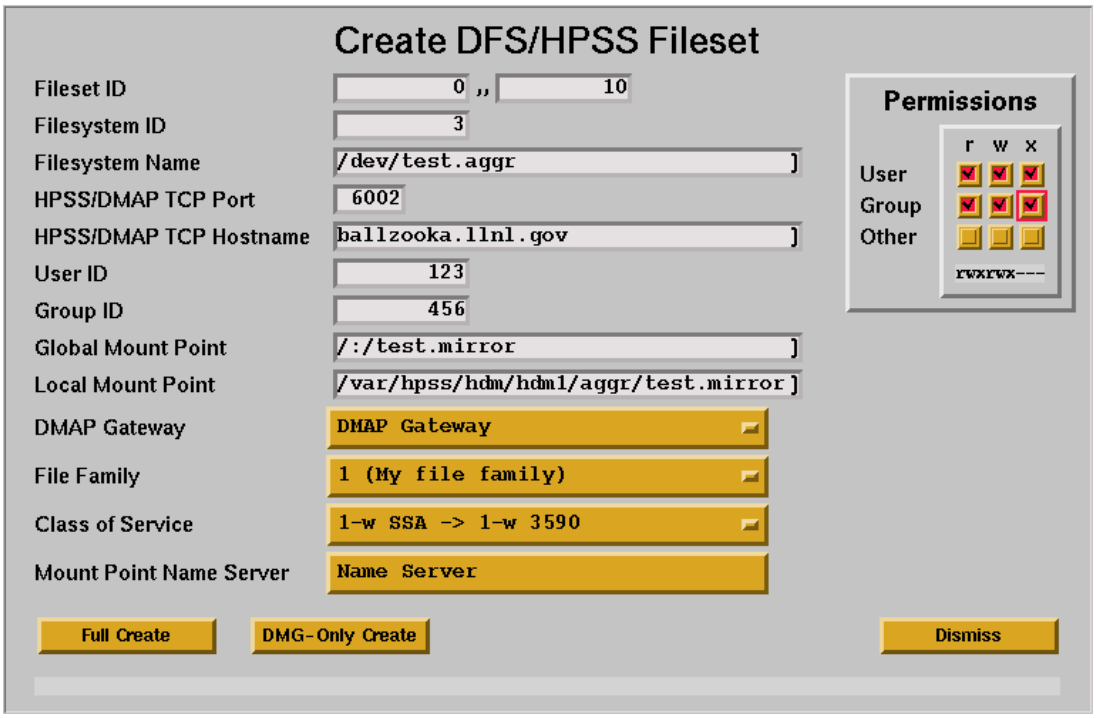


Figure 7-3 Create DFS / HPSS Fileset (DMG-Only Create)

After reviewing the settings on the screen and ensuring correctness, select the "DMG-Only Create" button at the bottom left of the screen. The SSM will send the request to the specified Name Server to create the fileset. Upon successful completion, a message will appear at the bottom of the screen, otherwise an error message will be displayed in a separate window.

The utility **create_fset** can be used instead of the SSM, if desired.

Verify that the name used in the "HPSS/DMAP TCP Hostname" field is the same exact format as the **HPSSDMAPHostName** parameter in **config.dat**.

19. Create a junction to the HPSS fileset: On the *HPSS Health and Status* screen, select "**Operations**". From that menu, select "**Create Junctio**n". The SSM screen, titled "Create Junction" appears. (Care must be taken when choosing the path name of a junction to a mirrored fileset. The path name to the HPSS fileset must be identical to the HPSS mount point.) Enter data for the fields on this window as follows:

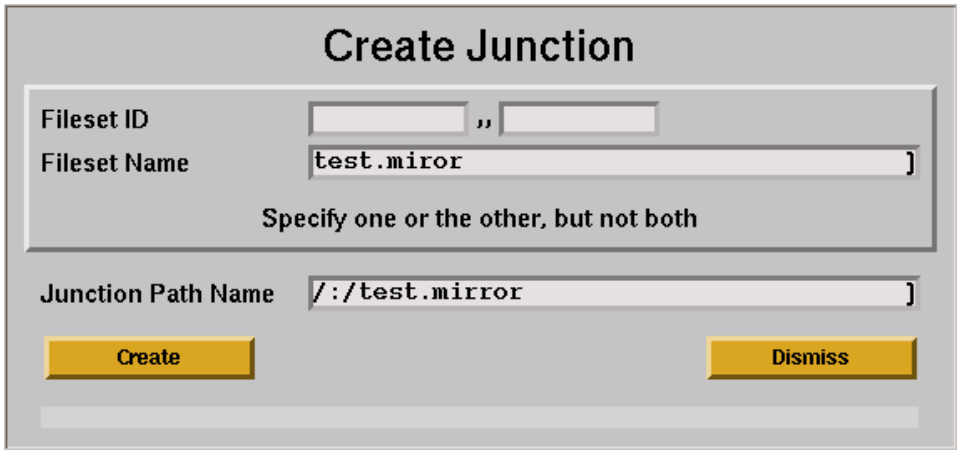


Figure 7-4 Create Junction

After reviewing the settings on the screen and ensuring correctness, select the "**Create**" button at the bottom left of the screen. If the create is successful, a message will appear at the bottom of the screen, otherwise an error message will be displayed in a separate window.

The utility **crtjunction** can be used instead of the SSM, if desireable.

20. Mount the DFS fileset:

    login ballzooka.llnl.gov as root

    fts crmount /:/test.mirror test.mirror

21. Assign the fileset owner, group and permissions:

    login ballzooka.llnl.gov as root

```
chown FilesetOwner test.mirror
chgrp FilesetGroup test.mirror
chmod 770 test.mirror
```

22. Delete the intermediate files:

    login ripsaw.llnl.gov

    ```
    cd /WorkingDir
    rm Dump.MyHPSSOnlyFileset.out
    rm Load.test.mirror.out
    rm DumpHPSSFSRestartFile
    login ballzooka.llnl.gov
    rm /WorkingDir/Dump.MyHPSSOnlyFileset.out
    rm /WorkingDir/Load.test.mirror.out
    ```

7.4.6.3 Recovery after Losing a DFS Aggregate (Creating a New Aggregate - Mirrored Only)

The procedure used to recover after losing a DFS aggregate containing one or more DFS / HPSS mirrored filesets is basically the same as the procedure used for an "Initial Import of an HPSS Fileset into a New Aggregate on DFS". The main difference is that many of the steps must be run once per fileset being recovered. Once again, these assumptions were made: HDM is up and running and the platform is AIX.

Change the state of the HPSS-Mirrored filesets to be recovered to prevent users from using the fileset during the recovery.

1. Use SSM to change the HPSS Fileset State to read only and the Fileset Type to Mirrored. It is extremely important that the fileset not be written to during this process. See Section 6.5.1.4 for details. This step will have to run run once per fileset being recovered.

2. Use SSM to remove any junctions to the HPSS fileset. See Section 6.5.2 for details. This is an optional step, but is strongly recommended to reduce the possibility of a user modifying the HPSS fileset during the recovery process. This step will have to be run once per fileset being recovered.

Dump the HPSS fileset for each fileset being revovered.

3. Use the HPSS tool, **dumphpssfs**, to generate a UNIX file containing an ASCII text representation of the HPSS fileset. This step may take awhile depending on how many objects the fileset contains and the speed of the processor. So, while this step is running, continue with steps (3) through (10). Step (11) requires the completion of this step. The **dumphpssfs** tool creates a checkpoint / restart file in the current working directory (thus requiring write access to the current working directory) and has to be run as DCE principal `hpss_dmg'. This tool can be restarted if it should fail for some unforeseen problem (e.g. power failure). Refer to Appendix I of this manual for a detailed description of this too. This step must be run once for each fileset being recovered. It is possible to run concurrent jobs but make sure that each job has it' own working directory to prevent conflicts. Make sure the ouput file is different for each fileset being recovered.

Steps 4-10 can be performed at the same time that step 3 is being performed. Step 3 may take a significant amount of time if there are many objects in the fileset.

Setup the DFS fileset counterpart for each fileset to recover

4. Fefore re3covering any DFS filesets, the aggregate must first be detached from DFS and all knowledge of the aggregate and filesets being recovered removed from **filesys.dat.** Detach the aggregate by using the **dfsexport -detach** command, edit **filesys.dat** to, remove entries for the filesets and aggregate being recovered, then stop at restart the HDM. Before creating the DFS fileset, a logical volume (disk partition) for the lost aggregate must be created. The aggregate will be created with the **newaggr** DFS command, which must be run as root. (For more information about creating the aggregate and how to use it in the DFS / HPSS environment, see *DFS Administration Guide and Reference* and Section 7.4.2.1.) The aggregate ID of the recovered aggregate can be the same as that of the lost aggregate.

5. Use the AIX utility, **dmaggr**, to mark the aggregate to generate and process XDSM events. This utility must be run as root. See Section 7.4.2.2 for details about using **dmaggr**.

6. The **dfstab** file which contains information about aggregates and partitions that can be exported to the DCE name space, must be edited to contain information about the new aggregate. (For more information about dfstab and how to modify it for the DFS / HPSS environment, see *DFS Administration Guide and Reference* and Section 7.4.2.3.) Write permission is required to edit the file. The file can be found in the directory **/opt/dcelocal/var/dfs**. Each entry in the file must appear on its own line. The file system type for the new aggregate must be **dmlfs** for XDSM support.

7. Make the aggregate visible to the DFS name space. SMIT can be used to accomplish this task or the aggregate can be made visible with the **dfsexport** DFS command, which must be run as root. (For more information about **dfsexport** and how to use it in the DFS / HPSS environment, see *DFS Administration Guide and Reference* and Section 7.4.2.5 of this manual titled "Creating a DFS aggregate".) An error may occur during this step if the the HDM is not running with the **permissiveMount** configuration option. If this is the case add the option to the HDM's configuration file and thn stop and restart the HDM (The option will need to be turned off at a later time).

8. Use the DFS command , **fts create**, to create the new DFS fileset. (For more information about **fts create**, see *DFS Administration Guide and Reference*.) For this command to succeed, the administrator must be listed in the **admin.ft** file on the DFS server machine where the aggregate resides. Additionally, the administrator must be listed in the **admin.fl** file on all fileset database machines or own the server entry for the machine where the aggregate resides. The DCE administrative user **cell_admin** has the correct authority to issue **fts create**. The output information from **fts create** includes the Fileset ID; this number must be remembered for subsequent steps.

9. When a fileset is created in DFS, 5000 Kbytes are allocated to it. Depending on the intended use of the fileset, this may be inadequate. The DFS command, **fts setquota**, can be used to reset the quota of the fileset. (For more information about **fts setquota**, see *DFS Administration Guide and Reference*.) For this command to succeed, the administrator must be listed in the **admin.ft** file on the machine where the fileset resides. The quota must be large enough to hold the fileset contents. If the quota is not set large enough, then ann error will occur during step 11.

Load the DFS fileset.

10. Use the AIX **mount** command (using the **-v dmlfs** option) to mount the new DFS fileset in the local UNIX file system so that we can import / load the metadata. (For more information about **mount** and how to use it in the DFS / HPSS environment, see Section 7.4.4.4 of this manual.)

Steps 3 and 10 must be completed before step 11 can be attempted.

11. **cd** to the local mount point and use the HPSS **loadtree** tool to read the UNIX file produced in step (3) and load the DFS fileset. Capture the **loadtree** output (i.e. redirect stdout) to a UNIX text file which will contain mappings between the XDSM ID and HPSS ID, which allow the DFS and HPSS name spaces to be kept in sync.

Load the XDSM IDs into the HPSS fileset.

12. Use SSM to change the HPSS fileset state to allow writing since the next step will need to be able to modify the HPSS fileset metadata objects to contain back-pointers to the DFS fileset metadata objects. If a user attempts to write to the fileset during this time, an error may be returned to the user, depending on whether or not the XDSM ID has been loaded.

13. Use the HPSS **loadhpssdmid -r** tool to read the UNIX file produced in step (11) and load the XDSM IDs into the HPSS fileset objects. This tool has to be run as DCE principal `hpss_dmg'. Refer to Appendix I of this manual for a description of this tool. Make sure to specify the **-r** recovery option to loadrecovery informsation (this actually sppeds up the process since there are less SFS metadata updates during recovery).

Finish DFS/HPSS fileset metadata setup so HDM and DMG know about the recovered DFS/HPSS mirrored filesets.

14. Use the HPSS **create_fsys** tool to register the DFS aggregate with the HDM. (For more information **create_fsys**, see Section 7.4.2.4 of this manual.) This tool must be executed on the machine where the HDM is running. For this tool to succeed, the HDM must be running and the administrator must have write permission to the HDM configuration file, **filesys.dat**. **hdm_admin** can be used to verify that HDM is running.

15. Edit **filesys.dat** to include all the fileset entries for each recoveredfileset. Since the filesets already exists, HDM can not modify **filesys.dat** to include the information about the new fileset.

16. Use **hdm_admin** to stop and restart the HDM so the changes made in step (15) take effect.

17. For each fileset delete the DMG-Only instance of the fileset, so that the DMG knows nothing about the fileset. Use SSM to modify the HPSS fileset attributes. The Fileset ID and Fileset Name must match the Fileset ID and Fileset Name from the DFS fileset created in step (8) above.

18. Since the HPSS Name Server already knows about the HPSS fileset, use SSM to create the corresponding DMG-Only fileset using the "**DMG-Only Create**" button on the "Create DFS / HPSS Fileset" screen. Refer to the Section 6.5.1.2] on how to fill in the information. Thjis will have to be done for every recovered aggregate.

Verify that the name used to fill in the "HPSS / DMAP TCPHostname" field is in the same exact format as the **HPSSDMAPHostName** parameter in **config.dat**.

Prepare the recovered DFS/HPSS filesets to allow access.

19. For each recovered fileset, create an HPSS junction as described in the instructions in the section titled "Creating the HPSS Junction" so that the fileset will be available through the HPSS interface.

20. For each recovered fileset, mount the DFS fileset as described in Section 7.4.2.6.

21. For each recovered fileset, assign the fileset owner and permissions as described in Section 7.4.2.12.

Cleanup extraneous files.

22. If no errors are reported, delete the intermediate files generated by **dumphpssfs**, **loadtree**, and **loadhpssdmid**.

Example Initial Import of DFS Fileset to Mirrored Fileset

In the following example, an existing HPSS-Mirrored fileset, named test.mirror, will be recovered. A DFS aggregate called `test.aggr' will be recovered and registered with the HDM on the host ballzooka.llnl.gov. Following is a list of assumptions:

Aggregate ID	3
Aggregate Mount Path	/opt/dcelocal/var/dfs/aggrs/test.aggr
Aggregate Name	test.aggr
Block Size of Aggregate	8192
Device Name of Aggregate	/dev/test.aggr
Directory to store UNIX files	/WorkingDir
Fragment Size of Aggregate	8192
Global Mount Point	:/test.mirror
Host's HDM TCP port	6002
Host where DFS & HDM run	ballzooka.llnl.gov
Host where Gateway runs	ripsaw.llnl.gov
Local Mount Point	/var/hpss/hdm/hdm1/aggr/test.aggr/test.mirror
Migration Policy Name	MigPolicy1
New Fileset Group	FilesetGroup (GID 456)
New Fileset ID	0,,10
New Fileset Name	test.mirror
New Fileset Owner	FilesetOwner (UID 123)
New Fileset Permissions	rwxrwx--- (770)
New Fileset Quota	100,000 kbytes
New Fileset Type	Mirrored
Old HPSS Fileset ID	0,,999
Old HPSS Fileset Junction	/:/test.mirror
Old HPSS Fileset Name	test.mirror
Old HPSS Fileset Type	HPSS-Mirror
Old & New Fileset File Family	1 (My file family)
Purge Policy Name	PurgePolicy4

1. Use SSM to change the HPSS Fileset State to read only and the Fileset Type to Mirrored: (See Section 6.5.1.2 for details about changing fileset attributes.) On the *HPSS Health and Status* screen, select "**Monitor**". From that menu, select "**HPSS Objects**". From that menu, select "Filesets". The SSM screen titled "Identify A Fileset" appears. Enter the fileset ID or fileset name of the HPSS fileset (e.g. MyHPSSOnlyFileset) and select the "**Get NS Info**" button. The SSM screen, titled "Name Server Fileset Information" appears. Click on "**Read**" so that a check is placed in the box. If a check appears in the "**Write**" or "**Destroyed**" boxes, click on them so that a check does not appear in either box. Also, select "**Mirrored**" from the Fileset Type pop-up list.

2. Use SSM to remove the HPSS junction points to the HPSS fileset: (See Section 6.5.2 about deleting junctions.) On the *HPSS Health and Status* screen, select "**Operations**". From that menu, select "**Delete Junction**". The SSM screen titled "Delete Junction" appears. Enter the Junction Path Name (e.g. /:/test.mirror) to the HPSS fileset and select the "**Delete**" button.

   The utility **deljunction** can also be used during this step instead of the SSM.

3. Dump the HPSS fileset:

   dce_login into ripsaw.llnl.gov as hpss_dmg

   ```
   ksh
   cd /WorkingDir
   . /usr/lpp/hpss/config/hpss_env
   /usr/lpp/hpss/bin/dumphpssfs test.mirrorDump.test.mirror.out
   ```

exit # exit dce login

4. Detach “test.aggr” from DFS (**dfsexport -detach test.aggr**).

   Edit **filesys.dat** and get rid of the aggregate and any filesets being recovered.

   Stop and restart the HDM.

   Create the DFS aggregate called `test.aggr' on the host ballzooka.llnl.gov:

   login to ballzooka.llnl.gov as root

   newaggr -aggregate /dev/test.aggr -blocksize 8192 -fragsize 8192 -overwrite

5. Mark the aggregate to generate and process XDSM events:

   login to ballzooka.llnl.gov as root

   dmaggr -on /dev/test.aggr

6. Update the dfstab file with information about the new aggregate:

   login to ballzooka.llnl.gov as root

   edit /opt/dcelocal/var/dfs/dfstab to add the following line
   /dev/test.aggr test.aggr dmlfs 3

7. Make the aggregate visible to the DFS name space. If an error occurs during this step the HDM may have to be stopped and restarted with the **permissiveMount** option set:

   login to ballzooka.llnl.gov as root

   dfsexport test.aggr

8. Create the DFS filesets for all filesets being recovered:

   dce_login to ballzooka as cell_admin

   fts create -ftname test.mirror -server ballzooka.llnl.gov -aggregate test.aggr

   Remember the fileset ID returned from the **fts create** command (e.g. 0,,10)

9. Set the DFS fileset quota for each recovered fileset:

   login to ballzooka.llnl.gov as root

   fts setquota -fileset test.mirror -size 100000

10. Mount new DFS fileset locally. This step is necessary for each fileset being recovered:

    login to ballzooka.llnl.gov as root

    **mount -v dmlfs -o aggregate=test.aggr -n ballzooka.llnl.gov test.mirror **
    /var/hpss/hdm/hdm1/aggr/test.aggr/test.mirror

11. Load the DFS fileset. This step is necessary for each fileset being recovered:

    login to ballzooka.llnl.gov as root

    ftp ripsaw:/WorkingDir/Dump.test.mirror.out to ballzooka:/WorkingDir
    cd /var/hpss/hdm/hdm1/aggr/test.aggr/test.mirror
    **/usr/lpp/hpss/bin/loadtree < /WorkingDir/Dump.test.mirror.out **
    > /WorkingDir/Load.test.mirror.out

12. Use SSM to modify each recovered HPSS fileset to allow writing: (See Section 6.5.1.4 for details about setting fileset attributes.) On the *HPSS Health and Status* screen, select "**Monitor**". From that menu, select "**HPSS Objects**". From that menu, select "**Filesets**". he SSM screen titled "Identify A Fileset" appears. Enter the fileset ID or fileset name of the HPSS fileset (e.g. MyHPS-SOnlyFileset) and select the "**Get NS Info**" button. The SSM screen, titled "Name Server Fileset Information", will appear. Click on "**Write**" so that a check is placed in the box.

13. Load the XDSM IDs into the HPSS fileset. Remember to set the recover option:

    ftp ballzooka:/WorkingDir/Load.test.mirror.out to ripsaw:/WorkingDir

    dce_login to ripsaw.llnl.gov as hpss_dmg

    ksh
    cd /WorkingDir
    . /usr/lpp/hpss/config/hpss_env
    /usr/lpp/hpss/bin/loadhpssdmid -r < Load.test.mirror.out
    exit #exit dce login

14. Register the DFS aggregate with the HDM:

    login ripsaw.llnl.gov as root

    **create_fsys ballzooka.llnl.gov 6002 3 **
    **/opt/dcelocal/var/dfs/aggrs/test.aggr **
    **/dev/test.aggr mirrored MigPolicy1 **
    PurgePolicy4 whole

15. Edit **filesys.dat** to include the new fileset entry foreach recovered fileset:

    login ballzooka.llnl.gov as root

    After the line that describes the aggregate, test.aggr, add the line:

test.mirror ? ? 0.10 ? 0

Where 0.10 is the fileset ID returned in step 8 and test.mirror is the name of the fileset.

16. Shutdown and restart HDM so that the changes in step (15) take effect. It may be desireable to turn off the **permissiveMount** option before restarting the HDM.:

    login ballzooka.llnl.gov as root

    cd /usr/lpp/hpss/bin
    ./hdm_admin <shmkey for the hdm>
    > stop
    > start

17. Delete the old fileset entry in the DMG. Make sure to use the dlete DMG-only option. The SSM or the tool **delete_fset** can be used for this step.

    Use SSM to modify the HPSS Fileset ID and Fileset Name attributes: (See Section 6.5.1.4 for details about changing fileset attributes.) On the *HPSS Health and Status* screen, select "**Monitor**". From that menu, select "**HPSS Objects**". From that menu, select "**Filesets**". The SSM screen titled "Identify A Fileset" appears. Enter the fileset ID or fileset name (e.g. `MyHPSSOnlyFileset') of the HPSS fileset and select the "**Get NS Info**" button. The SSM screen, titled "Name Server Fileset Information" appears. Enter the new Fileset ID (e.g. 0,,10) returned by the **fts create** command in step (8) into the "**Fileset ID**" field. Both fields of the number must be entered. Click on the **Update** button to the right of the new Fileset ID. Next, enter the new Fileset Name (e.g. `test.mirror') that was used in the **fts create** command in step (8) into the "**Fileset Name**" field.

18. Use SSM to create the corresponding DMG-Only fileset: On the *HPSS Health and Status* screen, select "Operations". From that menu, select "Create DFS/HPSS Fileset". The SSM screen titled "Create DFS/HPSS Fileset" appears. Fill in the fields as follows:

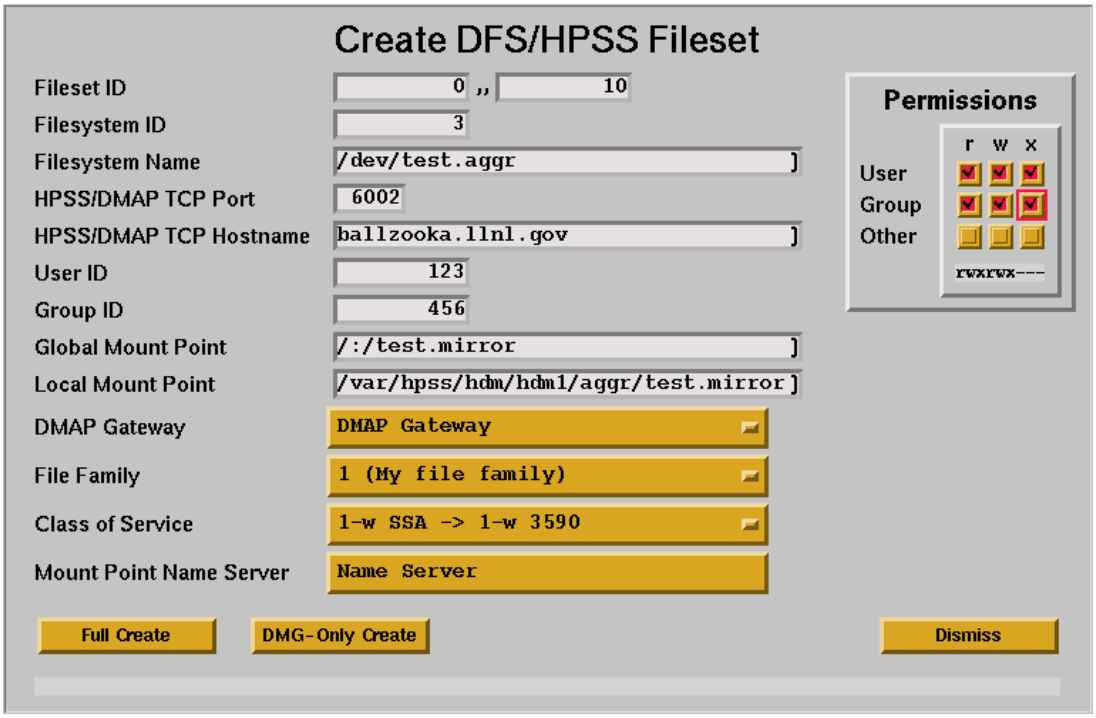


Figure 7-5 Create DFS / HPSS Fileset (DMG-Only Create)

After reviewing the settings on the screen and ensuring correctness, select the "DMG-Only Create" button at the bottom left of the screen. The SSM will send the request to the specified Name Server to create the fileset. Upon successful completion, a message will appear at the bottom of the screen, otherwise an error message will be displayed in a separate window.

The utility **create_fset** can be used instead of the SSM, if desired.

Verify that the name used in the "HPSS / DMAP TCP Hostname" field is the same exact format as the **HPSSDMAPHostName** parameter in **config.dat**.

19. Create a junction to each recovered HPSS fileset: On the *HPSS Health and Status* screen, select "**Operations**". From that menu, select "**Create Junctio**n". The SSM screen, titled "Create Junction" appears. (Care must be taken when choosing the path name of a junction to a mirrored fileset. The path name to the HPSS fileset must be identical to the HPSS mount point.) Enter data for the fields on this window as follows:

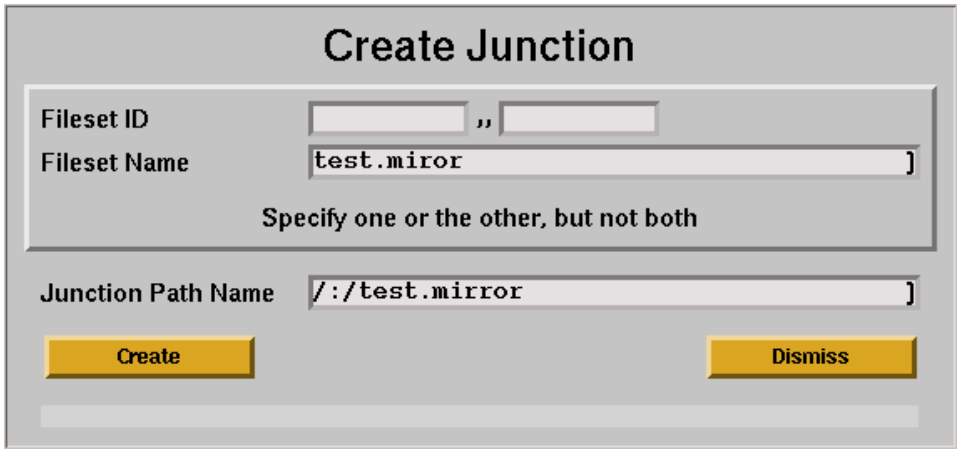


Figure 7-6 Create Junction

After reviewing the settings on the screen and ensuring correctness, select the "**Create**" button at the bottom left of the screen. If the create is successful, a message will appear at the bottom of the screen, otherwise an error message will be displayed in a separate window.

The utility **crtjunction** can be used instead of the SSM, if desireable.

20. Mount each recovered DFS fileset:

    login ballzooka.llnl.gov as root

    fts crmount /:/test.mirror test.mirror

21. Assign the fileset owner, group and permissions for each recovered fileset:

    login ballzooka.llnl.gov as root

    chown FilesetOwner test.mirror
    chgrp FilesetGroup test.mirror
    chmod 770 test.mirror

22. Delete the intermediate files. If many filesets are being recovered at the same time each recovery will have a working directory. The intermediate files should be removed from each working directory:

    login ripsaw.llnl.gov

    cd /WorkingDir
    rm Dump.MyHPSSOnlyFileset.out
    rm Load.test.mirror.out
    rm DumpHPSSFSRestartFile
    login ballzooka.llnl.gov

rm /WorkingDir/Dump.MyHPSSOnlyFileset.out
rm /WorkingDir/Load.test.mirror.out

7.4.6.4 Importing a DFS fileset into HPSS (Mirrored Only)

The following is description of each step needed to import a DFS fileset into HPSS, creating a mirrored DFS / HPSS fileset. These assumptions were made: HDM is up and running, the DFS fileset type is **lfs,** and the platform is AIX.

1. Change the DFS fileset type from **lfs** to **dmlfs**.

2. Use the AIX **mount** command (using the **-v dmlfs** option) to mount the DFS fileset in the local UNIX file system so that we can export / dump the metadata. (For more information about **mount** and how to use it in the DFS / HPSS environment, see Section 7.4.4.4 of this manual.)

3. Acquiesce the DFS fileset. To accomplish this step, all mount points to the fileset may have to be deleted, and the aggregate detached and re-exported into DFS. This needs to be done so DFS clients cannot modify the fileset during the import.

4. Use **hdmdump** to generate a Nix file containing an ASCII text representation of the DFS fileset. The **SetMigrate** option must be used so that the file data will be migrated from DFS to HPSS after importing the DFS fileset.

If SetMigrate is not used then the HDM will not migrate any of these files into HPSS and the files will remain on Episode disk until they are accessed through the HPSS interfaces.

5. Use SSM to create an HPSS Only fileset which will become the HPSS counterpart to the DFS fileset.

6. Use SSM to change the HPSS Fileset Type to Mirrored.

7. Use **loadhpssfs** to read the Unci file produced in step (4) and load the HPSS fileset. The **loadhpssfs** produces a UNIX ASCII text file containing the mappings between the XDSM ID and HPSS ID. This tool creates a checkpoint / restart file in the current working directory (thus requiring write access to the current working directory) and has to be run as DCE principal `hpss_dmg'. This tool can be restarted if it should fail for some unforeseen problem (e.g. power failure). Refer to Appendix I of this manual for a description of this tool and its restart feature.

8. Use **loadhpssid** to read the UNIX file produced in step (7) and load the HPSS IDs into the DFS fileset objects.

9. Use **create_fsys** to register the DFS aggregate with HDM. (For more information on **create_fsys**, see Section 7.4.2.1.) This tool must be executed on the machine where HDM is running. For this tool to work, HDM must be running and the administrator must have write permission to the HDM configuration file, **filesys.dat**. (**hdm_admin** can be used to verify that HDM is running.)

10. Edit **filesys.dat** to include the new fileset entry for the loaded fileset.

11. Unmount the locally mounted DFS fileset.

12. Use **hdm_admin** to stop and restart HDM so the new changes made it step (10) take effect. This program must be run as root.

13. Use SSM to modify the HPSS Name Server fileset attributes. The Fileset ID and Fileset Name must match the Fileset ID and Fileset Name from the DFS fileset dumped in step (4) above.

14. Since the HPSS Name Server already knows about the HPSS fileset, use SSM to create the corresponding DMG-Only fileset using the "DMG-Only Create" button on the "Create DFS/HPSS Fileset" screen. Refer to the section titled "Creating the DFS fileset" on how to fill in the SSM fields.

Verify that the name used in the "HPSS/DMAP TCP Hostname" field is in the same exact format as the HPSSDMAPHostName parameter in config.dat.

15. Un-acquiesce the DFS fileset to allow DFS clients access to the files.

16. Create an HPSS junction as described in the section titled "Creating the HPSS Junction", so the fileset will be available through the HPSS fileset.

17. If no errors are reported, delete the intermediate files generated by hdmdump, loadhpssfs, and **loadhpssid**.

Example

In the following example an existing DFS non-mirrored fileset named `ProjectX.Fileset' on DFS aggregate `big.aggr' will be mirrored to an HPSS fileset. Following is a list of assumptions:

Aggregate ID	5
Aggregate Mount Path	/opt/dcelocal/var/dfs/aggrs/big.aggr
Aggregate Name	big.aggr
Block Size of Aggregate	8192
Device Name of Aggregate	/dev/big.aggr
DFS Fileset ID	0,,8
Directory to store UNIX files	/WorkingDir
DMAP Gateway Desc Name	DMAP Gateway
Fileset Class of Service	<not specified>
Fileset Family ID	<Project X Family)
Fileset Group	ProjectX (GID 456)
Fileset Name	ProjectX.Fileset
Fileset Owner	FilesetOwner (UID 123)
Fileset Permissions	rwxrwx--- (770)
Fragment Size of Aggregate	1024
Global Mount Point	/:/ProjectX.Fileset
HPSS/NS Fileset ID	2178050145,,814108288
Host's HDM TCP port	6002
Host where DFS & HDM run	ballzooka.llnl.gov
Host where Gateway runs	ripsaw.llnl.gov

Local Mount Point	/var/hpss/hdm/hdm1/aggr/big.aggr/ ProjectX.Fileset
Mount Point Name Server	Name Server

1. Change the DFS fileset type from lfs to dmlfs:

   login to ballzooka.llnl.gov as root

   dfsexport -detach big.aggr
   dmaggr -on big.aggr

   edit /opt/dcelocal/var/dfs/dfstab changing the big.aggr entry type from lfs to dmlfs

   dfsexport big.aggr

2. Mount the DFS fileset locally:

   login to ballzooka.llnl.gov as root

   **mount -v dmlfs -o aggregate=big.aggr -n **
   **ballzooka.llnl.gov ProjectX.Fileset **
   /var/hpss/hdm/hdm1/aggr/big.aggr /ProjectX.Fileset

3. Acquiesce the DFS fileset.

4. Dump the DFS fileset:

   login to ballzooka.llnl.gov as root

   **/usr/lpp/hpss/tools/dmapi/dmapitools/hdmdump/hdmdump **
   **0.8 ProjectX.Fileset **
   **/var/hpss/hdm/hdm1/aggr/big.aggr/ProjectX.Fileset **
   SetMigrate > /WorkingDir/Dump.ProjectX.out

5. Create the HPSS fileset: On the *HPSS Health and Status* screen, select "**Operations**". From that menu, select "**Create HPSS-only Fileset**". The SSM screen titled "Create HPSS-only Fileset" appears. Fill in the following fields:

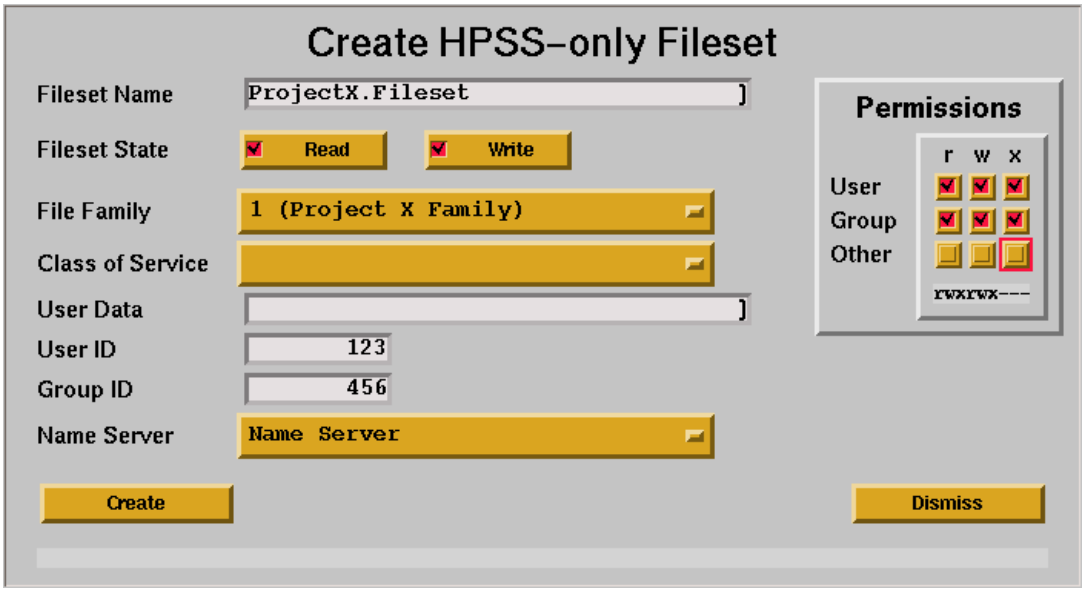


Figure 7-7 Create HPSS-only Fileset

After reviewing the settings on the screen and ensuring correctness, select the "Create" button at the bottom left of the screen. The SSM will send the request to the specified Name Server to create the fileset. If the create is successful, a message will appear at the bottom of the screen, otherwise an error message will be displayed in a separate window.

6. Use SSM to change the HPSS Fileset Type to Mirrored: On the HPSS Health and Status screen, select "Monitor". From that menu, select "HPSS Objects". From that menu, select "Filesets". The SSM screen titled "Identify A Fileset" appears. Enter the fileset ID or fileset name of the HPSS fileset (e.g. ProjectX.Fileset) and select the "Get NS Info" button. The SSM screen, titled "Name Server Fileset Information" appears. Select "Mirrored" from the Fileset Type pop-up list.

7. Load the HPSS fileset:

   dce_login to ripsaw.llnl.gov as hpss_dmg

   ftp ballzooka:/WorkingDir/Dump.ProjectX.out to ripsaw:/WorkingDir
   ksh
   cd /WorkingDir
   . /usr/lpp/hpss/config/hpss_env
   **/usr/lpp/hpss/bin/loadhpssfs ProjectX.Fileset Dump.ProjectX.Fileset.out **
   Load.ProjectX.Fileset.out
   exit # exit from dce_login

8. Load the HPSS IDs into the DFS fileset objects:

   login to ballzooka.llnl.gov
   ftp ripsaw:/WorkingDir/Load.ProjectX.Fileset.out to ballzooka:/WorkingDir

login to ballzooka as root
cd /var/hpss/hdm/hdm1/aggr/big.aggr/ProjectX.Fileset
**/usr/lpp/hpss/tools/dmapi/dmapitools/loadhpssid/loadhpssid **
< /WorkingDir/Load.ProjectX.Fileset.out

9. Register the DFS aggregate with HDM:

   login ballzooka.llnl.gov as root

   **create_fsys ballzooka.llnl.gov 6002 5 **
   **/opt/dcelocal/var/dfs/aggrs/big.aggr **
   /dev/big.aggr mirrored wait wait whole

10. Edit **filesys.dat** to include the new fileset entry for the loaded fileset:

    login ballzooka.llnl.gov as root

    after the big.aggr entry add the line:

    ProjectX.Fileset ? ? 0.8 ? 0

11. Unmount the locally mounted DFS fileset.

    login to ballzooka.llnl.gov as root

    umount /var/hpss/hdm/hdm1/aggr/big.aggr/ProjectX.Fileset

12. Shutdown and restart HDM so the changes in step (10) take effect:

    login ballzooka.llnl.gov as root

    cd /usr/lpp/hpss/bin
    ./hdm_admin <hdm shmkey>
    > stop
    > start

13. Use SSM to modify the HPSS Fileset ID: On the *HPSS Health and Status* screen, select "Monitor". From that menu, select "HPSS Objects". From that menu, select "Filesets". The SSM screen titled "Identify A Fileset" appears. Enter the fileset ID or fileset name (e.g. `ProjectX.Fileset') of the HPSS fileset and select the "Get NS Info" button. The SSM screen, titled "Name Server Fileset Information" appears. Enter the DFS Fileset ID (e.g. 0,,8). Both fields of the number must be entered. Click on the update button to the right of the new Fileset ID. (If you should get an error, try first changing the Fileset Type to `HPSS-only', then change the Fileset ID, and then set the Fileset Type back to `Mirrored'.)

14. Use SSM to create the corresponding DMG-Only fileset: On the HPSS Health and Status screen, select "Operations". From that menu, select "Create DFS / HPSS Fileset". The SSM screen titled "Create DFS / HPSS Fileset" appears. Fill in the following fields:

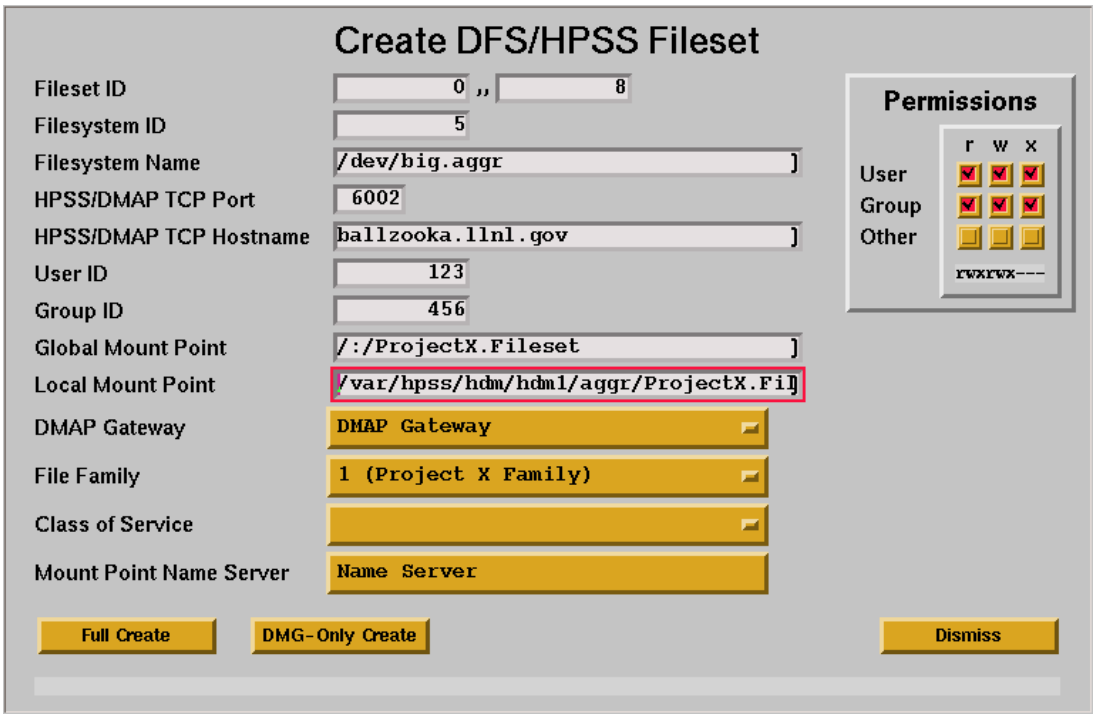


Figure 7-8 Create DFS / HPSS Fileset (DMG-Only Create 2)

After reviewing the settings on the screen and ensuring correctness, select the "DMG-Only Create" button at the bottom left of the screen. The SSM will send the request to the specified Name Server to create the fileset. Upon successful completion, a message will appear at the bottom of the screen, otherwise an error message will be displayed in a separate window.

Verify that the name used in the "HPSS/DMAP TCP Hostname" field is in the same exact format as the HPSSDMAPHostName parameter in config.dat.

15. Allow outside access to the DFS fileset.

16. Create a junction to the HPSS fileset: On the *HPSS Health and Status* screen, select "**Operations**". From that menu, select "**Create Junction**". The SSM screen, titled "Create Junction" appears. (Care must be taken when choosing the path name of a junction to a mirrored fileset. The path name to the HPSS fileset must be identical to the DFS mount point.) Fill in the following fields:

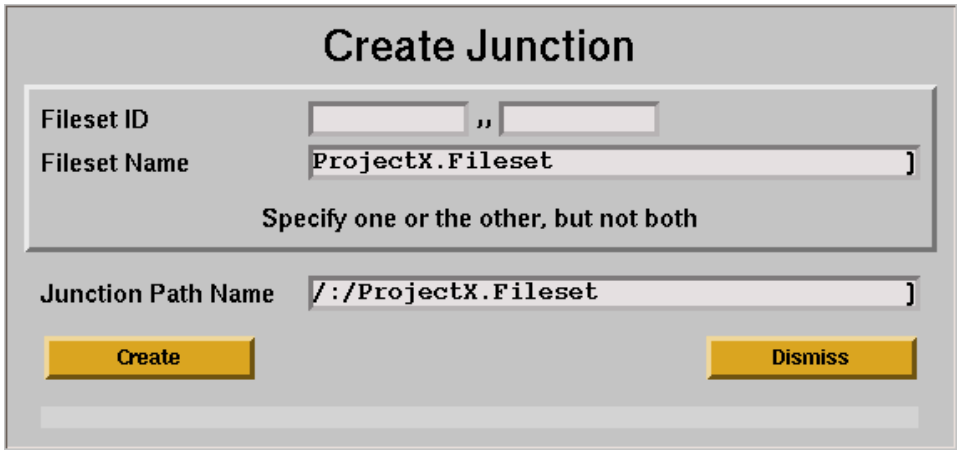


Figure 7-9 Create Junction 2

After reviewing the settings on the screen and ensuring correctness, select the "**Create**" button at the bottom left of the screen. If the create is successful, a message will appear at the bottom of the screen, otherwise an error message will be displayed in a separate window.

17. Delete the intermediate files:

login to ripsaw

cd /WorkingDir
rm Dump.ProjectX.Fileset.out
rm Load.ProjectX.Fileset.out
rm LoadHPSSFSRestartFile
login to ballzooka
rm /WorkingDir/Load.ProjectX.Fileset.out

7.4.7 Mirrored Fileset Management Tools

7.4.7.1 dumphpssfs

The tool, **dumphpssfs**, rumbles through an HPSS fileset and generates a UNIX file containing an ASCII representation of the fileset name space. The resulting file can be used by another utility to generate the same fileset on another system.

dumphpssfs can be found in **/usr/lpp/hpss/bin/dumphpssfs**. To run this tool, the administrator must be logged into DCE as the DMG principal and, since this tool loads with the Client API library, must have sourced **/usr/lpp/hpss/config/hpss_env** which an HPSS System Administrator has set up (see the *HPSS Programmer's Reference Guide* for more information on the Client API environment). The syntax for the **dumphpssfs** command is:

dumphpssfs <Fileset Name> <Output File> [-c] [-r]

where

<Fileset Name>	The name of the HPSS Fileset to be dumped.
<Output File>	The name of the UNIX file where the ASCII dump will be written.
-c	Optional command indicating to dump the consistency flags. These flags describe synchronization states between DFS and HPSS. The default is not to include the consistency flags. Dumping these flags will degrade performance and is not required by **loadtree**. Dumping the consistency flags may be useful on an already mirrored fileset for synchronization verification.
-r	Optional command indicating to restart the dump where the previous dump terminated. This option may be used if a previous run of **dumphpssfs** terminated prematurely. Note: The restart option assumes the fileset being dumped has NOT been altered since the last dump attempt; unpredictable results may occur if the fileset has been altered.

See Appendix I of the *HPSS Administration Guide* for more details on **dumphpssfs**.

7.4.7.2 hdmdump

The tool, **hdmdump**, traverses the DFS fileset, and generates a UNIX file containing an ASCII representation of the fileset name space. The resulting file can be used by another utility (**loadhpssfs**) to import this fileset into HPSS. The number of records in the output file will match the number of objects contained in the DFS fileset. The type of the aggregate that contains the fileset must be **dmlfs**.

hdmdump can be found in **/usr/lpp/hpss/tools/dmapi/hdmdump**. The syntax for the **hdmdump** command is:

hdmdump <Fileset ID> <Fileset Name> <Root of Fileset> [SetMigrate] > [Output File]

where

<Fileset ID>	The fileset ID of the DFS fileset to be dumped. The format of this ID is: [num].[num] (e.g., 0.4 or 0.22).
<Fileset Name>	The name of the DFS fileset to be dumped.
<Root of Fileset>	The path to the root of the DFS fileset.
SetMigrate	Optional command that marks the files in DFS that the data needs to be migrated to HPSS.
Output File	The name of the UNIX file to place the ASCII dump into.

See Appendix I of the *HPSS Administration Guide* for more details on **hdmdump**.

7.4.7.3 loadhpssdmid

The tool, **loadhpssdmid**, uses an ASCII text file (produced by **loadtree**), and sets the XDSM ID in each corresponding object in the HPSS Fileset.

loadhpssdmid can be found in **/usr/lpp/hpss/bin/loadhpssdmid**. To run this tool, the administrator must be logged into DCE as the DMG principal and, since this tool loads with the Client API library, must have sourced **/usr/lpp/hpss/config/hpss_env** which an HPSS System Administrator has set up (see the *HPSS Programmer's Reference Guide* for more information on the Client API environment). The syntax for the **loadhpssdmid** command is:

loadhpssdmid [-r] < [Input File]

where

-r	Optional command indicating this is a recovery of a fileset rather than an initial import. If a recovery is being performed, then the -r option should be used.
Input File	The name of the UNIX file containing information to be loaded into the HPSS fileset.

See Appendix I of the *HPSS Administration Guide* for more details on **loadhpssdmid**.

7.4.7.4 loadhpssfs

The tool, **loadhpssfs**, reads a UNIX ASCII text file generated by hdmdump and loads the objects into an HPSS mirrored fileset.

loadhpssfs can be found in **/usr/lpp/hpss/bin/loadhpssfs**. To run this tool, the administrator must be logged into DCE as the DMG principal and, since this tool loads with the Client API library, must have sourced **/usr/lpp/hpss/config/hpss_env** which an HPSS System Administrator has set up (see the *HPSS Programmer's Reference Guide* for more information on the Client API environment). The syntax for the **loadhpssfs** command is:

loadhpssfs <Fileset Name> <Input File> <Output File> [-r]

where

<Fileset Name>	The name of the HPSS Mirrored Fileset to load.
<Input File>	The name of the UNIX file containing the ASCII dump text to be loaded into the fileset.
<Output File>	The name of the UNIX file that will contain the Output text produced by **loadhpssfs**.
-r	Optional command indicating the user wishes to restart the load where the previous load left off. This option may be used if a previous run of **loadhpssfs** terminated prematurely. Note: The restart option assumes that the fileset being loaded has NOT been altered since the last load attempt. Unpredictable results may occur if the fileset has been altered.

See Appendix I of the *HPSS Administration Guide* for more details on **loadhpssfs**.

7.4.7.5 loadhpssid

The tool, **loadhpssid**, rumbles through an ASCII text file (produced by **loadhpssfs**), and sets the HPSS ID in each corresponding object in the DFS fileset.

loadhpssid can be found in **/usr/lpp/hpss/tools/dmapi/loadhpssid**. The syntax for the **loadhpssid** command is:

loadhpssid < [Input File]

where

Input File	The name of the UNIX file containing information to be loaded into the DFS fileset.

See Appendix I of the *HPSS Administration Guide* for more details on **loadhpssid**.

7.4.7.6 loadtree

The tool, **loadtree**, reads a UNIX ASCII text file generated by **dumphpssfs** and loads the objects into a DFS fileset.

loadtree can be found in **/usr/lpp/hpss/tools/dmapi/dmapitools/loadtree**. The syntax for the **loadtree** command is:

loadtree < [Input File] > [Output File]

where

Input File	The name of the UNIX file containing the ASCII data generated by **dumphpssfs** to be loaded into the DFS fileset.
Output File	The name of the UNIX file containing the ASCII data which will be input to **loadhpssdmid**. This file will contain DM Handle to HPSS ID mappings.

See Appendix I of the *HPSS Administration Guide* for more details on **loadtree**.

Chapter 8

HPSS Special Intervention Procedures

8.0.0.1 Overview

This chapter provides instructions and supporting information for special operations that are not considered usual or regularly-performed procedures for HPSS. Refer to Chapter 6 for information on day-to-day operations and regular management procedures.

The procedures described in this chapter are for use while HPSS is up and operational. Most of the procedures assume that a failure has occurred and that special intervention is required.

8.1 Manually Dismounting Tape Drives

HPSS provides the ability to manually dismount a tape drive in unusual situations, such as when an administrator wants to dismount a drive without waiting for HPSS to do so, or when HPSS is unable to do so due to a failure.

Manual dismount of a drive should only occur in exceptional circumstances (hardware/software error cases). Dismounting a drive/tape from HPSS will cause the operations being performed on that drive/tape combination to fail.

Using the Device/Drive List Window

From the HPSS Devices and Drives window shown in Figure 8-1, select the desired device/drive entry and click on the **Dismount** button. A confirmation window will pop up requiring you to confirm the dismount request. Click on the **Confirm** button to confirm the request.

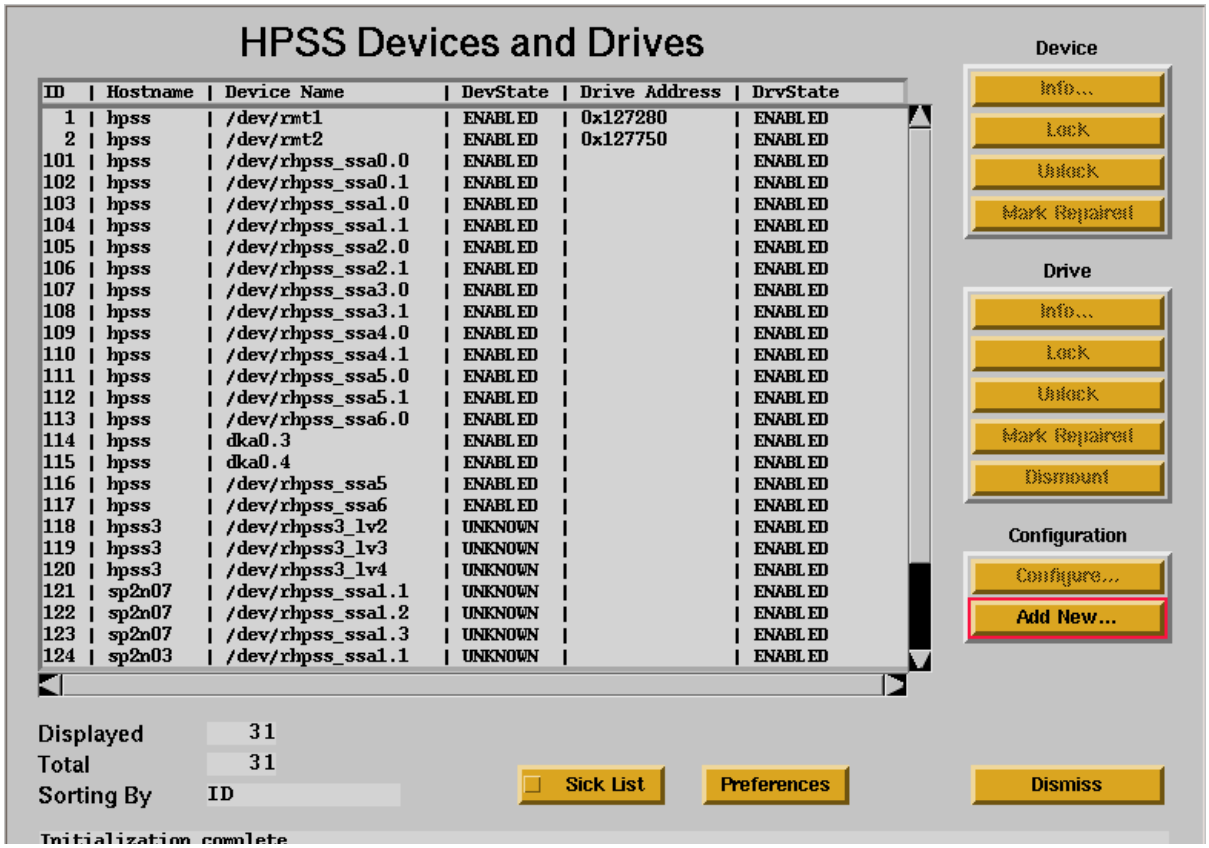


Figure 8-1 Device / Drive List Window

8.2 Canceling Queued PVL Requests

The intent of canceling a queued PVL request is to cancel a request that will never be able to complete (due to hardware or software error) or to cancel a request to help the overall performance of the HPSS system.

With the exception of canceling a deferred dismount job, cancellation of a queued PVL request will cause all operations associated with that request to fail. Cancellation should therefore be done only after careful consideration of its consequences.

When a deferred dismount job is cancelled, the PVL issues dismount requests to all drives loaded with cartridges in deferred dismount state. This will result in the affected cartridges being physically dismounted from drives. Canceling a deferred dismount job will not result in an error propagated through the HPSS servers.

Using the PVL Job Queue Window

From the PVL Job Queue window shown in Figure 8-2, select the desired job entry and click on the **Cancel Job** button. A confirmation window will pop up requiring you to confirm the cancel request. Click on the **Confirm** button to confirm the request.

Canceling a PVL job should be done with great care. The Tape Storage Server reserves the device a cartridge is mounted on while reading/writing. Canceling the PVL job which represents the mounted cartridge will result in a dismount request issued to the reserved device. The PVL generated dismount request (issued periodically) will fail (generating an alarm) until the Tape Storage Server releases its reservation of the device. Canceling a PVL job will NOT result in the termination of Storage Server request and the cartridge dismount will NOT occur until the Tape Storage Server releases its reservation on the device.

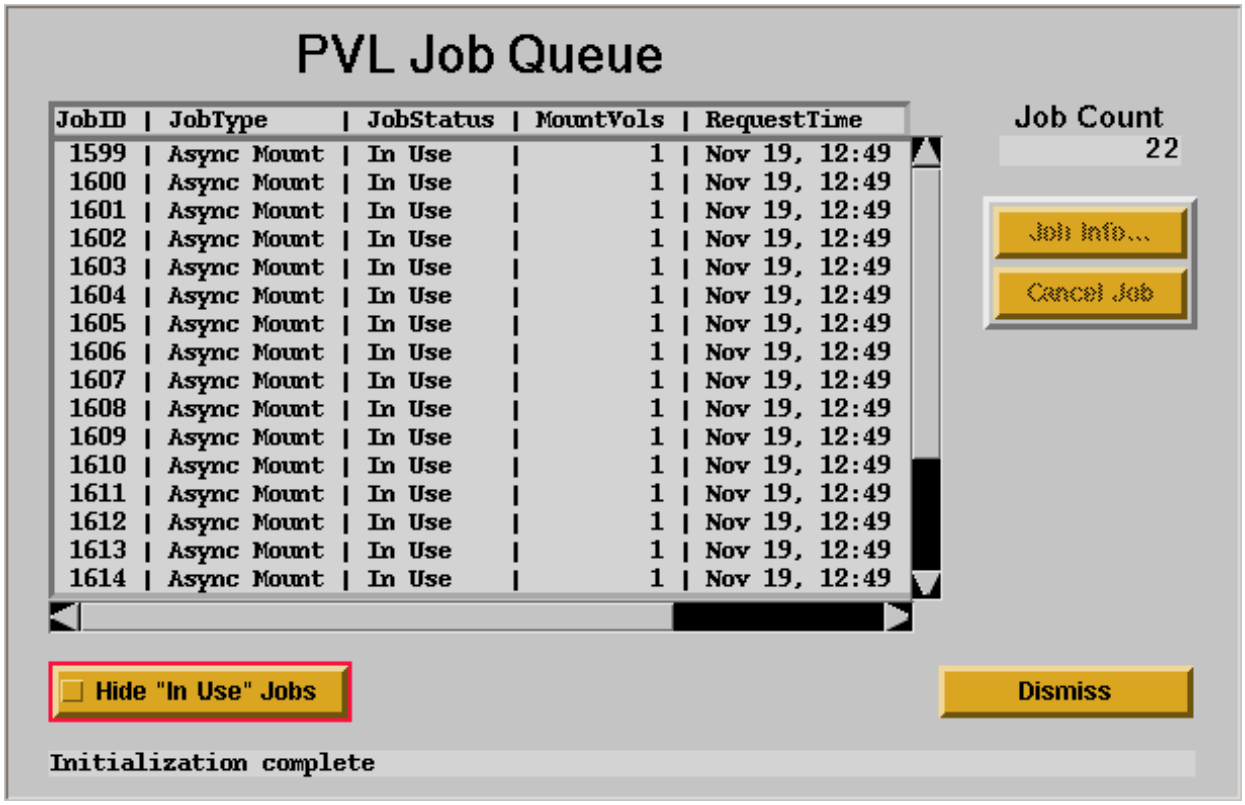


Figure 8-2 PVL Job Queue Window

8.3 Changing the COS of a File

The intent of changing the COS of a file is to move the file to a more appropriate COS or to correct an improper placement of a file in HPSS. The **scrub** utility can be used to initiate the COS change request of a file as follows:

$ dce_login <principle>

$ scrub

$ scrub> changecos <fullPathname> <newCOSid>

$ scrub> quit

A BFS COS change record will be created. It will be processed later by the BFS COS change thread.

8.4 Canceling a COS Change Request for a File

The intent of canceling a COS change request for a file is to rescind a COS change request that failed repeatedly due to lack of resources or other system problems. The **scrub** utility can be used to cancel the COS change request as follows:

$ dce_login <principle>

$ scrub

$ scrub> changecos <fullPathname> 0

$ scrub> quit

The associated BFS COS change record will be removed immediately. However, this request may fail if the existing COS change request is currently being processed by the BFS COS change thread. In this case, the user should attempt to reissue the request at a later time.

8.5 Moving PVR Cartridges

The intent of the Move Cartridge operation is to notify HPSS that a cartridge has been moved from one robot to another.

Note that this operation will not cause a cartridge to move. The operation will fail if the cartridge has not been physically removed from its original robot and injected into the new robot. See the operator manual(s) for the specific robots involved to determine the procedures for manually ejecting and injecting a cartridge. Either the source or destination PVR (or both) during a move operation may be an operator mounted set of drives instead of a robot.

Using the Move Cartridge Window

From the HPSS Health and Status window (Chapter 6, Figure 6-1) click on the Operations menu and select the **Move Cartridge** option. The Move Cartridge window will be displayed as shown in Figure 8-3.

Enter the name of the cartridge to be moved. Select the destination PVR from the pop-up option list. The cartridge should already have been physically moved to that PVR. Click on the **Move**

button when ready. HPSS will verify that the cartridge is in the new PVR and the operation will complete.

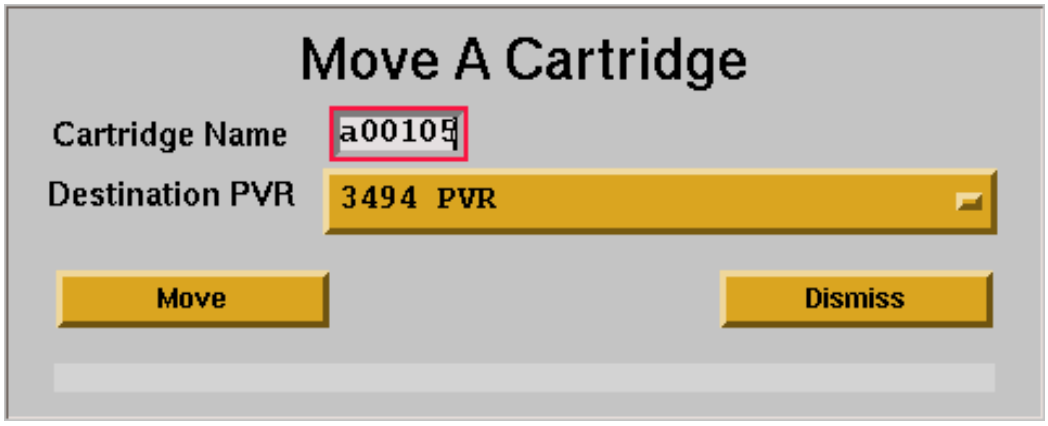


Figure 8-3 Move Cartridge Window

8.6 HPSS Data Recovery

In some situations, an HPSS tape or disk volume may become damaged such that the user data stored on the volume cannot be retrieved by the HPSS servers. Similarly, an SFS volume may become damaged such that the HPSS metadata stored on it becomes inaccessible to HPSS. The following paragraphs describe how to recover the data from the damaged media.

8.6.1 Recovering HPSS Files from Damaged HPSS Volumes

Occasionally, a tape or disk volume is damaged and HPSS will experience errors while trying to read data stored on the damaged volume.

The following steps must be done to determine whether a volume is actually damaged and, if there is, to prepare and to perform the recovery process:

1. **Determine the ID of the potentially damaged volume.**

   Attempts to read the damaged volume will result in Mover alarm messages being issued to SSM. The alarm messages will contain the ID of the physical volume for which the error occurred. Record the volume ID.

2. **Determine whether the volume is actually damaged.**

   Typically, an alarm message from the Mover when it encounters a damaged volume would indicate the source of the problem. The system administrator should check the block size of the tape device on which the error occurred. HPSS tape devices should be configured with variable block sizes (see Section 5.7). When satisfied that the problems are not configuration related, the system administrator should first try to repack the suspect volume (see the **repack** man page in Section I.49 for instructions on running **repack** from the command line). If after several attempts, all segments are not repacked, continue with the recovery procedures outlined in this section.

   I*f the first repack attempt fails, it may be worthwhile to retry the repack several times before running the recover utility.This is especially true if copies aren't available for the storage level that contains the suspect volumes. There are a number of intermittent or environmental errors that may cause an attempt to repack a volume (especially a tape volume) to fail, while a subsequent attempt may succeed (e.g., a failing drive, problems reading a volume in a particular drive, a drive that requires cleaning).*

3. **Make the damaged volume completely unavailable.**

   From the Health and Status window (Figure 6-1), click on the **Monitor** menu, select the **HPSS Objects** and then the **Cartridges and Volumes** options to bring up the Identify Cartridges and Volumes window (Figure 6-30). From this window, enter the damaged Volume ID then click on the **SS Virtual Volume** button to bring up the associated Virtual Volume Information window (Figure 6-34). The **Physical Volumes** field on this window identifies all physical volumes which made up the associated virtual volume. These physical volumes and the data stored on them will be involved in the recovery process.

   To take a VV completely off-line so that it will not be read, written or mounted until the administrator is ready to perform the data recovery, set the **Administrative State** field in the Virtual Volume Information window to **Locked**.

4. **Determine the storage class ID of the damaged volume.**

   From the Identify Cartridges and Volumes window(Figure 6-30), click on the **SS Storage Map** button to bring up the associated Storage Map Information window (Figure 6-35 or Figure 6-36). Note the storage class name to which the volume belongs. Use the Storage Class List window (Figure 6-9) to determine the storage class ID.

   To prevent new data from being written to the tape virtual volume while the recovery is in progress, set the **Map State** field in the Storage Map Information window to **EOM**. For disk virtual volume, lock the storage map.

5. **Recover the data on the damaged virtual volume.**

   From the Virtual Volume Information window (Figure 6-34), set the **Administrative State** field to **Unlocked.**

   The system administrator can now use the **recover** utility to recover HPSS data from a secondary copy. If no secondary copy exists, the recover utility can be used to clean up storage resources and report what data has been lost. The **recover** utility has several options (see its man page in Section I.47). These options are used depending on the severity of the damage. Refer to Sections 8.6.1.1, 8.6.1.2, 8.6.1.3 for more information on how and when to use these options.

8.6.1.1 Recover Partially Damaged Disk or Tape Volume

In this situation the tape or disk can be mounted and the volume label can be read successfully by the Mover, but the media can not be repacked due to segment move errors. In this case, **recover** should be run in the mode where it first attempts a repack of the specified volume, and then recovers any segments not repacked from a secondary copy (if one exists).

With Secondary Copies

The **recover** utility is executed with the names of one or more HPSS physical volumes. These volumes must be in the same storage class. The virtual volumes that contain the suspect physical volumes are identified and all the storage segments on those virtual volumes are moved to other virtual volumes. If any segment moves fail, all of that bitfile's storage segments are purged from the primary storage level and recovered from the secondary level.

For example, to run a recovery of two volumes (VOL00100 and VOL00200) that are members of a primary copy storage class (storage class ID 3), enter:

```
recover -r -c 3 VOL0010 VOL0020
```

The **recover** utility logs status messages (see the recover manual page for status log information) as it tries to recover damaged segments. The messages should be similar to the following:

```
========= Trying to recover bitfile =========
003fcadc-53fb-10cf-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/file1)
========= Trying to recover bitfile =========
163001bc-2274-10ef-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/file2)
========= Trying to recover bitfile =========
0786ab2c-156b-1047-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/file3)
========= Trying to recover bitfile =========
.
.
.

========= Deleting VV =========
0723ab2c-176b-1047-8c81-02608c2f971f
00226b52 4631 10cf 00 00 00 17
========= Deleting VV =========
0456ab2c-176b-1047-8c81-02608c2f971f
00226b52 4631 10cf 00 00 00 17
```

At the end of the recovery, all virtual and physical volume metadata associated with the damaged tape volumes will be deleted. All physical volumes which are part of the virtual volume will be exported from the PVL. The damaged volume(s) can be thrown away and the others imported back into HPSS to be reused.

Without Secondary Copies

When no secondary copy of the data exists, the system administrator can use the **recover** utility to clean up storage resources and report what data has been lost. In this case, the system administrator should make every attempt to repack the damaged volumes.

If all attempts to repack the damaged volumes fail and there are no secondary copies, then any file segments on the damaged areas of the bad volumes are lost. The **recover** utility should be executed again, this time to retry the repack, and then delete storage resources that refer to the lost storage segments. To begin the cleanup, enter:

```
recover -r -c 3 VOL0010 VOL0020
```

Status messages and lost segments are logged as metadata for irrecoverable segments are deleted. The messages should be similar to the following:

```
========= Can't recover bitfile =========
003fcadc-53fb-10cf-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/file1)
lost segments from storage level 1
offset = 0    , length = 32768
offset = 32768, length = 32768
offset = 65536, length = 32768
.
.
========= Can't recover bitfile =========
163001bc-2274-10ef-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/file2)
lost segments from storage level 1
offset = 0    , length = 32768
offset = 32768, length = 32768
offset = 65536, length = 32768
.
.
.

========= Deleting VV =========
0723ab2c-176b-1047-8c81-02608c2f971f
00226b52 4631 10cf 00 00 00 17
========= Deleting VV =========
0456ab2c-176b-1047-8c81-02608c2f971f
00226b52 4631 10cf 00 00 00 17
```

At the end of the recovery, all virtual and physical volume metadata associated with the damaged tape volumes will be deleted. All the physical volumes which are part of the virtual volume will be exported from the PVL. The damaged volume(s) can be thrown away and the others imported back into HPSS to be reused.

8.6.1.2 Recover Totally Damaged Tape

In this situation a tape cartridge or label is damaged, preventing any attempt to repack the volume. To handle this situation **recover** should be run in a mode where it doesn't try to repack the specified volume. Running in this mode, **recover** simply purges all the segments on the specified volume and then tries to recover the volume data from a secondary copy.

With Secondary Copies

To run a recovery of two volumes (VOL00100 and VOL00200) that are members of a copy storage class, enter:

```
recover VOL0010 VOL0020
```

The **recover** command logs status messages as it tries to recover damaged segments. The messages should be similar to the following:

```
========= Trying to recover bitfile =========
003fcadc-53fb-10cf-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/file1)
========= Trying to recover bitfile =========
163001bc-2274-10ef-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/file2)
========= Trying to recover bitfile =========
0786ab2c-156b-1047-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/file3)
========= Trying to recover bitfile =========
.
.
.

========= Deleting VV =========
0723ab2c-176b-1047-8c81-02608c2f971f
00226b52 4631 10cf 00 00 00 17
========= Deleting VV =========
0456ab2c-176b-1047-8c81-02608c2f971f
00226b52 4631 10cf 00 00 00 17
```

At the end of the recovery, all virtual and physical volume metadata associated with the damaged tape volumes will be deleted. All physical volumes which are part of the virtual volume will be exported from the PVL. The damaged volume(s) can be thrown away and the others imported back into HPSS to be reused.

Without Secondary Copies

When no secondary copy of the data exists, the system administrator can use the **recover** command to clean up storage resources and report what data has been lost. In this case, the system administrator should make every attempt to repack the damaged volumes.

To begin cleanup for two volumes (VOL00100 and VOL00200) that are not members of a copy storage class, enter:

```
recover VOL0010 VOL0020
```

The **recover** command logs status messages as metadata for irrecoverable segments are deleted. The messages should be similar to the following:

```
========= Trying to recover bitfile =========
003fcadc-53fb-10cf-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/file1)
========= Trying to recover bitfile =========
163001bc-2274-10ef-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/file2)
========= Trying to recover bitfile =========
0786ab2c-156b-1047-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/file3)
========= Trying to recover bitfile =========
.
.
.

========= Deleting VV =========
0723ab2c-176b-1047-8c81-02608c2f971f
00226b52 4631 10cf 00 00 00 17
========= Deleting VV =========
0456ab2c-176b-1047-8c81-02608c2f971f
00226b52 4631 10cf 00 00 00 17
```

At the end of the recovery, all virtual and physical volume metadata associated with the damaged tape volumes will be deleted. All physical volumes which are part of the virtual volume will be exported from the PVL. The damaged volume(s) can be thrown away and the others imported back into HPSS to be reused.

8.6.1.3 Recover Totally Damaged Disk

Imagine a situation where you have a disk over tape hierarchy, all files have been migrated to tape at some point, and one or more of your disks have been unrecoverably damaged or inadvertently reformatted. In this case you would want to clean up all storage resources that point to the damaged volume(s) without even attempting a repack. To perform this type of clean up procedure, the **recover** utility can be executed with the "-x" option. Running in this mode **recover** removes all storage resources associated with the specified storage class and volume. To remove all storage resources associated with volume LV000001, enter:

```
recover -x LV000001
```

The output displayed when **recover** is running in this mode should consist of notification of lost bitfile segments. The messages should be similar to the following:

```
========= Cleaning up segments for bitfile =========
003fcadc-53fb-10cf-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/file1 )
lost segments from storage level 0
this file has been migrated since last update!
========= Cleaning up segments for bitfile =========
163001bc-2274-10ef-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/file2 )
lost segments from storage level 0
This file has NOT been migrated since last update!
========= Cleaning up segments for bitfile =========
.
.
.

========= Deleting VV =========
0723ab2c-176b-1047-8c81-02608c2f971f
00226b52 4631 10cf 00 00 00 17
```

At the end of the cleanup, all the virtual and physical volumes associated with the targeted volumes will be deleted. All the physical volumes contained in the virtual volume will be exported from the PVL. The media can then be imported back into HPSS to be reused. At this point, any of the files once residing on the damaged disk volume can be stage from tape and accessed.

Notice that the output contains indications of whether each file on disk was successfully migrated to the next storage level. In the example above, file1 was migrated successfully and can be staged back to the disk level after cleanup. The other file (file2) was not migrated successfully and may contain invalid data after the cleanup.

The recover utility will not remove any name entries. If the file is to be removed from HPSS entirely, one of the other tools (such as **scrub**, see Section I.51) should be used for that purpose.

8.6.2 *Recovering HPSS Metadata from Damaged Encina SFS Volumes*

Important Note: Failure to follow appropriate procedures in attempting to restore damaged SFS volume data could result in the loss of additional data. Consult the Encina Server Administration: System Administrator's Guide and Reference for AIX and the Encina Structured File Server Administrator's Guide and Reference for AIX for the version of Encina being used to ensure the proper recovery procedures are followed. Given the condition of having damaged HPSS metadata and the severity of this situation, it would be appropriate to contact your vendor's Encina technical support organization and have them assist you in properly recovering from this situation.

This section assumes that Encina for IBM AIX platforms is being used. References to suggested commands and procedures may not apply to other platforms. It is also assumed that media archiving has been enabled for the SFS server for which a volume needs to be recovered.

The need to recover HPSS metadata in SFS could result from an application failure, a system hardware crash, or some type of media failure. Encina will recover on its own under the first two

conditions mentioned (application or system crashes) by using data in the transaction log it maintains to restore all previously-committed metadata (SFS data).

Administrative intervention is required to handle the recovery condition resulting from media failure. If the volume on which the media error occurred has been mirrored and there is still at least one good copy of the data available, the data in both copies will be synchronized automatically when the volume groups are activated (via the **varyonvg** command). If it is necessary to replace the associated physical disk(s), the proper steps should be followed to bring the physical volume into the original volume group and have it synchronized with the contents of the "good" volume in which it mirrors. Refer to "How To Recover from Disk Drive Problems" in the AIX System Administration documentation (*IBM AIX Version 4.2 for RISC System/6000 Installation Guide* or *IBM AIX Version 4.2 for RISC System/6000 Installation Guide*) for the specific steps and commands to use in this recovery process.

When no "good" copy of the logical volume is available, having media archiving enabled allows you to use the **tkadmin restore** command to restore your volumes. The following paragraphs discuss recovering an Encina SFS volume and recovering an Encina SFS log volume.

8.6.2.1 Recovering an Encina SFS Data Volume

The following information has been extracted from Encina Overview, an IBM ITSO Redbook (GG24-2512-00).

Perform the following steps to recover an Encina SFS data volume when no "good" copy of the data volume exists:

1. **Recreate the lost logical volume if necessary.** If the error is due to some bad blocks on the physical volume, bad block relocation should normally relocate the blocks transparently, provided bad block relocation has not been disabled. If bad block relocation is not possible, you might have to recreate the logical volume at a different location. If the physical volume itself is somehow defective, or unreliable, you should recreate the logical volume on a different physical volume.

2. **Start SFS in administrative mode.** View the **rc.encina** file to find the argument values used when starting SFS. The precise directory path for the location of the **rc.encina** file depends on the name of the SFS server. For example, the default path is **/opt/encinalocal/encina/sfs/hpss/rc.encina** for server **/.:/encina/sfs/hpss**. Start SFS manually with these arguments, along with the "**-a**" and "**-A encina_admin**" flags to start it in administrative mode.

3. **Make the backup files available on-line.** Copy the backup files (TRB and MRA) from off-line to on-line storage. The backup files should be copied into the directory expected by the server. This is the directory specified in the output of the **tkadmin query backup** command. Not all files have to be restored online at the same time. Restore the first set of files and SFS will prompt you when to restore the next set of files.

4. **Restore the volume.** Issue the **tkadmin restore lvols** command to restore the volume. For example, to restore the **sfsVol1** data volume for SFS server **/.:/encina/sfs/hpss** using backup file number 8, enter:

   % tkadmin restore lvols -server /.:/encina/sfs/hpss 1 sfsVol1 -backupfilenum 8

If multiple volumes must be restored, it is generally better to restore them at the same time rather than separately. The above tkadmin command allows more than one volume to be specified.

5. **Wait for the restore to finish.** Monitor the SFS output file for prompting of any required MRA files. The progress of the restore operation can be monitor as follows:

   % tkadmin query restore -server /.:/encina/sfs/hpss

After the restore operation completes, the current state of your data is that from your last backup.

6. **Recover the volume. To recover all volumes on the /.:/encina/sfs/hpss server, enter:**

   % tkadmin recover lvols -server /.:/encina/sfs/hpss

This step will use the data in the restored log volumes to bring the system to the state at the last commit.

7. **Switch SFS back to normal mode.** To stop and restart SFS in normal mode, enter:

   % tkadmin stop server

   % rc.encina

8.6.2.2 Recovering an Encina SFS Log Volume

The following information has been extracted from Encina Overview, an IBM ITSO Redbook (GG24-2512-00).

Perform the following steps to recover an Encina SFS log volume when no "good" copy of the log volume exists:

1. **Recreate the lost logical volume if necessary.** If the error is due to some bad blocks on the physical volume, bad block relocation should normally relocate the blocks transparently, provided bad block relocation has not been disabled. If bad block relocation is not possible, you might have to recreate the logical volume at a different location. If the physical volume itself is unreliable or somehow defective, recreate the logical volume on a different physical volume.

2. **Start SFS in administrative mode.** View the **rc.encina** file to find the argument values used when starting SFS. The precise directory path for the location of the **rc.encina** file depends on the name of the SFS server. For example, the default path is **/opt/encinalocal/encina/sfs/hpss/rc.encina** for server **/.:/encina/sfs/hpss**. Start SFS manually with these arguments, along with the "**-a**" and "**-A encina_admin**" flags to start it in administrative mode.

3. **Make the backup files available on-line.** Copy the log archive (LA) files from off-line to on-line storage to the archive directory for the server (usually **/opt/encinalocal/encina/sfs/hpss/logArchives**). This can be done incrementally.

4. **Restore the volume.** Issue the **tkadmin restore logvol** command to restore the log volume. For example, to restore the **logVol1** log volume for SFS server **/.:/encina/sfs/hpss** using archive file number 12 (use the latest number available) located in the **/archives** directory, enter:

% tkadmin restore logvol -server /.:/encina/sfs/hpss logVol1 /archives logVol1.LA.12

5. **Wait for the restore to finish.** Monitor the SFS output file for prompting of any required MRA files. The progress of the restore operation can be monitor as follows:

   % tkadmin query restore -server /.:/encina/sfs/hpss

6. **Enable and restore the volume.** To enable and restored the log volume, enter:

   % tkadmin enable logfile logVol1/logFile

7. **Restore** *all* **sfs data volumes, enter:**

   For systems with only a single volume:

   % tkadmin restore lvols -server /.:/encina/sfs/hpss 1 sfsVol1

   % tkadmin recover lvols -server /.:/encina/sfs/hpss

   For systems with *n* volumes:

   % tkadmin restore lvols -server /.:/encina/sfs/hpss *n* **sfsVol1 sfsVol2 ... sfsVol***n*

   % tkadmin recover lvols -server /.:/encina/sfs/hpss

This step will use the data in the restored log volume to bring the system to the state at the last commit.

8. **Switch SFS back to normal mode.** To stop and restart SFS in normal mode, enter:

   % tkadmin stop server

   % rc.encina

8.7 Handling HPSS Space Shortage

Over time, due to insufficient space allocation or unanticipated growth, the free HPSS storage space, SFS space, or NS name space may run low. When this occurs, if the space thresholds were configured for these entities, HPSS will issue the appropriate threshold-exceeded alarms to notify the SSM user. The following sections discuss how to handle the appropriate space shortage problem.

8.7.1 HPSS Storage Space Shortage

If free storage space in HPSS is not sufficient, HPSS will warn the administrator if the storage class thresholds have been set up. Responding to this situation will require analyzing why the shortage is being experienced. In many cases, the appropriate action is to add additional storage resources to the affected storage classes. If the shortage is being experienced in a storage class that can be migrated, either more space needs to be added to this storage class or the migration and purge parameters need to be adjusted. Refer to Section 6.4.1.2 for more information on creating additional

storage space. Refer to Section 5.4.1 and Section 5.4.2 for more information on adjusting the migration and purge policies.

8.7.2 SFS Space Shortage

Before executing any of the suggestions or steps listed in this section, consult the appropriate Encina Server Administration: System Administrator's Guide and Reference for AIX and the Encina Structured File Server Administrator's Guide and Reference for AIX for the appropriate version of Encina. It would also be wise to verify any significant changes to, or reallocation of, production SFS space through the Encina technical support channel.

Once an AIX or Encina volume is expanded to a new size, it can never be reduced in size.

If the amount of SFS free space begins to run low on a particular volume, perform the following steps to increase the size of the volume:

1. **Expand the size of the AIX logical volume.** The AIX logical volume associated with the SFS data volume must be expanded using the appropriate operating-system utility (e.g., using the **extendlv** command).

2. **Expand the size of the Encina data volume.** Once the size of the AIX logical volume has been expanded, SFS needs to be instructed to increase the size of the corresponding data volume. This can be done using the **tkadmin expand lvol** command.

   Before using this command, first determine the new total size of the AIX logical volume. Then determine how many 4KB pages will fit on the volume (AIX logical volume size in bytes divided by 4096) and subtract a couple of pages for AIX overhead. Assuming the new number of pages is 5000, issue the following command:

   % tkadmin expand lvol -server /.:/encina/sfs/hpss sfsVol1 5000

   The preceding command expands the **sfsVol1** volume. Its new size is now 5000 pages, which is equivalent to 20,000KB.

If the particular AIX logical volume cannot be extended further or the volume is the maximum size supported by Encina, consider the following options:

1. **Move one or more SFS files to another volume.** If multiple HPSS metadata files are allocated to the volume that is low on space, a new volume can be created and one or more of the files can be moved to that new volume, freeing up space on the original volume.

   Before starting this effort, HPSS should be brought down. A complete data dump (e.g., using the **dd** command) of the associated logical volume should be done. Use the **sfsadmin export file** and **sfsadmin import file** commands to export and subsequently import the contents of the old file into the new file. On the import command, specify a new, temporary file name as well as the new target volume name. Refer to the Encina documentation for instructions on how to do this.

   After verifying the export and import operations were successful, delete the old SFS file (e.g., **sspv**) and rename the new file to the original file name (e.g., **rename sspv.new to sspv**).

For site with large metadata, SFS import and export operation can take a long time and should be avoided.

2. **Move one or more SFS files to another SFS server.** For performance and/or disk availability reasons, it may be necessary to relocate an SFS file to another node. In this case, the HPSS installation tools should be used to create a new SFS server on a different machine.

   Before starting this effort, HPSS should be brought down. A complete data dump (e.g., using the **dd** command) of the associated logical volume should be done. Use the **sfsadmin export file** and **sfsadmin import file** commands to export and subsequently import the contents of the file on the original SFS server into a new file on the new SFS server. Refer to the Encina documentation for instructions on how to do this.

3. **Move one or more secondary keys to a different volume.** Normally, **managesfs**, the utility used to create HPSS metadata files in SFS, allocates all data and secondary keys for a given file onto the same volume. The number of pages used by secondary indices can be checked via the **sfsadmin query index** command. If the number of pages is significant enough to justify adding them to the space available for actual data, the key(s) can be deleted and then recreated onto another volume. Use the **sfsadmin delete index** and **sfsadmin add index** commands to accomplish this. When the index is deleted, the space allocated by the index will be freed and made available to other SFS data and indices allocated to that volume.

HPSS must be down during the process of deleting and rebuilding the indices. When the index is recreated, it must be created exactly as the original was specified. The **managesfs** *utility should be reviewed to determine the exact naming and syntax. Depending on the number of records in the SFS file, adding/recreating the indices may take a significant amount of time.*

8.7.3 Name Server Space Shortage

When the Name Server space usage has exceeded either the warning or the critical threshold, the administrator should analyze the situation based on the anticipated usage and the actual usage of the HPSS name space. The administrator needs to determine whether the current growth is legitimate or due to user or application errors where files and directories are created unintentionally. If more space is actually needed, additional name space can be created after careful considerations. Refer to Section 2.9.2.2 for more information on determining the required name space.

Additional name space can be allocated by increasing the Name Server's **Maximum Records** field in the Name Server Configuration window. Perform the following steps:

1. **Determine the additional SFS records required.** Determine how many additional SFS records are required to satisfy the current name space usage requirements and any anticipated future requirements.

2. **Verify the SFS space availability.** Verify that the current allocation of the SFS space will be able to absorb the name space increase and still be able to satisfy other SFS space requirements. The Metadata Monitor Information window can be used to determine the SFS space availability.

3. **Increase the name space.** Change the **Maximum Records** field using the Name Server Information window and/or the Name Server Configuration window depending on whether the Name Server is executing when the update needs to be made.

- If the Name Server is not running, bring up the Name Server Configuration window. Update the **Maximum Records** field with the desired value. The Name Server will use the modified value when it is restarted.
- If the Name Server is running, the **Maximum Records** field in the Name Server Information window can be changed. However, to avoid an inconsistent Name Server database, the **Maximum Records** field in the Name Server Configuration window must be updated first. Once this change has been made, bring up the Name Server Information window. Update the **Maximum Records** field with the desired value. The executing Name Server will use the new value immediately and will clear any threshold-exceeded conditions.

8.8 Managing Storage Server Volumes

The paragraphs that follow discuss some special intervention procedures that may be followed to manage storage server tape and disk volumes when tape volumes are marked full prematurely or when volumes need to be made unavailable.

8.8.1 Tape Volumes Marked Full Too Soon

Some unusual error conditions can cause the Tape Storage Server to change a Virtual Volume's state to **Allocated and Full** before the Virtual Volume (VV) is filled, leaving some portion of the VV unused. These unusual errors include attempts to write at an address other than the end of the tape and certain I/O errors from a Mover. When one of these errors occurs, the Storage Server generates an **HPSS_EOM** error. Depending on the exact situation at hand, the server may then change the PV and/or VV state to **Allocated and Full** and the VV's storage map state to **EOM** before the VV is actually full.

While some of the errors that lead to this condition generate alarms, others do not. Since these errors are normally quite rare, an occasional VV left in this condition should not pose a problem in most systems. However, if for some reason one of these unusual errors becomes frequent, the administrator will probably notice VVs being consumed at a high rate, or VVs being selected for repacking that were just written. In this event, intervention may become necessary.

Study the Storage Server and Mover log messages to determine the underlying problem that is causing the Storage Server to terminate the VV prematurely. Debug level logging will probably be necessary to obtain enough detailed information to identify and correct the problem.

HPSS will “recycle” these virtual volumes under normal operation (either due to storage segments on those volumes being deleted or due to those volumes being chosen to be repacked as part of operational tape management - and they are likely to be selected since they have a relatively small amount of used space compared to tape that have truly been written to the end of media). To speed up the process, especially if a number of VVs are affected, the **repack** utility can be issued from the command line to repack specific volumes (the SSM window does not support specifying specific volumes). Refer to Section I.49 in Appendix I for more information on the **repack** utility. Once these volume have been repacked, they can be reclaimed for subsequent use by HPSS or removed from the system.

Because the relationships between the PV, VV, Storage Map, and Storage Segment metadata are complex, the administrator should not attempt to manually change the VV metadata state to return a VV to service.

8.8.2 Making Volumes Unavailable

To take a VV completely off-line so that it will not be read, written or mounted, the preferred method is to set the **Administrative State** field in the VV metadata record to **Locked**. The procedure for displaying the VV metadata and changing the **Administrative State** is detailed in section 6.6.5.2, *Monitoring Storage Server Virtual Volumes*.

When the **Administrative State** of a VV is **Locked,** the Storage Servers (both tape and disk) will return an error for any request to perform a read or write on the VV. Attributes of the VV can be changed, and the VV can be deleted (if empty), but all other requests will be returned with the **HPSS_EPERM** error.

8.8.3 Making Disk Volumes Read-Only

To prevent the creation of new storage segments on disk VVs, set the **Administrative State** field in the associated disk storage map to **Shut Down**. The server will immediately set the **Administrative State** of the disk storage map to **Locked**, preventing new storage segments from being allocated on the volume. Existing segments can be read, written, and deleted. To empty disks while in use, set the **Administrative State** field to **Shut Down** and empty the disk by purging it or moving all of its storage segments to other volumes (Refer to Section I.49 in Appendix I for more information on how to use the **repack** utility to repack disk volumes). The procedure for displaying the disk storage map and changing the **Administrative State** is detailed in section 6.6.5.3, *Monitoring Storage Server Storage Maps.*

8.8.4 Making Tape Volumes Read-Only

To prevent the creation of new storage segments on tape VVs, set the **Administrative State** field in the associated storage map to **Shut Down**. If the storage segment located at the logical end of the volume is not writable, the **Administrative State** will change immediately to **Locked** and no further storage segments will be created on the VV. Since only the segment at the logical end of the volume can be written, no further writes will take place. If the storage segment at the end of the volume is writable, the **Administrative State** will either be set to **Locked** or remain set to **Shut Down,** depending on the exact circumstance at hand. The server will change the **Administrative State** from **Shut Down** to **Locked** when a process writing the volume finishes with it.

The procedure for displaying the tape storage map and changing the Administrative State is detailed in section 6.6.5.3, *Monitoring Storage Server Storage Maps.*

Chapter 9

HPSS Problem Diagnosis and Resolution

9.1 Overview

This chapter provides advice for solving selected problems with HPSS infrastructure components, servers, and user interfaces. Information presented in this chapter can be used as a supplement to the *HPSS Error Messages Reference Manual*, HPSS Release 4.1.1, which contains descriptions of specific logged error messages and suggestions for administrative action. Note that a problem may have more than one diagnosis and resolution.

9.2 HPSS Infrastructure Problems

The paragraphs below describe infrastructure problems with DCE, Encina, and Security.

9.2.1 DCE Problems

9.2.1.1 One HPSS server cannot communicate with another

Diagnosis **1**: The target server may not have registered properly with the Cell Directory Service (CDS), or with the endpoint mapper.

Resolution: Verify proper registration and existence of a running CDS and/or endpoint mapper. If shutting down the target server and restarting it does not fix the problem, you may have to manually delete the server's entry in the CDS and/or endpoint mapper.

Diagnosis **2**: The CDS entry for the target server may be corrupted.

Resolution: Stop the target server and then use the command **cdscp list object <ServerName>/*,** where **<ServerName>** is the CDS name of your server. This will give a list of several CDS objects. For each one, use the command **cdscp delete object <ServerName>/<objectName>**, where

<object-Name> is the appropriate object name. This will delete the objects. See Section 9.2.1.3 for additional information that may help with this problem.

Under no circumstances should the Server Security Object be deleted.

Diagnosis **3**: The endpoint mapper may have stale entries.

Resolution: To find out if the entries are stale, enter the command **rpccp show mapping > myFile** on the host where the server is running and inspect the resulting file. If there are more than a few entries for the server, it may be best to delete them all and start fresh. To do that, use the command **rpccp remove mapping -i <interface> -b <binding>,** where **<interface>** and **<binding>** can be obtained by cut and paste from the output of the **show mapping** command.

Diagnosis **4**: A communications failure may exist or security may be disallowing communication.

Resolution: First verify that the network is up and the server is running. A less obvious cause for the problem may be that the server is not accepting calls from the client because of security reasons. DCE converts some security violations into communication failures, thus obscuring the real problem. To fix this problem, make sure that the client and server are using consistent security policies, and that they have both been properly authenticated.

Diagnosis **5**: The DCE servers may not be working.

Resolution: If **/var** is full, try running **rmxcred** and clean out **/var**, or increase its size. Contact your DCE administrator if necessary. Check to see if a **dce_login** is possible. If not, DCE has problems.

For AIX 4.2, check to see if the following DCE processes are running: **cdsclerk** (depends on configuration), **cdsd** (depends on configuration), **cdsadv** (depends on configuration), **secd** (only on DCE server machine) and **dced**. Contact your DCE administrator if necessary.

Diagnosis **6**: The server may be logging too many messages.

Resolution: If a server (most notably the SSM) is forced to process too many log messages, it can become too busy to communicate with other servers. To fix the problem, turn off unneeded messages (particularly, trace and debug messages).

Diagnosis **7**: A server may be too busy to respond.

Resolution: If a server is very busy, other servers will not be able to communicate with it. To solve the problem, do what is necessary to decrease the load on the server. For example, you might try increasing the server's thread pool size and/or maximum connection count, moving the server to a different machine, or adjusting one of the server-specific configuration parameters.

Diagnosis **8**: A server may be trying to use TCP protocol.

Resolution: HPSS servers are expected to use User Datagram Protocol (UDP). Look at the server's endpoints in the endpoint mapper, using the following commands:

```
rpccp show mapping > mapping.out
vi mapping.out
```

You will get a listing that includes lines that look like this:

```
<object>          fab1d082-bb32-11cf-ae96-08005a4726ef
<interface id>    0020699e-cd72-1d35-a847-000000337238,2.0
<string binding>  ncadg_ip_udp:146.246.243.17[1553]
<annotation>      /.:/hpss/hpssd_tardis

<object>          fab1d082-bb32-11cf-ae96-08005a4726ef
<interface id>    0020699e-cd72-1d35-a847-000000337238,2.0
<string binding>  ncacn_ip_tcp:146.246.243.17[135]
<annotation>      /.:/hpss/hpssd_tardis
```

The first set of lines shows a binding that uses UDP and should work. The second set of lines shows a binding that uses TCP. It will work properly with many DCE servers, but can cause problems with HPSS servers.

If a TCP binding does appear, make sure that the **RPC_SUPPORTED_PROTSEQS** environment variable is set to **ncadg_ip_udp** in the **/usr/lpp/hpss/config/hpss_env** file. You will need to restart the Startup Daemon and HPSS servers to verify that the change has taken place.

Note that it might seem like the problem can be fixed by adding the following line to the AIX-specific file **/etc/environment**:

```
RPC_SUPPORTED_PROTSEQS=ncadg_ip_udp
```

However, this is strongly discouraged because it will affect all servers, including the DCE and Encina servers, with unpredictable consequences. In particular, as of this writing, the DCE endpoint mapper (**rpcd**) will not work with this value.

Diagnosis **9**: A multi-homed host may be using the wrong interface.

Resolution: Machines that have several IP addresses may have trouble communicating over all the network connections. To fix the problem, edit **/etc/environment** to include a line such as this: **RPC_UNSUPPORTED_NETIFS=fi0,** where **fi0** is replaced by the name of the hardware interface that needs to be disabled.

Diagnosis **10**: A node does not have a network route to an interface being used by a server on a different host.

Resolution: Verify that all nodes that will be clients of a server running on another node (including servers that are clients of other servers) have network routes to all of the network interfaces that may be used by the server nodes. For example, if a server running on a node registers its service interface on both an ethernet and Fiber Distributed Data Interface (FDDI), verify that all nodes that contain clients of that server have connectivity to both those interface addresses. Refer to Diagnosis 9 above for a related problem and / or alternate resolution to this problem.

9.2.1.2 A server cannot obtain its credentials

Diagnosis: There may be a problem with the keytab table.

Resolution: Make sure the keytable (typically **/krb5/hpss.keytabs**) is readable by the UNIX username under which the server is running. Make sure that the key contained in the keytab table is the correct one. Look for extra versions of the servers key; they can interfere with the

authentication process. If necessary, rebuild the keytab table using the **rgy_edit ktadd** command, and restart HPSS.

9.2.1.3 A server cannot register itself with the CDS

Diagnosis: The server's CDS entry may not exist or its Access Control List (ACL) may not have been set up to allow the creation of new objects in that directory.

Resolution: The DCE administrator should use **dcecp** and possibly **acl_edit** to properly configure the CDS for the server.

The SSM System Manager sets ACLs when it creates servers initially via the Add Basic Server Configuration in the following manner:

CDS directories, all server types:

acl_edit DIRNAME -m group:hpss_server:rwdtcia
acl_edit -e DIRNAME -m group:hpss_server:rwdtc
acl_edit DIRNAME -io -m group:hpss_server:rwdtc
acl_edit DIRNAME -ic -m group:hpss_server:rwdtcia

Security objects, all server types:

acl_edit DIRNAME/Security -m user:hpss_ssm:rwdtc
acl_edit DIRNAME/Security -m group:hpss_server:rwdt (except for NS and BFS)

Security objects, additional ACL for Bitfile Servers:

acl_edit DIRNAME/Security -m user:hpss_mps:rwdtc
acl_edit DIRNAME/Security -m user:hpss_nfs2:rwdtc
acl_edit DIRNAME/Security -m user:hpss_dmg:rwdtc
acl_edit DIRNAME/Security -m user:hpss_cns:rwdtc

Security objects, additional ACL for Name Servers:

acl_edit DIRNAME/Security -m user:hpss_dmg:rwdtc
acl_edit DIRNAME/Security -m user:hpss_ftp:rwdtc
acl_edit DIRNAME/Security -m user:hpss_bfs:rwdtc
acl_edit DIRNAME/Security -m user:hpss_nfs2:rwdtc

9.2.1.4 One of the servers cannot find another in the CDS

Diagnosis **1**: A CDS directory for the missing server may not exist or may have an ACL that does not permit the server to read it.

Resolution: Check the contents of the directory **cdscp list object /.:/<root>/<server>/*** where **<root>** is the path name for the root directory for the HPSS CDS names and **<server>** is the name of the server being sought. There should be several entries. The following entry is a constant value and is always present: **0068e69c-b7d8-1c39-a18a-000000337238,1.0.** This is where the server registers its connection manager endpoint. In addition, there will be other objects, each named by an interface ID that the server is using. If an object is missing, it means that the server is not running or is unable to register its services. If any of the objects contain bad information, the entry should be deleted. In

particular, watch for entries that do not contain any endpoints, entries that contain too many endpoints, entries that use unsupported protocols, entries that use unsupported hardware interfaces, and entries that either do not specify an object Universal Unique Identifier (UUID) or specify more than one. Also watch for entries that do not correspond to the contents of the endpoint mapper. For additional information that may help with this problem, see Section 9.2.1.3.

Diagnosis **2**: Servers are observing time skew problems (invalid credentials).

Resolution: Ensure that the date and time are correctly set for all hosts executing HPSS servers. Server credentials (and user credentials) have a 5-minute window between the DCE Registry and / or each other. If possible, implement the DCE **dts** component or some other mechanism to maintain time synchronization.

Diagnosis **3**: The server may be too busy to respond.

Resolution: If a server is very busy, other servers will not be able to communicate with it or reconnect to it. To solve the problem, do what is necessary to reduce the load on the busy server.

Diagnosis **4**: Two servers may be using the same CDS entry.

Resolution: This problem will cause confusion because other servers will not be able to tell the two servers apart. For example, even if there are several kinds of Storage Servers running at the same time, they must still use different CDS entries. Likewise, if testing is being done alongside production, the test servers must use different CDS entries from the production servers. To fix the problem, use SSM to reconfigure any servers that are using the CDS entries already in use, and stop and restart the modified servers. It may also be necessary to restart the server that was legitimately using the CDS entry and any servers with whom the affected servers were trying to communicate.

9.2.1.5 The connection table may have overflowed

Diagnosis: The server may be so heavily loaded that it is unable to free up connections easily.

Resolution: Do what is necessary to reduce the load on the server. The problem may also indicate that a server is configured incorrectly, or that there is a software problem in handling connections properly. To solve the problem, increase the **maximum connections** parameter in the basic server configuration file.

9.2.2 Encina Problems

9.2.2.1 HPSS servers cannot communicate with SFS

Diagnosis: DCE and / or Encina SFS is not running.

Resolution: Verify whether DCE is running and restart as appropriate. Determine whether SFS is running using **ps -e | grep sfs**. Execute **rc.encina** to start SFS if necessary.

9.2.2.2 One or more HPSS servers are receiving metadata or SFS errors

Diagnosis **1**: Permissions on the SFS file may not be set correctly.

Resolution: Change the file ACLs, using **acl_edit**, to include the permission **group:hpss_server:ADEIQRU**.

Diagnosis **2**: Some other Encina error condition exists.

Resolution: Examine the SFS or metadata return code(s) reported by the HPSS server. Using the **translateError** utility, determine what specific Encina error condition is occurring. Consult the Encina documentation or the Encina / HPSS technical support personnel to determine the appropriate resolution to the problem.

9.2.2.3 Cannot start SFS server

Diagnosis: The last security key table (keytab) update was not successful (error DCE-SEC-0253 "Invalid Password").

Resolution: Delete the last entry (the entry with the highest version number) for the SFS server principal in the Encina keytab file. The default pathname of the keytab file used is either **/opt/encinalocal/encina/sfs/<server name>/keyfile** or **/opt/encinalocal/encina/server/<server name>/keyfile**. The **rgy_edit** utility can be used to delete the desired entry as follows;

$rgy_edit

rgy_edit => ktl -f <keytab filename>

Example list of principals / versions

/.../hpss.clearlake.ibm.com/encina/sfs/hpss32_41 170

/.../hpss.clearlake.ibm.com/encina/sfs/hpss32_41 173

/.../hpss.clearlake.ibm.com/encina/sfs/hpss32_41 174

/.../hpss.clearlake.ibm.com/encina/sfs/hpss32_41 171

/.../hpss.clearlake.ibm.com/encina/sfs/hpss32_41 172

rgy_edit => ktd -p <principal> -v <version> -f <keytab filename>

rgy_edit => exit

9.2.3 Security Problems

9.2.3.1 HPSS servers are unable to connect to other HPSS servers

Diagnosis: This problem can be caused by several configuration or installation errors.

Resolution: Check to see that the CDS directories of the affected servers (both client and server) are correctly defined with appropriate permissions granted on the Security object in each directory. If the CDS directory entries appear valid but servers still cannot connect, turn on **TRACE** and **DEBUG** logging and attempt to restart both servers. Check the HPSS for DCE-related log messages

and take action accordingly. For additional information that may help with this problem, see Section 9.2.1.3.

9.2.3.2 SSM is unable to get HPSS server attributes.

Diagnosis: This problem is similar to the problem stated in Section 9.2.3.1 above.

Resolution: Check to see that SSM and the target server CDS directories are defined correctly and that the SSM principal has control permission on the Security object in the target server CDS directory.

9.2.3.3 Client API users get "credentials expired" errors

Diagnosis: This problem occurs because the DCE security service puts a time limit on the credentials cache obtained through **dce_login**.

Resolution: Review the DCE documentation on the **dce_login** and **kinit** utilities for your cell to identify options for increasing the credential's lifetime and renewability.

9.2.3.4 dce_login *fails*

Diagnosis **1**: This problem can occur for several reasons: (1) the user does not have a DCE account or the account has expired; (2) the user entered an invalid password; (3) time skew between the user's system and the DCE security server system is greater than 5 minutes; and (4) the DCE Security Server or Client is not running.

Resolution: Contact the DCE cell administrator to correct the problem.

Diagnosis **2**: A user's password is changed on one host but the user is also running on a second host.

Resolution: DCE keeps two versions of the user's password in the registry until the second process completes or logs out. After completion or log out, the problem is resolved.

9.2.3.5 dce_login *</…/other_cell/user> fails*

Diagnosis: This problem can occur for several reasons: (1) The GDA Daemon for the cell may not be running; (2) The Network between the two cells is inoperative; (3) The Domain Name Servers are not correctly setup; and (4) The remote cell has disabled your account or has not correctly setup your account.

Resolution: Contact the local DCE Cell Administrator to determine if this is the problem and subsequently start the GDAD on the appropriate host.

9.3 HPSS Server Problems

The paragraphs below discuss server problems common to all servers; problems with the Name Server, Bitfile Server, Storage Server, Migration / Purge Server, PVL, PVR, and Mover; and problems with Logging Services, NFS, Startup Daemon, and SSM.

HPSS Servers are started in separate directories to prevent the overwriting of core files in the event of a server terminating abnormally. The parent of these directories is controlled by the **HPSS_PATH_CORE** environment variable, with the default being **/var/hpss/adm/core**. Within that directory, subdirectories are created based on the Server descriptive name (with spaces replaced by underscores and parentheses dropped - e.g., for the Mover with descriptive name “Mover (hpss)”, the directory created will be “Mover_hpss”). Core files detected during server startup will be renamed based on the date and time that the core file was written (of the form core.YYYY_MMDD_hhmmss, where YYYY is the year, MMDD the month and day, and hhmmss is the hour, minute and second).

In addition, a log is maintained of servers that terminate with an abnormal termination code. The parent directory of the log file is controlled by the **HPSS_PATH_ADM** environment variable, with the default being **/var/hpss/adm**. The file name is **hpssd.failed_server**.

*If an HPSS server terminated abnormally, the appropriate core file should be saved. In addition, an HPSS delog should be performed to capture the HPSS messages logged by the server and other servers interfacing with the server around the abnormal ending period.*Problems Common to All Servers

9.3.0.1 Servers cannot be started

Diagnosis **1**: The **Executable** flag for a particular server is not set in the server’s configuration file.

Resolution: If the **Executable** flag for a server is not set, SSM will refuse to start that server. To fix the problem, set the flag.

Diagnosis **2**: The Startup Daemon on the server's host is not running. SSM cannot start the Startup Daemon.

Resolution: The Startup Daemon is normally started automatically at system boot time. If it is not running on the affected host, start it manually by executing the **rc.hpss** script there. Once the Startup Daemon is up and SSM can connect to it, try starting the target server again.

Diagnosis **3**: The Startup Daemon on the server's host is running, but SSM cannot connect to it.

Resolution: The only way SSM can start servers is by asking the Startup Daemon to do it, so it is essential that SSM is able to connect to the Startup Daemon. See whether SSM is connected to any other HPSS servers on that host, or whether any non-HPSS program (such as **ping** or **telnet**) can communicate between the two hosts. If the network itself is all right, try the **Force Connect** button from the Server List window (Chapter 6, Figure 6-2) to get SSM to connect to the Startup Daemon. If this does not work, the SSM System Manager and / or the Startup Daemon may have to be restarted.

Diagnosis **4**: The Startup Daemon on the server’s host is running and reachable, but SSM refuses to try to connect to it because the Startup Daemon is marked non-executable.

Resolution: SSM spends a good deal of time pinging each server to make sure it is still connected to it (and trying to reconnect if it is not). In an attempt to avoid wasting resources on a server that is not going to be running anyway, SSM ignores servers whose **Executable** flag is not set. If the Startup Daemon itself does not have the **Executable** flag set, SSM will never try to connect to it. In that case, SSM will not be able to start any servers on that host because it was not connected to the Startup Daemon there. Therefore, even though SSM does not start the Startup Daemon itself, make sure the Startup Daemon's **Executable** flag is set.

9.3.0.2 Server performs sluggishly

Diagnosis **1**: The size of the thread pool or the maximum number of connections allowed for a server may be too large.

Resolution: Decreasing the number of concurrent requests that a server can process may improve performance. Consider reducing the size of the thread pool or the number of connections to investigate performance implications. To reduce the size of the thread pool or to decrease the number of connections, call up the Server Configuration window (Chapter 5, Figure 5-4) and select the server in question. Modify the **Thread Pool Size** or **Maximum Connections** configuration variables as necessary. The server must be reinitialized or else stopped and restarted to pick up the new configuration.

Diagnosis **2**: The Domain Name Service (DNS) is *not* reachable.

Resolution: Add *all* necessary entries to the **/etc/**host file. Terminate all HPSS servers, Encina SFS servers, and DCE. Restart the system without DNS support. Fix DNS.

Diagnosis **3**: Either one or more network components is unusable or the network interface is unavailable.

Resolution: Repair the network. Implement the **RPC_UNSUPPORTED_NETIFS** environment variable for the disrupted network interface and restart the system.

Diagnosis 4: The RPC port mapper has incorrect, invalid, or extra mappings.

Resolution: Use the **rpccp show mapping** command to observe RPC mappings. Use the **rpccp** command to remove invalid mappings. Contact your site DCE administrator or reference the *DCE V.1.3 for the AIX Administration Guide* for syntax of commands.

9.3.1 Name Server (NS) Problems

9.3.1.1 NS response to a list directory operation is slow

Diagnosis: The NS configuration parameter **MaxByteSizeOfBuffer** is too small.

Resolution: Increasing the maximum number of bytes that the NS returns from a list directory operation may improve the slow response time. Section 5.6.1 describes how the Maximum Buffer Size parameter can be changed.

9.3.1.2 NS is unable to create new directory entries

Diagnosis **1**: The NS may have exhausted its allocation of SFS records.

Resolution: If Diagnosis 1 is correct, the NS will have issued alarms indicating that it is low on space. If SFS has sufficient disk space available, increasing the number of SFS records allocated to the Name Server may solve the problem. Section 8.7.3 describes how the Maximum Records parameter can be increased. If SFS does not have sufficient disk space, it may be possible to increase the size of the SFS volume. Section 8.7.2 describes how the SFS volume size can be increased.

Diagnosis **2**: The CDS ACL for the NS may not be set correctly.

Resolution: In this scenario, bitfiles cannot be added to directories. The creation of new directories, hard links, or symbolic links will not be effected. The CDS ACL for the Name Server might not have "control" permission on for the BFS. Table 5-1 under *Advice* for **Server CDS Name** describes how to properly set the CDS ACL for the Name Server.

9.3.2 Bitfile Server (BFS) Problems

9.3.2.1 BFS cannot connect to Storage Servers

Diagnosis **1**: One or more of the Storage Servers is not up.

Resolution: Restart the Storage Server in question. The BFS will attempt to connect to this Storage Server the next time it is needed. This should happen fairly quickly because the background thread that monitors storage space usage statistics for Storage Servers will attempt to contact the Storage Servers within a few seconds.

Diagnosis **2**: Storage Server information for the BFS is not configured correctly.

Resolution: Change the BFS specific configuration information to accurately reflect the Storage Servers. Recycle the BFS. For additional information that may help with this problem see Sections 9.2.1.1 and 9.2.1.4.

9.3.2.2 BFS cannot connect to SSM

Diagnosis: The SSM System Manager is not running or not responding.

Resolution: If the SSM window is responding, check the status of SSM's connection to the BFS at the Server Status window and reconnect if necessary. For additional information that may help with this problem, see Sections 9.2.1.1 and 9.2.1.4.

9.3.2.3 Errors reading, writing, creating, or deleting entries in metadata (SFS) files

Diagnosis: Error codes indicate that SFS is not running or not responding.

Resolution: Check the status of SFS with Encina administration tools, and restart if necessary.

9.3.2.4 BFS takes a long time starting

Diagnosis **1**: The server is taking a long time to close outstanding SFS Open File Descriptors (OFDs).

Resolution: The BFS closes any OFDs that were left open from a prior execution of the BFS that crashed. This is to be expected if there was a large amount of activity when the BFS crashed and is a normal recovery operation.

Diagnosis **2**: When the BFS starts, it attempts to connect to all of the Storage Servers and SSM at initialization time, which can lead to a time delay if one or more of the servers it not operational.

Resolution: When the servers that were not up are started, BFS will automatically connect to them. To minimize the delays, start all of the servers that BFS connects to before starting BFS.

9.3.2.5 Service parameters have been changed and BFS does not recognize them

Diagnosis: The BFS has not been recycled since the changes were made. (Potentially includes Class of Service, Storage Class, Storage Hierarchy, and Migration Policy definitions.)

Resolution: The BFS does not pick up this information automatically. Recycle the BFS.

9.3.2.6 Unable to write new records to the SFS BFS storage segment unlink file

Diagnosis **1**: One or more of the Storage Servers is down.

Resolution: If one or more of the Storage Servers is down, the BFS will not be able to successfully delete storage segments targeted for that server, and the associated storage segment unlink records cannot be eliminated. Restart the needed Storage Server. If Storage Servers are to be down for substantial periods of time, it may be necessary to expand the storage segment unlink file so that it can hold more records.

Diagnosis 2: The storage segment unlink file is too small.

Resolution: A heavy load of delete activity on the system can cause unlink records to be created at a faster rate than the rate at which BFS is able to unlink storage segments. Increase the size of the storage segment unlink file.

9.3.2.7 Receiving messages from BFS indicating inconsistencies in account summary records

Diagnosis 1: Account summary record has been corrupted due to an incomplete SFS recovery.

Resolution: Specialized procedures are provided to deal with this problem. Contact IBM support.

Diagnosis 2: Software problem in HPSS has resulted in the inconsistency.

Resolution: Contact IBM support.

9.3.3 Storage Server (SS) Problems

9.3.3.1 SS cannot connect to PVL

Diagnosis: The PVL is not running, or is not responding.

Resolution: Restart the PVL. The Storage Server will try to connect to the PVL for up to 5 minutes before returning an error to a request to mount a disk or tape. Each additional mount request will try for 5 minutes before returning an error. For additional information that may help with this problem, see Sections 9.2.1.1 and 9.2.1.4.

9.3.3.2 SS cannot connect to SSM

Diagnosis: The SSM System Manager is not running or not responding.

Resolution: If the SSM window is responding, check the status of SSM's connection to the SS in the Status field on the Server List window (Chapter 6, Figure 6-2) and reconnect if necessary. The SS will attempt to connect to SSM indefinitely, but retains a limited number of messages to send to SSM in a queue. When the queue becomes full, additional messages are discarded until the connection is re-established. For additional information that may help with this problem, see Sections 9.2.1.1 and 9.2.1.4.

9.3.3.3 Errors reading, writing, creating, or deleting entries in metadata (SFS) files

Diagnosis **1**: Error codes indicate that SFS is not running or not responding.

Resolution: Check the status of SFS with Encina administration tools, and restart if necessary. The Storage Server will retry timed-out SFS I/O operations a fixed number of times before generating an error.

Diagnosis **2**: Error codes indicate that an invalid OFD was found.

Resolution: The Storage Servers will discard an invalid OFD and obtain a new one and retry the operation in most cases. In other cases, an error will be returned immediately to the client. If no error was returned to the client, no action need be taken (the server recovered the error). If an error was returned, log messages should be noted, but little can be done unless the problem becomes persistent.

9.3.3.4 Server takes a long time to come up

Diagnosis **1**: The server is taking a long time to close outstanding SFS OFDs.

Resolution: If the problem does not repeat, a large number of OFDs were successfully closed. This is the intended operation of the system and no action need be taken if the reason for the large number of outstanding OFDs is known. If the problem repeats each time the server is started, investigate the outstanding OFDs (using **sfsadmin,** for instance) to find out why the OFDs are not being closed when the server starts. If the OFDs were created by another server, SS restarts can be affected

because all of the OFDs at the SFS server must be examined by the SS to find OFDs to close. Restarting the server that created the OFDs may correct the problem.

Diagnosis **2**: (for Disk SS only): The Disk SS is taking a long time to mount each disk physical volume via the PVL.

Resolution: Check the status of the PVL and the status of the connection to the PVL from the Disk SS. Each disk physical volume must be mounted in the PVL before the Disk SS can complete its initialization. The server will "hang" until these steps are complete, or will produce a fatal error message and halt.

9.3.3.5 Disk storage server reports "no space"

Diagnosis **1**: The disk VVs are fragmented.

Resolution: The Bitfile Server makes requests to the Disk Storage Server to create one or more disk storage segments in which a file will be recorded. The Bitfile Server determines the size of these storage segments according to the size of the file to be recorded. The Disk Storage Server attempts to find free space on the disk virtual volumes (VVs) it manages in which to create the disk storage segments. The free space for each segment must be made of contiguous disk VV blocks.

If all of the disk VVs are fragmented to a point where a storage segment cannot be created at the requested length, the Disk Storage Server will report that it is out of disk space and return an error to the Bitfile Server. An alarm is sent to the "Alarms and Events" display.

The total free disk space in the system may exceed the requested storage segment size, but if the disks are sufficiently fragmented, it may not be possible to create one or more of the fragments.

Two solutions are available for this problem. Disk VVs could be repacked, which will create large blocks of free space, and purge parameters can be changed to increase the amount of free space in the VVs, which may increase the sizes of the largest free blocks.

9.3.4 Migration/Purge Server (MPS) Problems

9.3.4.1 No storage class information reported on Storage Class List window (Chapter 6, Figure 6-5)

Diagnosis **1**: The startup of the MPS is not completed.

Resolution: Wait until the **OpState** of the MPS server is **ENABLED** before selecting the Storage Class List from the **monitor** pull-down menu.

Diagnosis **2**: There is an incorrect or missing specification of the MPS in the storage class configuration entry.

Resolution: Correct the storage class configuration entries. Shut down and restart the MPS.

Diagnosis **3**: The SS controlling the missing storage class has not been started.

Resolution: Start the SS.

9.3.4.2 A storage class does not show up in the Storage Class List window

Diagnosis **1**: The storage class has been added or updated after the MPS startup is completed.

Resolution: Shut down and restart the MPS.

Diagnosis **2**: No SS resources have been created in the storage class.

Resolution: Create the missing resource. From the HPSS Health and Status window (Chapter 6, Figure 6-1), select the **Operation** pull down menu and then select **Create SS Resources**.

Diagnosis **3**: The SS controlling the missing storage class has not been started.

Resolution: Start the SS.

Diagnosis **3**: The storage class configuration does not have an MPS associated with it.

Resolution: Modify the storage class configuration then recycle the MPS.

9.3.4.3 MPS cannot retain the connection with the SS

Diagnosis **1**: The MPS and SS were started at about the same time.

Resolution: Shut down and restart the MPS.

9.3.4.4 MPS is not migrating or purging data

Diagnosis: The run type is set to **Simulation**.

Resolution: The purpose of the **Simulation** run type is to produce reports only. It does not do any actual data migration or purging. If you want the migration / purge to take place, change the run type to **Regular** from the **Control** pull-down menu on the Storage Class Information window, (Chapter 6, Figure 6-11).

9.3.4.5 The purging runs more frequently or less frequently than desired

Diagnosis: The **Start Used Percent** and / or the **Target Free Percent** parameter(s) are set incorrectly in the purge policy.

Resolution: Correct the **Start Used Percent** and / or **Target Free Percent** parameter(s) in the purge policy. Shut down and restart the MPS.

9.3.4.6 The migration runs more frequently or less frequently than desired

Diagnosis **1**: The parameters are set incorrectly in the migration policy.

Resolution: Correct the **Runtime Interval, Last Read Interval, Last Update Interval** and / or **Free Space Target** parameter(s) in the migration policy. Shut down and restart the MPS.

Diagnosis **2**: The **Storage Class Update Interval** parameter is set incorrectly in the Migration / Purge Specific Configuration.

Resolution: If the **Storage Class Update Interval** parameter is set too large, the MPS will sample SS statistics too infrequently and act on the information too late. Set this parameter to a smaller value. Shut down and restart the MPS.

9.3.4.7 A purge is tried for a tape storage class

Diagnosis: (SSM issues an invalid argument message following a force purge command.) Force Purge is not supported for tape storage classes.

Resolution: Delete the purge policy from the tape storage class. The MPS purges all the storage segments from a virtual volume as part of the tape migration. There is no specific purge for tapes.

9.3.5 Physical Volume Library (PVL) Problems

9.3.5.1 Tape mount requests are not being satisfied

Diagnosis **1**: A connectivity failure exists between the PVL and either the MVR, PVR, or SS.

Resolution: Loss of inter-server connectivity should be evident via the HPSS Alarms and Events window (Chapter 6, Figure 6-47). Determine where connectivity problems exist and proceed to the appropriate problem diagnosis below.

Diagnosis **2**: Mount requests are queued in the PVL waiting for resources.

Resolution: Check PVL job queues and devices for resource shortages such as all devices being in use, multiple requests for the same cartridge, or drives being disabled. This examination should reveal the resource shortage as being drive or cartridge related. If the shortage is caused because a drive is in a disabled state, enable the drive (if appropriate) using the Drive Information window. If a true resource shortage exists, wait for resources to become available or cancel appropriate PVL jobs to free the required resource. If no resource shortage exists, then proceed to Diagnosis 3.

Diagnosis **3**: An internal PVL job queue error has occurred.

Resolution: Use the PVL Job Queue window (Chapter 8, Figure 8-2) to select the job in question. Cancel the job and retry it. If problems exist for all PVL mounts, restart the PVL.

9.3.5.2 A PVL job cannot be canceled

Diagnosis **1**: A connectivity failure exists between the PVL and the MVR or PVR.

Resolution: Loss of inter-server connectivity should be evident via the HPSS Alarms and Events window (Chapter 6, Figure 6-47). Determine where connectivity problems exist and proceed to the problem diagnosis below.

Diagnosis **2**: An internal PVL queue inconsistency exists.

Resolution: Restart the PVL. If the problem persists, check to see if the job involves a tape mount by clicking the **Job Info** button on the PVL Job Queue window (Chapter 8, Figure 8-2). If the job is a tape mount, determine the PVR involved by using the Device / Drive List window (Chapter 8, Figure 8-1) to select the **Drive ID** and using the **Info**... button to activate the Drive Information window. Shut down and restart the PVR in question.

9.3.5.3 SS mount requests are not appearing in PVL job queues

Diagnosis **1**: A connectivity failure exists between the PVL and the SS.

Resolution: Loss of PVL-to-SS connectivity should be evident via the HPSS Alarms and Events window (Chapter 6, Figure 6-47). Proceed to the SS connectivity failure problem given below in Section 9.3.5.7.

Diagnosis **2**: PVL and SS queues are not synchronized.

Resolution: After ensuring that the requests in question do not exist, restart the PVL. If this fails to correct the problem, restart the appropriate tape SS.

9.3.5.4 A tape cartridge is physically mounted in a drive but is not recognized by the system as being mounted

Diagnosis **1**: A connectivity failure exists between the PVL and an MVR.

Resolution: Loss of PVL-to-MVR connectivity should be evident via the HPSS Alarms and Events window. Proceed to the MVR connectivity failure problem description given below in Section 9.3.5.8.

Diagnosis **2**: Drive polling is not enabled for operator PVR.

Resolution: If the drive in question is in an operator PVR (is mounted by hand), polling may not have been enabled for the drive in question. Enable polling for the appropriate drive using the PVL Drive Information window.

9.3.5.5 A drive has been added to the PVL but is not being used by the system

Diagnosis **1**: The drive in question has not been enabled.

Resolution: Use the Drive Information window (Chapter 6, Figure 6-25) to enable the drive in question for reading and / or writing.

Diagnosis **2**: The PVL and MVR were not restarted after reconfiguration.

Resolution: Restart both the appropriate MVR and PVL for the modified configuration to take effect.

9.3.5.6 Imports of cartridges fail due to improper labeling

Diagnosis: The **Import Type** field was used incorrectly.

Resolution: The value of the **Import Type** field can be either **Default** or **Scratch**. For disk, always use the **Scratch** import type.

- Specifying **Scratch** will cause a label to be written on to the media no matter what is currently on it, potentially causing any data on the media to be lost. Because of the potential danger of importing media as **Scratch**, a dialog box will appear to confirm the choice.

- Specifying **Default** will cause an action depending on how the media is labeled (i.e., tape or disk). For tape media, the action taken is based on the current volume label type:

  HPSS—Media imported. The volume label type for this HPSS is: media has an ANSI label; i.e., it starts with an 80-byte block starting with the characters **VOL1**. The owner field of the ANSI label is set to **HPSS**.

  Foreign—Media imported. The volume label type for Foreign is: media has an ANSI label, but the owner field is not **HPSS**.

  Non-ANSI—Import fails. The volume label type for Non-ANSI is: media starts with an 80-byte block that does not start with the characters **VOL1**.

  No label, but data found—Import fails.

  No label or data—Cartridge is labeled and imported.

For disk media, the current volume label is read and if the volume identifier matches the identifier specified in the import request, the label is rewritten. This is done in case the volume is being re-imported with either a different block size or number of blocks, because these values are placed in the disk volume label. The MVR can then verify that the label matches the device configuration metadata. If the current volume identifier does not match the identifier specified in the import request, the import will fail.

9.3.5.7 PVL cannot connect to the SS

Diagnosis: The SS is not running or is not responding.

Resolution: Restart the SS. For additional information that may help with this problem, see Sections 9.2.1.1 and 9.2.1.4.

9.3.5.8 PVL cannot connect to an MVR

Diagnosis: The involved MVR is not running or is not responding.

Resolution: Restart the MVR in question. For additional information that may help with this problem, see Sections 9.2.1.1 and 9.2.1.4.

9.3.5.9 PVL cannot connect to the PVR

Diagnosis: The PVR is not running, or is not responding.

Resolution: Restart the PVR. For additional information that may help with this problem, see Sections 9.2.1.1 and 9.2.1.4.

9.3.5.10 PVL cannot connect to SSM

Diagnosis: The SSM System Manager is not running or is not responding.

Resolution: If the SSM window is responding, check the status of SSM's connection to the PVL in the Status field on the Server List window (Chapter 6, Figure 6-2) and reconnect if necessary. The PVL will attempt to connect to SSM indefinitely. For additional information that may help with this problem, see Sections 9.2.1.1 and 9.2.1.4.

9.3.5.11 Errors occur while reading, writing, creating, or deleting entries in metadata (SFS) files

Diagnosis: Error codes indicate that SFS is not running or is not responding.

Resolution: Check the status of SFS with Encina administration tools, and restart if necessary. The PVL will retry timed-out SFS I/O operations a fixed number of times before generating an error.

9.3.6 Physical Volume Repository (PVR) Problems

9.3.6.1 PVR is unable to communicate with a robot

Diagnosis: These errors are usually caused by configuration problems outside the control of HPSS.

Resolution: Verify that a non-HPSS process is able to talk to the robot. It is best to use the robot's own control software. For example, for a 3494 robot, try to mount and dismount a tape using the **mtlib** command on the workstation that is running the PVR. For the STK robot, try to mount and dismount a tape from the Automated Cartridge System Library Software (ACSLS) console. Additionally for STK, make sure the Storage Server Interface (SSI) process is running on the same workstation as the PVR. The SSI process must have been started before the PVR. For an ADIC AML robot, try to mount and dismount a tape using **dasadmin** commands. Note that the user must use the command "**mt -f /dev/rmtxx rewoffl**" to rewind and elevate the tape before issuing the **dasadmin** dismount command.

If the non-HPSS processes are able to mount and dismount tapes, check the PVR configuration. For 3494/3495 PVRs, verify that the Command and Async devices (generally **/dev/lmcp0**) are valid and available. Verify that the **HPSS_3494_COMMAND_DEVICE** and **HPSS_3494_ASYNC_DEVICE** HPSS environment variables are not defined. For STK robots, check that the packet version used by the PVR is the same as the packet version used by the SSI and ACSLS. Note that the packet version number is usually one less than the ACSLS software version number. Be sure that this number agrees with the HPSS environment variable **ACSAPI_PACKET_VERSION**. For ADIC AML, check the configured Server Name and Client Name fields in the PVR Type Specific Configuration entry. Make sure that these are the same as in the OS/2 PC configuration file. Also, the user should

monitor the OS/2 log file for additional information. The error messages are described in detail in the *EMASS Storage Systems AMU Reference Guide*.

9.3.6.2 PVR operational state is Major

Diagnosis **1***:* A cartridge has failed to mount.

Resolution: If a cartridge is supposed to be mounted by a human operator and the mount has been outstanding for about 20 minutes, the Operational State will be set to **Major** to signify that the mount is taking too long. If a cartridge is supposed to be mounted by a robot, but the robot is unable to mount the cartridge, a message will be logged indicating the problem and the Operational State will be set to **Major**.

Correct the problem indicated in the log and then force the mount to retry by setting the PVR's Administrative State to **Repaired**. If the mount fails again, the Operational State will remain set to **Major**. All mounts that have failed will be retried when the PVR is repaired. They will also be retried every 5 minutes. If any mount fails, the Operational State will be set to **Major**.

It is possible for the Operational State to be set to **Major** even if there are no mounts currently pending. If a mount fails due to a transient condition, the Operational State will be set to **Major**. If the automatic retry successfully mounts the cartridge later, the Operational State will remain set to **Major**. This allows the operator to identify and correct the transient condition. Set the Administrative State to **Repaired** to clear the Operational State.

Diagnosis **2**: For IBM 349x robots, insufficient drives are available to honor a mount request.

Resolution: This problem may have occurred due to the injection of a cleaning cartridge into a drive. The PVL is responsible for maintaining the available drive count; however, the PVL has *no* ability to know when a cleaning cartridge is injected. The PVR is very persistent and the problem will correct itself usually within 5 minutes. It is necessary to notify the PVR of repair in order to reset the Operational State to **Normal**. The PVR will continue to recheck for an available drive at 5-minute intervals until the problem is resolved.

9.3.6.3 3494 PVR fails to shut down

Diagnosis: The 3494 PVR spawns a child process. If that process fails to shut down, the **lmcp** daemon has hung the process.

Resolution: To correct this problem in the short term, as root User ID **kill -9** the **lmcp** daemon. When the daemon dies, the PVR processes will halt. This is indicative of an out-of-date **lmcp** daemon. To permanently fix this problem, install the latest version of the **lmcp** daemon.

9.3.7 Mover (MVR) Problems

9.3.7.1 MVR performs poorly

Diagnosis **1***:* A problem exists with the MVR internal buffer size.

Resolution: If the MVR buffer size is too small, the MVR will perform numerous separate requests when a single request could be made to perform the same input or output operation. If the MVR buffer size is too large, the MVR could reserve too much system virtual memory, requiring frequent paging of MVR and other process memory (which will also decrease performance) Also, if the buffer size is too large, transfers may be completed without the MVR receiving any benefit from double buffering. (For example, if the MVR buffer size is 4 MB but a majority of client requests are 4 MB or less, the Mover will complete the transfer using one buffer, thus not allowing any client and device I/O time to be overlapped.)

Diagnosis **2***:* A disk device is configured to use the block special file.

Resolution: If the device is configured to use the block special file, the data will be buffered by the operating system, which could cause additional overhead during read (primarily) and write operations. Also, data that the MVR believes has been written to disk may in fact only be stored in system memory, waiting to be flushed to disk.

Diagnosis **3***:* A disk device is not configured to use multiple MVR tasks.

Resolution: If the device is not configured to use multiple MVR tasks, the MVR will single thread I/O requests for that device. Modify the device configuration to enable multiple MVR tasks.

9.3.7.2 MVR cannot be started

Diagnosis **1***:* The MVR could not bind to the TCP/IP port number specified in the MVR specific configuration file.

Resolution: Verify that the hostname specified in the MVR specific configuration file relates to a valid network interface for the machine on which the MVR is running, and that the port number specified is a valid port number and one that is not in use by another process (possibly another HPSS MVR that was previously started on the same machine).

Diagnosis **2***:* The MVR could not bind to a UNIX domain socket used for intra-MVR communication.

Resolution: The MVR uses a set of UNIX domain sockets that are placed in **/var/hpss/tmp** while the MVR is running. If an MVR was previously running under a different UNIX user ID and was not cleanly shut down, the sockets may be left in the file system, and the newly started MVR may not be able to remove them. If this is the case, a user with sufficient privilege must remove the socket files in **/var/hpss/tmp** before the MVR can be run by the second user. The socket file names all begin with the prefix **Mvr**.

Diagnosis 3: The MVR cannot start either the DCE request process or the TCP/IP request process.

Resolution: The MVR DCE program must be located in the same directory as the main MVR process (or else the environment for the Startup Daemon must contain the directory in its executable path). Since the DCE process reads the MVR configuration metadata, it must be started before any configured values are known. The MVR TCP/IP program pathname is contained in the MVR specific configuration file, and may be verified by examining that information.

Diagnosis 4: The Mover is configured for non-DCE mode and the non-DCE node *inetd* configuration is incorrect.

Resolution: This diagnosis is likely to be correct if the Parent Mover process (on the DCE/Encina node) generated an alarm message indicating that it cannot establish a connection to the remote (non-DCE/Encina) node. To correct the problem, verify the /etc/services and /etc/inetd.conf configuration are correct (see Section 5.8.6). Also verify (typically via *netstat*) that there is a listen waiting on the appropriate TCP port.

Diagnosis 5: The Mover is configured for non-DCE mode and the encryption key is out of sync between the two Mover nodes.

Resolution: In this case, the Mover should generate an alarm message indicating that there is an encryption key mismatch with the non-DCE/Encina node. To resolve the problem, verify that the encryption key file (referenced in the */etc/inetd.conf* file) on the non-DCE/Encina node contains the same value as is configured in the Mover's type specific configuration.

Diagnosis 6: The Mover is configured for non-DCE mode and the skew between the DCE/Encina node and the non-DCE/Encina node (the two nodes that this Mover is executing across) is greater than the maximum allowable difference (currently 5 minutes).

Resolution: In this case, the Mover should generate an alarm message indicating that the clock skew is too great between the DCE/Encina node and the non-DCE/Encina node. To resolve the problem, one or both of the nodes' clocks must be adjusted so that they are within the allowable difference.

9.3.7.3 MVR cannot write a label to a tape

Diagnosis **1**: The MVR does not have the required privilege to access the device.

Resolution: Verify that the tape device special file is defined such that the user under which the MVR is running is able to access the file for both reading and writing. If the device is a DD-2 tape device, the MVR must be run under the **root** user id to allow appropriate access to the device.

Diagnosis **2**: The tape device is not configured to support reading and writing variable size blocks.

Resolution: Verify that the tape device is defined such that it will support variable size blocks. In AIX, this usually involves defining the block size of the device to be zero (either via the **smit** configuration utility or the **chdev** command).

9.3.7.4 MVR cannot read the label from a previously labeled tape

Diagnosis **1**: The tape device is configured as being able to support using the **no delay** flag on open, but in fact the device driver does not support issuing tape operations if the device was opened using the **no delay** flag.

Resolution: Change the device configuration to turn off the **no delay** support flag.

Diagnosis **2**: The tape device is not configured to support variable block sizes (either because the device was reconfigured or the tape was read on a device other than the one that was used to write the label).

Resolution: See the Resolution for Diagnosis 2 in Section 9.3.7.3.

9.3.7.5 Tape positioning operations are performing poorly

Diagnosis **1**: The tape device is not configured to support absolute positioning (fast locate).

Resolution: Change the device configuration to turn on the fast locate support flag if the device and driver interface support **fast locate**.

Diagnosis **2**: The MVR was not built with the compilation flag to include code for the device specific device driver interface, which would allow absolute positioning (fast locate) to be used.

Resolution: Rebuild the MVR to include support for the specific device driver interface being used, and modify the device configuration to turn on the **fast locate** support flag (if necessary).

9.3.7.6 MVR cannot perform an IPI-3 data transfer

Diagnosis **1**: The MVR cannot open the IPI-3 slave device for performing the data transfer.

Resolution: Verify that the IPI-3 slave records in the IPI-3 configuration file **ipi3en.conf** are properly defined: (1) the slave name must match the one being requested by the MVR, (2) the slave device must be the correct slave device for the machine on which the MVR is running (and the MVR must have the necessary privilege to access the device), and (3) the slave identifier must be unique and non-negative.

Diagnosis **2**: The MVR cannot find the client's configuration information.

Resolution: Verify that the same client name appears in the client configuration file **ipi3data.conf** on the client machine and in the client records in the IPI-3 configuration file on the machine on which the MVR is running.

Diagnosis **3**: The client record in the IPI-3 configuration file does not contain the correct **i-field** for the client.

Resolution: Determine the correct **i-field** value (based on the HiPPI connection topology and whether logical or source routing is being used), and verify that the correct value is entered in the client record of the IPI-3 configuration file **ipi3en.conf** on the machine on which the MVR is running.

9.3.7.7 MVR cannot read and write data from a HiPPI attached disk array

Diagnosis **1**: The device is configured incorrectly in the IPI-3 configuration file.

Resolution: Verify that the **DEVICE** and **PARTITION** stanzas are defined correctly in the **ipi3en.conf** configuration file. See Section 5.8.5 for further details.

Diagnosis **2**: The IPI-3 configuration for the machine on which the Mover is running is incorrect.

Resolution: Verify the correctness of the client configuration file **ipi3data.conf** on the client machine and the client records in the IPI-3 configuration file on the machine on which the MVR runs. See Section 5.8.5 for further details.

9.3.7.8 Network transfers are performing poorly

Diagnosis **1**: Routing tables on the node on which the Mover is running is incorrect.

Resolution: Verify that the system routes defined are causing the Mover to use the expected network connectivity when communicating with a remote client.

Diagnosis **2**: The networking options defined in the HPSS network option file (**hpss_netopt.conf**) are not optimally set for the utilized networks.

Resolution: Verify the correctness of the network configuration file **hpss_netopt.conf** on the mover machine for the utilized networks. See Section 5.8.6 for further details.

9.3.8 Non-DCE Client Gateway/Non-DCE Client API problems

9.3.8.1 The Non-DCE Client Gateway will not start

Diagnosis: The NDCG could not bind to the TCP/IP port number specified in the NDCG specific configuration file.

Resolution: Verify that the port number specified is a valid port number and one that is not in use by another process (possibly another NDCG that was previously started on the same machine).

9.3.8.2 Non-DCE client applications cannot communicate with the NDCG

Diagnosis: The Non-DCE Client API is attempting to connect to the Non-DCE Client Gateway through the wrong TCP/IP port.

Resolution: Verify that the port number specified in the Non-DCE Client Gateway's specific configuration is the same as the port specified with the **HPSS_NDCG_SERVERS** and **HPSS_NDCG_TCP_PORT** environment variables in the environment in which the non-DCE client application is running.

9.3.8.3 Non-DCE client applications occasionally have failed API calls

Diagnosis: The Non-DCE Client Gateway has more concurrent requests than it is configured to handle from a single client.

Resolution: Increase the values of the Maximum Request Queue Size and/or Maximum Thread Pool size in the Non-DCE Client Gateway server-specific configuration.

Diagnosis: The Non-DCE Client Gateway has established its maximum number of client connections, and all subsequent connection attempts fail.

Resolution: Increase the value of the Maximum Processes field in the Non-DCE Client Gateway server-specific configuration. Alternatively, a better solution may be to configure an additional Non-DCE Client Gateway on another node in order to distribute the work load between multiple machines.

9.3.9 Logging Services Problems

9.3.9.1 Logging performance is sluggish

Diagnosis: A large number of messages are being generated.

Resolution: Change the Logging Policy to filter out unneeded messages (the recommended record types to filter out first are Trace, Request and Debug messages). To set or modify Logging Policy, call up the HPSS Configuration window (Chapter 5, 5.4.4) and select **Logging Policy.** Modify metadata and reinitialize logging service components as necessary.

9.3.9.2 Log Daemon (or Log Client) will not start/restart

Diagnosis **1**: A Log Daemon (or Log Client) may already be executing.

Resolution: If the process is already executing and must be recycled, terminate the existing process and attempt to restart.

Diagnosis **2**: The group permissions for the Log Client and / or Log Daemon may not have sufficient permissions to access the log files.

Resolution: The Logging processes should be started from the same group as the Logging processes that originally created the files.

Diagnosis **3**: The Log Daemon will not initialize if both the primary and secondary log files (**logfile01** / **logfile02**) are marked current, if neither is marked current, or if either file is marked invalid.

Resolution: This error is not likely to occur, but if either log file does become corrupted, **logfile01** and **logfile02** should be deleted. Restart the Log Daemon and the new files will be recreated.

9.3.10 Network File System (NFS) Problems

9.3.10.1 NFS Daemon will not start

This problem can occur for several reasons. Review the HPSS log for messages that explain the nature of the problem and then proceed as discussed below.

Diagnosis **1**: NFS daemon was unable to read the HPSS exports file.

Resolution: Check for file existence, permissions, and correctness.

Diagnosis **2**: The HPSS root **uid** does not have an account in the DCE security registry.

Resolution: Use **rgy_edit** to verify whether an account exists.

Diagnosis **3**: The NFS Daemon was unable to read the credentials map file.

Resolution: Check for file existence, permissions, and correctness.

Diagnosis **4**: The NFS Daemon was unable to read the data cache and checkpoint files.

Resolution: Check for file existence and permissions.

Diagnosis **5**: The NFS Daemon was unable to open its SFS configuration file.

Resolution: Check for file existence and permissions.

Diagnosis **6**: Security initialization failed.

Resolution: Review the HPSS log for security subsystem messages and take appropriate action.

Diagnosis **7**: Errors occurred while initializing DCE/Encina.

Resolution: Turn on **Trace** and **Debug** record types if they are not set and attempt to restart the NFS server. If the server is having trouble registering its service, make sure the NFS CDS directory exists and that it has a proper ACL.

Diagnosis **8**: The NFS Daemon will not remain up.

Resolution: Make sure the BFS and the NS are running.

Diagnosis **8**: The NFS Daemon died with "Biding Address In Use ..." error.

Resolution: Determine whether another application is using port 2049 then free up the port for NFS. Public domain tools, such as **lsof**, can be used to investigate the application currently using port 2049.

9.3.10.2 Mount Daemon will not start

Review the HPSS log for messages that help explain the nature of the problem and then proceed as discussed below.

Diagnosis **1**: The Mount Daemon was unable to read the HPSS exports file. This problem is similar to the NFS Daemon not starting.

Resolution: Check for file existence, permissions, and correctness.

Diagnosis **2**: The HPSS root **uid** does not have an account in the DCE security registry.

Resolution: Use **rgy_edit** to verify whether an accounts exists.

Diagnosis **3**: The Mount Daemon was unable to open its SFS configuration file.

Resolution: Check for file existence and permissions.

Diagnosis **4**: Security initialization failed.

Resolution: Review the HPSS log for security subsystem messages and take appropriate action.

Diagnosis **5**: Errors occurred while initializing DCE/Encina.

Resolution: Turn on **Trace** and **Debug** record types if they are not set and attempt to restart the Mount Daemon. If the server is having trouble registering its service, make sure the Mount Daemon CDS directory exists and that it has a proper ACL.

Diagnosis **6**: The Mount Daemon was unable to open the NFS Server SFS configuration file.

Resolution: Check for the file's existence and determine whether the Mount Daemon DCE principal has permission to read. If not using the default NFS descriptive name, set the **HPSS_NFS_DESC_NAME** environment variable in the **/usr/lpp/hpss/config/hpss_env** file.

Diagnosis **7**: The NFS Mount Daemon will not remain up.

Resolution: Make sure the BFS and the NS are running.

9.3.10.3 NFS server is unable to recover data cache

Diagnosis **1**: During recovery, if a bitfile is noticed to have been removed through other means, such as FTP, it is skipped and the remaining files are recovered. Errors during recovery are of two types: retryable and fatal. Retryable errors include communication errors, out of memory conditions, single bitfile errors, invalid NFS Daemon configuration parameters, and most other errors.

Resolution: When in doubt, assume an error is retryable. The underlying cause of the error should be fixed, if it is not transient, and the NFS Daemon should be restarted. For example, if a particular bitfile always returns an error during recovery, it should be removed, for example with **ftp**. Usually if a single bitfile is returning an error, the remaining files in the cache will be recovered properly before the NFS Daemon terminates.

Diagnosis **2**: Fatal errors include invalid checksums on either the checkpoint or cache file headers, missing checkpoint and / or cache files, and any retryable error that does not respond to the Diagnosis 1 Resolution above.

Resolution: For fatal errors, remove any remaining checkpoint and cache files and restart the NFS Daemon. Any unrecoverable information will be lost.

9.3.10.4 NFS server is unable to recover credentials map

Diagnosis: The credentials map file specified in the NFS configuration does not exist, does not allow access by the NFS UNIX UID, or is corrupted.

Resolution: Check the NFS configuration for a correct credentials dump file. If the configuration is correct and the file exists, view the file contents for corruption. If the file looks corrupted, delete the file and use the **/usr/bin/touch** binary to re-create the file. Then restart the NFS server.

9.3.10.5 Data transfer performance is slow

Several warning messages may be logged by the NFS Daemon that may indicate that the daemon's data cache configuration needs modification. Normally these messages only appear under heavy read / write load. If they start appearing often, follow the directions below.

Diagnosis **1**: "No free memory buffers available" message appears.

Resolution: This message indicates that all of the available memory allocated for handling read/write requests to and from HPSS is in use. While this temporary condition exists, requests will return a "busy" result and will be retried later, degrading performance. If this message appears often, increase the **Memory** buffers field. Since the total memory allocated is the number of memory buffers multiplied by the **Buffer Size** field, you might have to decrease the **Buffer Size** field if not enough real memory is available. It is a good idea to keep the **Buffer Size** field a power of two, with a minimum of half a megabyte.

Diagnosis **2**: "No free cache entries available" message appears.

Resolution: This message (a symptom that the entire cache is filled with information that needs to be written to HPSS) indicates that the NFS Daemon is having trouble writing information to HPSS faster than it is coming in from the client. While this temporary condition exists, requests will return a "busy" result and will be retried later, degrading performance. If this message appears often, there are several things that may be tried.

- Increase the **Cache Entries** field. If no more disk space is available, it might be necessary to also decrease the Buffer Size field. It is a good idea to keep the **Buffer Size** field a power of two, with a minimum of half a megabyte.

- Decrease the **Thread Interval** field, but not too small because performance will degrade as the system searches for entries to write to HPSS more often. The average period between cache entries written to HPSS is the **Thread Interval** field value divided by the number in the **Cleanup Threads** field.

- Increment the **Cleanup Threads** field by one. The average period between cache entries written to HPSS is the **Thread Interval** field value divided by the number in the **Cleanup Threads** field.

Diagnosis **3**: "Having trouble creating free data cache entries" message appears.

Resolution: The resolution is the same as for Diagnosis 2 above except that it may also be beneficial to increase the **Memory Buffers** field. If it does not appear that the data cache configuration is the problem, check the network configuration. Check for socket buffer overflows in the output to a **netstat -s** command. If this is happening, check the maximum socket buffer size by looking at the **sb_max** parameter in the output from the **no -a** command. If you increase this parameter, you will see a reduction in the socket buffer overflows.

9.3.10.6 **ls/stat/pwd** *performance is slow*

Diagnosis: This problem can be caused by inadequate memory storage for the attribute cache or inadequate cache hold time.

Resolution: Look at the NFS statistics **Directory Cache** and **Header Cache** blocks for the **Hits** and **Faults** fields. If the **Faults** field is constantly increasing and the NFS requests are not creating objects (for example, through **create**, **link**, **mkdir**, or **symlink** requests), adjust the values for **Directory Size**, **Table Entries**, **LRU Max Len**, and **Holdtime**. Refer to Chapter 5, HPSS Configuration, for suggested values. If reconfiguring the attribute cache does not help performance, check the network performance and SFS configuration.

9.3.10.7 Bitfiles written using NFS are not flushed to HPSS

Diagnosis: This problem can occur if the recover data cache option is not in the NFS configuration.

Resolution: Check the NFS configuration to verify whether the **Recover Cached Data** field is set. If it is not set, set it and restart the NFS server. If the field is set, check the status of the BFS and SS.

9.3.10.8 NFS top-level storage class is running out of space and migrate/purge does not reclaim space

Diagnosis: This problem can occur when the NFS server is shut down without flushing the data cache and files that were being read / written are deleted when the NFS server is restarted.

Resolution: Look at the BFS segment checkpoint SFS file. If it contains records, cycle the BFS.

9.3.11 Startup Daemon Problems

9.3.11.1 A server refuses to start when requested from SSM

Diagnosis: If a server refuses to start up and the log message indicates that the problem is in the **InitServer** function, check for lockfile contention.

Resolution: The **InitServer** function creates a lockfile on the server's host in the **/var/hpss/tmp** directory. The lockfile name is of the form **hpssd**.**NNNN.AAAA** where **NNNN** is a hexadecimal number and **AAAA** is the descriptive name of the server. If the server's real descriptive name contains imbedded spaces, the lockfile name will substitute underscores for the spaces. The lockfile contains the process ID of the server. The Startup Daemon on each host uses the lockfile to determine whether the server is currently running.

If the server was previously run under a different UNIX UID, it is possible that the lockfile already exists but the current server does not have permission to overwrite it. Check the owner and permissions on the file and the UNIX UID for the server in the generic system configuration file. If you are certain the server is not already running on that host, remove the lockfile and retry starting the server.

9.3.11.2 Cannot start a server

Diagnosis **1**: The server may already be running. The Startup Daemon has determined that an identical copy of the target server is already running.

Resolution: Make sure that there are not two servers with the same descriptive name (this should not be possible). If you force the server to run anyway, you may damage the HPSS system, so stop the old server first.

Diagnosis **2**: A lock file name collision may exist.

Resolution: On very rare occasions, two servers with different descriptive names will share the same lock file name. This can only happen if the first 22 characters of the descriptive names are identical,

and even then the problem occurs only rarely. To fix the problem, change the descriptive name of one of the servers.

Diagnosis **3**: The HPSS executable may not exist or may not be accessible.

Resolution: Make sure the path to the executable is specified correctly, that the executable exists, and that the UNIX user under which the server will be running has permission to access the executable.

Diagnosis **4**: The UNIX user may not exist. The Startup Daemon issues the "Cannot start server; no such unix user <userid>" error message. The SSM System Manager issues the "SSMSM unauthorized to access hpssd_Start_Server API ..." error message.

Resolution: Make sure the server's user name exists in the passwd file on the computer where the server will be running.

Diagnosis **5**: The Startup Daemon may not be running.

Resolution: The daemon must be started before HPSS servers can be brought up. To start the daemon, run the script **rc.hpss**.

Diagnosis **6**: There may be a problem with a Startup Daemon lock file.

Resolution: See the discussion below in Section 9.3.11.4 on how to check lock files.

Diagnosis **7**: The Startup Daemon may not be responding to requests.

Resolution: Kill the daemon using the **kill -9** command and then restart it using the script **rc.hpss**. Do this only as a last resort because it causes the daemon to lose some of the information it has about which servers are running.

9.3.11.3 Cannot stop a server

Diagnosis **1**: The target server may not be able to shut down gracefully.

Resolution: A server may have received a request to shut down, but cannot complete the request for some reason. To fix the problem, use the **force halt** button to force the server to shut down. This should only be done as a last resort when it is clear that the server will never complete a graceful shutdown.

Diagnosis **2**: This diagnosis applies only to stopping a server with the **force halt** button; it is not an issue for the **shutdown** button.

Resolution: To halt a server, SSM issues two requests: one to the specified server directing the server to halt immediately, and one to the Startup Daemon on the server's host directing the Startup Daemon to kill the server. Either request alone should be sufficient, but both are issued in case either fails. If SSM cannot communicate with the server, or if the server ignores the **halt** request, the only way the **halt** can succeed is if the Startup Daemon kills the server. In this case, if the Startup Daemon is not executing and communicating with SSM, the **halt** request will fail.

Diagnosis **3**: The Startup Daemon may not be running.

Resolution: The daemon must be started before HPSS servers can be stopped. To start the daemon, run the script **rc.hpss**.

Diagnosis **4**: There may be a problem with a Startup Daemon lock file.

Resolution: See the discussion in Section 9.3.11.4 below on how to check lock files.

Diagnosis **5**: The Startup Daemon may not be responding to requests.

Resolution: Kill the daemon using the **kill -9** command and then restart it using the script **rc.hpss**. Do this only as a last resort because it causes the daemon to lose some of the information it has about which servers are running.

9.3.11.4 A problem exists with a Startup Daemon lock file

Diagnosis **1**: The file may be empty.

Resolution: Delete the file. The name of the file that is causing the problem can usually be found in the log. You can also use the **ls -l** command to look for empty files in **/var/hpss/tmp**.

Diagnosis **2**: The server may not be able to access the file.

Resolution: This might happen if you have started the server manually under your own UID and then restarted it from the SSM. To fix the immediate problem, use the **chown** command. To prevent this problem from happening in the future, ensure that all accounts that run the server belong to the same group.

Diagnosis **3**: The file may contain invalid information.

Resolution: To better understand the situation, use the **cat** command to view the contents of the lock file. It should look similar to this:

DescName: Mount Daemon LockNum: 0 PID: 20016

The descriptive name must match the name of the file (in this example, **/var/hpss/tmp/hpssd.4302.Mount_ Daemon**.) If the names do not match, change one of the daemon's descriptive names to avoid a name collision. To avoid further trouble, delete the lock file.

9.3.12 SSM Problems

9.3.12.1 The SSM windows have incorrect colors, sometimes to the point of being unusable

Diagnosis: Sammi is not able to obtain all the X color resources that it needs.

Resolution: Check to see what other applications have active windows on the workstation where SSM is being run. Some applications which use a large number of colors prevent SSM from allocating the colors it needs, causing the SSM windows to appear all black or all white, for example. The Netscape Web browser is one application known to cause this problem.

If suspect applications are found, try shutting them down and restarting SSM to see if the color problem goes away. Sometimes SSM can coexist with these applications if SSM is started first, and the other application second.

9.3.12.2 Objects on the SSM windows are missing or not positioned properly

Diagnosis: X resources required by Sammi are not being set correctly.

Resolution: Make sure you have a copy of the file named **SAMMI** (from **/usr/lpp/hpss/config/templates/SAMMI.template**) in your home directory on the host where the Sammi Runtime is executing. If you have a private X **app-defaults** area, the file should go there instead of in your home directory. If Sammi is currently running, shut it down and restart it.

9.3.12.3 Sammi and the Data Server refuse to connect to one another on SSM session startup

Diagnosis **1**: Problems exist in the Sammi configuration file or the API configuration file.

Resolution: The Sammi configuration file is usually named **ssm_console.dat**, and each user should have a private copy customized for a unique console ID. This file is usually located in the **/usr/lpp/hpss/sammi/ssmuser/<userid>** directory. Near the bottom of this file are several lines beginning with **logical_server**. For the three vital Sammi processes (**s2_evtsvr**, **s2_mstalm**, and **s2_stream**), note the hexadecimal RPC addresses in field 3 of each line. Also make sure that the RPC version numbers (field 4) are equal to the console ID, which is set in the **console_id** line near the top of the file. There should also be a line for the Data Server (**ssm_ds**). Note this RPC address, also, and make sure that the RPC version number is 1.

Now check the API configuration file named **api_config.dat**. There is only one copy, located in **/usr/lpp/hpss/bin**. This file contains multiple sets of three **logical_server** lines, one set for each Sammi console ID that has been set up by the HPSS administrator. The three lines in each set refer to the same three Sammi processes (**s2_evtsvr**, **s2_mstalm**, and **s2_stream**) configured in the Sammi configuration file. There should be one set of lines for which the RPC version field (field 4) matches the console ID from the Sammi configuration file. The RPC addresses (field 3) in these lines must be identical to the RPC addresses for the corresponding Sammi processes in the Sammi configuration file.

Diagnosis **2**: Startup script problems exist.

Resolution: If both configuration files are correct (see Diagnosis 1 above), check the **HPSS_PORT_SSMDS** environment variable defined in the **/usr/lpp/hpss/config/hpss_env** file. The **HPSS_PORT_SSMDS** variable defines the hexadecimal RPC address of the Data Server. This address must match the RPC address for **ssm_ds** defined in the Sammi configuration file **(ssm_console.dat**).

9.3.12.4 Sammi and the Data Server connect to one another, but they generate many error messages and the Data Server cannot write to most fields

Diagnosis: Hostname resolution problems exist.

Resolution: The Sammi configuration file and the API configuration file (see Section 9.3.12.3 above) both contain hostname information in field 5 of their **logical_server** entries. The Sammi process lines in both files must reference the host where the Sammi runtime is executing. The Data Server (**ssm_ds**) line in the Sammi configuration file must reference the host where the Data Server is executing. Normally these host names should be **localhost** or the name of a remote host, as appropriate. Using **localhost** where possible can make Sammi operation more efficient.

If, however, your site uses Domain Name Services (DNS) for hostname resolution, you may encounter Sammi communication problems related to these hostname fields. Certain quirks in Sammi and / or DNS sometimes make it impossible for Sammi to resolve a hostname unless it is in a specific form. This problem usually becomes apparent when Sammi comes up and appears to connect normally to the Data Server, but then the Data Server finds it impossible to write data to fields on the SSM windows. Error messages generated by Sammi may also appear on the TTY used to start Sammi.

Unfortunately, there is no clear-cut method for fixing the problem; the technique involves some trial and error. You may have to tinker with hostnames in the configuration files. In some cases, you may have to change **localhost** to the actual fully-qualified hostname (for example, **node1.mysite.gov**). In other cases, the full name may not work, but the abbreviated hostname may work (using the same example, **node1**). You may find that one hostname format will work in the Sammi configuration file, but something else is required in the API configuration file.

The only way to find the right combination is to edit one or both configuration files and then restart Sammi and see what happens. If the problem still exists, try a different hostname format in one or both files, and then shut down Sammi and restart again. Repeat until the Data Server is able to write data to fields normally.

9.3.12.5 Sammi and the Data Server start up normally, but get disconnected in the middle of operations

Diagnosis **1**: The network or execution hosts are overloaded. This problem usually arises when data fields in windows suddenly get overwritten with red error messages reading “?unconnected dfd?”.

Resolution: If the network or SSM host machines are overloaded, communications problems may occur between Sammi and the Data Server in the middle of an SSM session. If this is truly the cause of the problem, you may need to take steps to free up machine or network resources.

Diagnosis **2**: RPC timeouts are too small. As in Diagnosis 1, the data field can be overwritten.

Resolution: The Sammi configuration file and the API configuration file (see Section 9.3.12.3 above) both contain RPC timeouts. These appear as the last two fields of the **logical_server** lines. The numbers are timeouts in seconds, and the second number on each line should always be at least 10 seconds larger than the first number.

The timeouts in the Sammi configuration file define the maximum time Sammi will wait with no response from a server before deciding that the server is disconnected. The timeouts in the API configuration file define the maximum time the Data Server will wait with no response from Sammi before deciding that Sammi is disconnected. Increasing these timeouts can help to avoid too-hasty disconnections in overloaded environments. Increasing them too much, however, will delay reporting of hard communications failures.

9.3.12.6 Communications problems exist between the Data Server and the System Manager

Diagnosis **1**: The number of servers running is incorrect.

Resolution: There must be one and only one System Manager in execution at any one time per HPSS installation. Several Data Servers may execute concurrently, but only if they are using different CDS names. Note that the **start_ssm** script will refuse to start either program if it finds a copy already running on the same host; however, the administrator may create customized start scripts to start more than one Data Server on the same host as long as the administrator makes sure each server uses different CDS names and different ports to talk to Sammi.

The **ps** command can be used on each host to determine whether there are more copies of the System Manager or Data Server executing than intended.

Diagnosis **2**: CDS entries are incorrect.

Resolution: The CDS names are set by environment variables in the **/usr/lpp/hpss/config/hpss_env** file. The CDS directory specified for the Data Server must exist and must contain a security object with a proper ACL before the Data Server is executed. The directory and security object will have been created by the HPSS Infrastructure Configuration script (**mkhpss**). To configure a second Data Server, you must create its CDS directory and its security object and set the proper ACLs on them using regular DCE administration tools (**cdscp** and **acl_edit**). The directory should be accessible by all HPSS servers; the security object should give all permissions to the SSM DCE principal and all permissions except control to the other HPSS servers. By default, the DCE registry group for all HPSS servers is **hpss_server** and the DCE principal for SSM is **hpss_ssm**.

For example, if you wish to configure a new Data Server to use CDS directory **/.:/hpss/ssmds_2**, use the following commands:

```
cdscp create directory /.:/hpss/ssmds_2
acl_edit /.:/hpss/ssmds_2 -m group:hpss_server:rwdtcia
acl_edit -e /.:/hpss/ssmds_2 -m group:hpss_server:rwdtc
acl_edit /.:/hpss/ssmds_2 -io -m group:hpss_server:rwdtc
acl_edit /.:/hpss/ssmds_2 -ic - m group:hpss_server:rwdtcia
cdscp create object /.:/hpss/ssmds_2/Security
acl_edit /.:/hpss/ssmds_2/Security -m user:hpss_ssm:rwdtc
acl_edit /.:/hpss/ssmds_2/Security -m group:hpss_server:rwdt
```

The System Manager will create its own CDS directory and security object, if they are not already there, using the CDS directory specified in the **hpss_env** file. This is how the System Manager bootstraps itself the first time it is executed after initial installation.

Diagnosis **3**: The programs are not running or have not registered.

Resolution: Each program logs a message if it cannot register. In addition, on successful registration the System Manager logs a message stating the CDS name and Descriptive Name it is using. The **rpccp show mapping** command, used on the host where each program is executing, will show whether the program is registered in the DCE rpc endpoint map there and, if so, which CDS name it is using. If either program has not registered with its endpoint mapper, the other program will not be able to contact it.

The **ps** command can be used on each host to determine whether the servers are running.

Diagnosis **4**: Servers are not finding each other.

Resolution: When the System Manager can connect to the Data Server, it logs a message saying "Network communications established," and specifies the CDS name of the Data Server. When the System Manager cannot reach the Data Server, it logs an error message specifying the CDS name it thinks the Data Server is using. If the System Manager is logging these error messages, make sure the CDS name it is trying is the same one under which the Data Server is actually registered.

When the Data Server has successfully connected to the System Manager, it logs a message saying "SSM Data Server has completed DCE bind to System Manager." See Diagnosis 2 above for more information.

Diagnosis **5**: The Data Server has been automatically checked out.

Resolution: The System Manager will automatically check out any Data Server which it cannot establish contact with within a certain interval.This interval is hard coded as *5 minutes*. After checking out the Data Server, the System Manager will no longer send it any notifications. If the screens appear to be stale and are not receiving any updates at all, try forcing a **CheckIn** from the **Admin Menu.** If this does not remedy the situation, check the log to see whether the System Manager is trying to check the Data Server out or has already done so.

Note that just because the Data Server can contact the System Manager, this does not mean that the System Manager can contact the Data Server. Two different DCE interfaces are involved here. One is advertised by the Data Server as a service for the System Manager. This interface allows the System Manager to send alarms, events, status messages, data change notifications, server list changes, drive list changes, and storage class list changes to the Data Server. The other interface is advertised by the System Manager as a service for the Data Server. This interface allows the Data Server to request reads and modifications of SFS files, current copies of managed objects from servers, startup and shutdown of servers, and most other administrative functions the Data Server performs on behalf of the user. It is when the System Manager cannot contact the Data Server on the first interface for a specified interval that it automatically checks it out. The other interface may be working correctly; for those functions, the Data Server appears responsive.

If the Data Server is automatically checked out, try to check it in again using the **CheckIn** function under the Admin Menu on the HPSS Health and Status window (Chapter 6, Figure 6-1). You may need to repeat this process several times. If this still does not correct the problem, you may need to restart the Data Server and / or System Manager.

9.3.12.7 Communications problems exist between the System Manager and other HPSS servers.

Diagnosis: Servers cannot find the System Manager.

Resolution: If the basic server configuration file contains more than one entry for the SSM System Manager, or if it contains one entry that does not match the values set by the environment variables in the **/usr/lpp/hpss/config/ hpss_env** file, the System Manager might not register under the name expected by the other HPSS servers, or if it does, it might not use the UUID they are expecting. If this is true, the other servers will not be able to find the System Manager and send it any notifications. This situation can happen if the administrator has added additional entries of type SSM to the configuration file using the GUI screens. It can also happen if the administrator runs

SSM using the **start_ssm** script, changes the System Manager descriptive name specified in the **hpss_env** file, and then reruns the System Manager.

The System Manager attempts to use the descriptive name and CDS name specified for it by the environment variables in the **start_ssm** script. If the basic server configuration file has no entry at all for the System Manager, the System Manager creates one using the values specified in the startup script. This is the normal behavior the first time SSM is executed after initial installation. If there are one or more System Manager entries in the configuration file, but none matches the descriptive name for the System Manager specified in the **hpss_env** file, the entries are ignored, and a new entry is created and used just as if none existed at all. If there is an entry in the configuration file that matches the System Manager descriptive name from **start_ssm**, but the CDS name does not match, the System Manager uses all the information from the entry except the CDS name. It registers under the CDS name specified in the **hpss_env** file, but it does not try to create a new SFS entry or write over the existing one.

In all cases, the System Manager logs which descriptive name and CDS name it is using at startup time, and logs any extra SSM entries it finds in the file.

The other HPSS servers find the System Manager by the **SSM ID** field in their own basic server configuration file entries. They will look up the System Manager in the file using this UUID as an index. If they find the entry, they will look for the System Manager under the CDS name specified by that entry.

If the administrator brings up SSM initially after installation, adds several server entries, and then adds a new SSM entry, the new SSM entry will have a different UUID than the original. In this case, any new servers added after this might get the value of the old or of the new SSM in the **SSM ID** field. However, if the administrator runs SSM, changes the descriptive name for the System Manager in the **hpss_env** file, and then runs SSM again, SSM itself will add a new SSM entry with a different UUID. In this second case, SSM will run under the new identity and the other servers will be looking for it under the old identity. In the first case, chances are even that the new servers will look for SSM under the old identity.

The point is that there should be exactly one entry for SSM in the basic server configuration file. It is worth an occasional audit of your system to ensure this. Simply bring up the HPSS Servers window (Chapter 6, Figure 6-2) and sort the list by **Type**. Check to make sure there is only one server of type **SSMSM**.

If you need to change the descriptive name or CDS name for SSM, first bring up SSM using the old variable value defined in the **hpss_env** file and change the necessary fields in the basic configuration for the SSM System Manager. Then immediately shut down SSM, modify the variable in the **hpss_env** file to match, and restart SSM.

For additional information that may help with this problem, see Sections 9.2.1.1 and 9.2.1.4.

9.3.12.8 HPSS utilities started from SSM do not run correctly

Diagnosis: Path names and / or environment variables are incorrectly set.

Resolution: Some HPSS functions are provided by command line utilities that can be run either directly from the shell, from a script or cron job, or from SSM. SSM determines the path name for these executables from the **/usr/lpp/hpss/config/ hpss_env** file. If SSM cannot start one of these

utilities, make certain the path name in the **hpss_env** file is correct and that the permissions on the executable are correct.

In addition, SSM uses the DCE command line utilities **cdscp** and **acl_edit**, and their path name must also be defined in the **hpss_env** file.

If any path name is not defined in the appropriate environment variable, the System Manager will use a hard-coded default name for that path.

For HPSS Release 4.1.1, there is no way to change SSM's definition of these path names without modifying the **hpss_env** script and then using the **start_ssm** script to restart the System Manager. The default path names for the HPSS utilities are defined as follows:

ENVIRONMENT VARIABLE:	UTILITY:	DEFAULT PATH NAME:
HPSS_EXEC_ACL_EDIT DCE	acl_edit program	/bin/acl_edit
HPSS_EXEC_CDSCP DCE	cdscp program	/bin/cdscp
HPSS_EXEC_ACCT HPSS	accounting program	/usr/lpp/hpss/bin/hpss_acct
HPSS_EXEC_DELOG HPSS	delog program	/usr/lpp/hpss/bin/hpss_delog
HPSS_EXEC_RECLAIM HPSS	reclaim utility	/usr/lpp/hpss/bin/reclaim.ksh
HPSS_EXEC_REPACK HPSS	repack utility	/usr/lpp/hpss/bin/repack

Note that the **reclaim** utility is implemented as a shell script, which invokes a binary executable. The **HPSS_EXEC_RECLAIM** variable should be set to the shell script path name. Make sure the shell script specifies the proper path name for the binary.

9.3.12.9 Configuration file errors exist

Diagnosis **1**: The number of basic server configuration files is incorrect.

Resolution: There must be one and only one basic server configuration file per HPSS installation. This file name is defined by the **CF** environment variable in the **/usr/lpp/hpss/config/ hpss_env** file and is passed to each server started by the System Manager.

Diagnosis **2**: The number of Accounting Policy files is incorrect.

Resolution: There must be one and only one accounting policy file per HPSS installation. This file name is defined by the **HPSS_CONFIG_ACCT** environment variable in the **hpss_env** file.

Diagnosis **3**: The number of server-specific configuration files is incorrect.

Resolution: It is not required but is recommended for simplicity that there be one and only one server-specific configuration file of each type per HPSS installation. This includes the BFS, NS, DMAP Gateway, Log Client, Log Daemon, Metadata Monitor, MPS, Mount Daemon, MVR, NFS Daemon, PVL, PVR, and SS configuration files. These file names are specified in the basic server configuration entry for each server.

Diagnosis **4**: Other files are being written by servers.

Resolution: Several servers maintain other SFS files to keep track of their resources, such as drives or media. In some cases, these SFS files may be shared by all servers of the same type; in other cases, each server must have its own private copy of each file.

The Movers may all share the same device file. In fact, it is not required but is recommended that there be one and only one MVR device file per HPSS installation. The name of the device file is specified for each MVR in its MVR configuration file entry.

Each PVR must have its own private cartridge file, not shared with any other PVR. The name of the cartridge file is specified in the PVR specific configuration file.

Each SS must have its own private storage map, storage segment, physical volume, and virtual volume files, not shared with any other SS. The names of these files are specified in the SS specific configuration file.

Each MPS must have its own private checkpoint file, not shared with any other MPS. The name of the checkpoint file is specified in the MPS specific configuration file.

Diagnosis **5**: The number of policy files shared by more than one server is incorrect.

Resolution: There must be one and only one Class of Service, hierarchy, log policy, migration policy, purge policy, and storage class file per HPSS installation, and these must be defined by environment variables in the **/usr/lpp/hpss/config/ hpss_env** file. If the appropriate environment variable is not set for any file name, SSM will use a hardcoded default for that name. These files are each shared by more than one HPSS server; for example, both the SS and the MPS read the storage class configuration file, and each Log Client reads the same log policy configuration file.

The servers that use these files are not aware of the SSM environment variables, but read the SFS file names from their individual configuration files. It is therefore possible for SSM to write to one file and for the server to try to read a different file. For instance, the MPS configuration file includes a definition for the migration policy file name. If the name in the MPS configuration file for the migration policy file does not match the name defined in the **hpss_env** file, the MPS and SSM will try to use different migration policy files. The names of these files in the **hpss_env** file and in the individual server configuration files *must* match.

For Release 4.1.1 of HPSS, there is no way to change SSM's definition of these file names without modifying the **hpss_env** file and then using the **start_ssm** script (which reads the **hpss_env** file) to restart the System Manager. Refer to Section 4.3 for the descriptions and default values defined for these SFS file names.

9.3.13 Location Server Problems

9.3.13.1 Location Server fails to start up

Diagnosis 1: The Location Server fails to startup for the first time after the HPSS system is initially set up due to a missing DCE RPC group.

Resolution: During testing the Location Server sporadically failed to create the DCE RPC group just after it's initial setup on some machines. You should go ahead and create an empty RPC group in DCE yourself. You should only need to do this one time. If you use the default values, and your keytab file is setup properly, you can force the creation of the group by entering the following commands.

```
% dce_login -k /krb5/hpss.keytabs hpss_ssm
```

```
% dcecp -c rpcgroup create /.:/hpss/ls/group

% exit
```

Diagnosis 2: The Remote HPSS Site file has not been created.

Resolution: Create an empty Remote HPSS Site metadata file. If the file is missing, the Location Server assumes there is a problem with the configuration and will not start up. Note that this file must exist even if you don't plan on entering any remote HPSS site information.

Diagnosis 3: The Location Server is unable to determine the Root Name Server

Resolution: There must be exactly one Root Name Server defined for client requests to be processed. Either no Root Name server has been defined or too many have been defined.

9.3.13.2 Clients unable to contact running Location Server

Diagnosis 1: Clients don't know the correct CDS path to the Location Server's DCE RPC group.

Resolution: Make sure each client has the CDS path of the DCE RPC group in it's environment. If you change this on the LS Policy screen, make sure you change it in the environment as well. The environment variable is **HPSS_LS_NAME**. See section 5.8.1 for more information.

Diagnosis 2: The Location Server is unable to contact a server during initialization.

Resolution: If there are any Bitfile Servers or remote Location Servers that are down, bring them up. The Location Server waits several minutes during initialization for servers to become available. At least on BFS must be up for it to be able to process requests.

9.3.13.3 Clients taking a long time to contact a replicated Location Server

Diagnosis 1: A replicated LS has crashed or was force halted and has not been restarted.

Resolution: Clients will be degraded until the replicated LS is brought back up or its name is removed from the Location Server DCE RPC group. If you want to keep the replicated LS down, remove it's entry from the Location Server's DCE RPC group. Another method of clearing this up is to bring the replicated server back up and then perform a normal shutdown.

9.3.13.4 Clients can't see or create files at a remote site.

Diagnosis 1: A server at the remote site is down.

Resolution: Verify that all Location Servers, Bitfile Servers and Name Servers are executing at the remote site.

Diagnosis 2: Remote site is improperly defined.

Resolution: Clients are unable to either see or create files on a newly defined remote HPSS site. Make sure each remote site is defined in the Remote HPSS Site file and that Cross Cell Trust has been established if the site is located in a remote DCE cell. Also make sure the remote site's administrator

has obtained your HPSS site's information. Each site uses the Remote HPSS Site file to determine all remote sites it needs to contact.

9.3.13.5 Location Server is unable to contact a server

Diagnosis 1: The server is not executing.

Resolution: Make sure the server is running or marked non-executable.

Diagnosis 2: Location Server connections to other servers are timing out.

Resolution: Raise appropriate timeout interval on the Location Policy screen. If you have many servers of the same type, raise the appropriate maximum update thread count on the Location Policy screen as well. Determine the root cause of why time outs are occurring.

Diagnosis 3: A new server is in the process of being defined and is marked as executable.

Resolution: Mark the server as non-executable. The Location Server assumes that any server marked as executable should be running. In the case of Bitfile Servers and Location Servers, it will periodically continue to try to contact them as long as they are marked executable.

9.4 HPSS User Interface Problems

The paragraphs below discuss interface problems with the Client API, FTP Daemon, and NFS problems.

9.4.1 Client API Problems

For a description of how to obtain detailed information regarding errors encountered by the Client API, refer to the section on Client API Configuration in Section 5.8.1.

9.4.1.1 Client API cannot connect to either the HPSS NS or BFS

Diagnosis 1: The Client API is not using the correct name for the Location Server RPC group.

Resolution: Verify the Location Server RPC group name being used by the HPSS system. If this name is the default RPC group name, verify that the environment variable **HPSS_LS_NAME** is not set. If the RPC group name is other that the default, use the environment variable as specified in the Client API configuration section, or the **hpss_SetConfiguration**() call as specified in the *HPSS Programmer's Reference Guide, Volume1, Release 4.1.1*.

9.4.1.2 Client API cannot initialize its security context

Diagnosis **1**: The Client API cannot find the keytable entry for the principal **hpss_client_api** in the HPSS keytable file.

Resolution: Verify that an entry exists for the principal **hpss_client_api** in the HPSS **keytable** file.

Diagnosis **2**: The Client API cannot access the HPSS **Client keytable** file.

Resolution: Ensure that the HPSS **Client keytable** file's permission allows read access to the Client API.

Diagnosis **3**: The Client API is not accessing the correct HPSS keytable file.

Resolution: Verify the name of the HPSS keytable file. If the name is the default, verify that the environment variable **HPSS_KTAB_PATH** is not set. If the name is other than the default, use the **HPSS_KTAB_PATH** environment variable to specify the name of the file to use, as specified in the Client API configuration section, or use the **hpss_SetConfiguration()** call as specified in the *HPSS Programmer's Reference Guide*, Volume 1, Release 4.1.1.

Diagnosis **4**: The currently established login context for the client is associated with an account in the DCE registry that refers to an alias.

Resolution: The DCE registry account that the client is using should be changed to refer to a valid principal, or the client should use another account that refers to a valid principal.

9.4.2 FTP Daemon Problems

For a description of how to obtain detailed information regarding errors encountered by the FTP Daemon, refer to the section on FTP Daemon configuration in Section 5.8.3.

9.4.2.1 FTP Daemon cannot connect to the HPSS NS, BFS

Diagnosis **1**: The FTP Daemon is not specifying the correct name for the Location Server RPC Group.

Resolution: Verify that the "L" option is specified on the HPSS FTPD startup line. Verify that the hpss_option LS specifier in the ftpaccess file is correct. Verify that the correct ftpaccess file is being requested – default: /var/hpss/ftp/etc/ftpaccess. Verify that the Location Server, Name Server, and Bitfile Servers are all active.

Diagnosis **2**: The FTP Daemon command line as specified in the **/etc/inetd.conf** file extends beyond the limit of input that will be read by the **inetd** daemon; therefore, command line arguments may be truncated.

Resolution: Shorten the length of the line in the **/etc/inetd.conf** file, either by removing some arguments or shortening path names or argument lengths.

For additional information that may help with this problem, see Sections 9.2.1.1 and 9.2.1.4.

9.4.2.2 The user cannot log into the FTP Daemon

Diagnosis **1**: The user does not have a valid entry in the FTP password file (/**var/hpss/ftp/etc/ftppasswd**).

Resolution: Insert the user's info into the FTP password file then use the **hpss_pftppw** utility to create the user's password. Refer to Section 5.8.3 for more information on FTP configuration.

Diagnosis **2***:* The user info is not in the DCE Security Registry.

Resolution: Use the **hpssuser** utility to create the DCE user account. It is NOT recommended that the **passwd_import/passwd_export** utilities be used!

Diagnosis **3**: The user name is associated with an account in the DCE Registry that refers to an alias (instead of a principal).

Resolution: The DCE account that is associated with the user name should be changed to refer to a valid principal, or the client should use another user name that refers to a valid principal.

9.4.2.3 All user file access and file creation is based on user id hpss_ftp

Diagnosis: The principal **hpss_ftp** does not have control permission in the Name Server's security object ACL.

Resolution: Update the ACL for the Name Server's security object to provide **hpss_ftp** with control permission.

9.4.2.4 FTP file transfer performance is poor

Diagnosis 1: The buffer size being used by the FTP Daemon is limiting the file transfer performance (this affects non-parallel transfers - i.e., "put", "get" and "append").

Resolution: Adjust the size of the buffers used by the FTP Daemon when transferring data using the **-b** flag as specified in the FTP Daemon configuration section, Section 5.8.3.

Diagnosis 2: The FTP Daemon is not using a high performance network for communication with the HPSS Movers (this affects non-parallel transfers - i.e., "put", "get" and "append").

Resolution: Modify the FTP Daemon configuration to use a higher performance network for data transfers to and from the HPSS Mover using the "-h" flag as specified in the FTP Daemon configuration section, Section 5.8.3.

9.4.3 NFS Problems

9.4.3.1 Client is unable to mount the HPSS directory

Diagnosis: The HPSS Mount Daemon or NFS server is not running.

Resolution: Execute the **showmount -e** command to determine if the Mount Daemon is running and if the directory you are trying to mount is exported to your system. Run the **rpcinfo -p** command to determine if the mounted and NFS RPC services are registered with the portmapper. If so, run the **rpcinfo -u** command for the NFS program and version to determine if the NFS server is

responding to NFS requests. If the NFS server is responding, contact the HPSS administrator to determine if the exports options for your client allow your mount request.

9.4.3.2 User is unable to access the NFS mounted directory

Diagnosis: The HPSS Mount Daemon or NFS server is not running.

Resolution: Use the same techniques as described above to determine if the NFS server is responding. Check with the HPSS administrator to determine if the user has permissions to the file (which may be protected through an ACL) and if a user identity mapping is required. If the **UIDMAP** option is set on the export entry, that allows the client system access. Check with the HPSS administrator to determine if the user has a map entry. If not, add an entry according to site policy.

9.4.3.3 RPC timeouts occur on client systems

Diagnosis: RPC timeouts do not, in general, indicate a problem with NFS. Because of the HPSS architecture, timeouts will occur, especially if files being accessed are non-disk media.

Resolution: Client systems can mount HPSS directories using the **hard, intr** mount options. This will cause the client to retry forever. Since NFS hard mounts do cause problems on some systems, soft mounts can be used, but it is recommended that the **timeo** and **retrans** options be used.

9.4.3.4 A stale or bad file system error occurs

Diagnosis: This problem can occur due to a configuration change or error.

Resolution: From the client system, unmount and remount the HPSS directory. If this does not work, check with the HPSS administrator for any inconsistencies between the Mount Daemon and NFS server **encrypt filehandles** and **exports file** fields.

9.4.4 HPSS/DMAP Problems

9.4.4.1 Changes made to DFS are not reflected on HPSS

Diagnosis 1: XDSM kernel extensions have not been loaded.

Resolution: Make sure the line to add XDSM extensions has been added to **rc.dfs**. The line should be: "**/usr/sbin/cfgdmepi -delay 1 -a /usr/lib/drivers/dmlfs.ext"**. If necessary, execute this command manually to verify that the extensions can be loaded.

Diagnosis 2: The HDM is not running.

Resolution: If HDM was not running when the file system was first exported, DFS users will be able to change their files without the corresponding changes being reflected on HPSS. Start HDM. Verify that **rc.dfs** is set up to export file systems only after HDM is known to be running. Also make sure that administrative procedures are set up to stress the importance of having HDM running at all times.

Diagnosis 3: The aggregate type has not been set to **dmlfs**.

Resolution: Run the **dmaggr** utility.

Diagnosis 4: HDM was not restarted after **dmmean** was executed.

Resolution: The **dmmean** utility should only be used in emergencies when end users do not have access to their files. After **dmmean** has been run, restart HDM before exporting file systems.

9.4.4.2 Changes made to HPSS are not reflected on DFS

Diagnosis 1: XDSM kernel extensions have not been loaded.

Resolution: Make sure the line to add XDSM extensions has been added to **rc.dfs**. The line should be: "**/usr/sbin/cfgdmepi -delay 1 -a /usr/lib/drivers/dmlfs.ext"**. If necessary, execute this command manually to verify that the extensions can be loaded.

Diagnosis 2: The aggregate type has not been set to **dmlfs**.

Resolution: Run the **dmaggr** utility.

Diagnosis 3: HDM was not restarted after **dmmean** was executed.

Resolution: The **dmmean** utility should only be used in emergencies when end users do not have access to their files. After **dmmean** has been run, restart HDM before exporting file systems.

9.4.4.3 Can access files from DFS but not from HPSS

Diagnosis 1: The mount points for the fileset were not specified.

Resolution: If the local and/or global mount points were not specified in **filesys.dat**, HDM will prevent the system from changing files through the HPSS interfaces. An unspecified mount point is indicated by **NO_MOUNT_POINT** in **filesys.dat**, and probably indicates the fileset was not configured correctly. To fix these problems, recreate the fileset using the SSM screens, and failing that, edit **filesys.dat**.

9.4.4.4 BFS is accumulating bitfiles that are not being deleted

Diagnosis 1: Files from archived filesets are not being purged often enough.

Resolution: If an aggregate is configured to use **archive/rename**, then when a file is deleted from DFS, it is not immediately deleted from HPSS, but renamed instead. To delete these files, the administrator must periodically run **archivedel**.

9.4.4.5 HDM will not start

Diagnosis 1: Executables are missing or in the wrong directory.

Resolution: Make sure all HDM executables are in the same directory as **hpss_hdm,** and that the mode bits set correctly.

Diagnosis 2: Configuration files are not all in the same directory.

Resolution: Ensure that all five configuration files **config.dat**, **filesys.da**t, **gateways.dat**, **policy.dat**, and **security.dat** reside in the same directory. The files must exist even if they are empty.

Diagnosis 3: The event log files are missing.

Resolution: Verify that the files named by the configuration parameters **MainLogName**, **AclLog**, **DestroyLogName**, and **ZapLogName** exist, even if they are empty. If HDM has been running successfully for a while, but the files are now missing, determine what happened to the files, and if possible, recover them. Failing that, create empty files to replace the ones that are missing.

Diagnosis 4: HDM is waiting.

Resolution: This might happen if the HDM only comes part way up, then waits for something (for example, an event handler to start.) In some cases, HDM will eventually come up, and it is just a matter of waiting long enough. For example, the event dispatchers may have to recover a number of events before they are up. Also, if the HPSS system is down, it will take longer for the HDM to come up.

Diagnosis 5: HDM is still running.

Resolution: This can happen if the HDM is already running, in which case, there is presumably nothing to fix. It can also happen if the HDM did not shut down completely when it was stopped. It could be that the main HDM process is running, or it could be one of the subprocesses that is still running. To fix this problem, use **hdm_admin stop**. If that fails, run **hdm_admin killall**. If that also fails, use the shell's "ps" command to look for processes that are still running and kill them manually.

Once in a while, it may be impossible to kill an HDM processes. In this case, try to force-start the HDM using **hpss_hdm -f config.dat 1** or **hdm_admin start -force**.

Diagnosis 6: The HDM semaphore set may be corrupted.

Resolution: If this happens, the message log should say the HDM could not be started because it was already running, when, in fact, none of the HDM processes are running. This may indicate that the HDM semaphore set has been corrupted, or it may indicate that some non-HDM process is using the same semaphore set. To fix the immediate problem, use the following command (assuming, the default shared memory key, 3789, is being used): **ipcrm -S 0x0ecd**

If the problem persists, use a different shared memory key.

Diagnosis 7: HDM was not correctly shut down.

Resolution: If HDM is not shut down correctly, there may be remnant processes that must be stopped manually.

Diagnosis 8: An event logging file has been corrupted.

Resolution: This could happen if the **destroy**, **main**, or **zap** log contains invalid data. If there has been a disk error, it may be possible to fix the problem, and then restart HDM. If all else fails, delete the

log file, replace it with an empty file, and then restart HDM. Do this only as a last resort, since it can cause the DFS and HPSS name and/or data spaces to become inconsistent.

9.4.4.6 HDM will not stop

Diagnosis 1: An HDM subprocess missed the shutdown signal.

Resolution: When HDM shuts down, it sends signals to its subprocesses, then waits for them to stop. Occasionally a subprocess will not recognize the signal or will be too busy to realize it needs to shut down. To fix this problem, use **hdm_admin killall**. If that fails, use the shell's **ps** command to look for processes that are still running and kill them manually.

Diagnosis 2: HDM is shutting down too slowly.

Resolution: Depending on how busy the system is, HDM may take some time to shut down. This is especially true if a gateway or some other part of HPSS is down. Give HDM enough time to shut down before resorting to more drastic measures.

9.4.4.7 HDM cannot be restarted

Diagnosis 1: HDM will not stop.

Resolution: To fix this problem, use **hdm_admin stop**. If that fails, run **hdm_admin killall**. If that also fails, use the shell's "ps" command to look for processes that are still running and kill them manually.

Once in a while, it may be impossible to kill an HDM processes. In this case, try to force-start the HDM using **hpss_hdm -f config.dat 1** or **hdm_admin start -force**.

Diagnosis 2: HDM has not been successfully stopped before attempting to restart it.

Resolution: Before HDM can be restarted, it must first be stopped. Stopping the HDM can be done directly with **hdm_admin stop**.

9.4.4.8 HDM does not seem to be doing anything

Diagnosis 1: The HDM is waiting for HPSS to respond to a request.

Resolution: If HPSS is not running, or there is a communication problem between HDM and the DMAP Gateway, HDM can spend a long time on an operation before aborting it. To correct this problem, make sure the network is up, and that all HPSS servers are running.

Diagnosis 2: DFS is waiting for HDM to respond to an event.

Resolution: If HDM is slow responding to events, the kernel will enter a backoff mode, where it waits for increasingly longer periods of time for HDM to respond. In some cases, the time can be as much as 17 minutes. To determine if this is the problem, use **hdm_admin mlog** to determine if there are any events that have been in the "waiting" stage for a long period of time. To solve this problem, restart HDM and/or reboot the machine.

To reduce the chance of having this problem, set **cfgdmepi's -delay** parameter to 1. For example: **/usr/sbin/cfgdmepi -delay 1 -a /usr/lib/drivers/dmlfs.ext**

Diagnosis 3: HDM is deadlocked

Resolution: This problem should be rare. Use **hdm_admin locks** to verify that HDM is managing its semaphores correctly. Use **hdm_admin mlog** to determine whether there are any events in the "waiting" stage and/or have been in the log for a long period of time. If necessary, restart HDM.

Diagnosis 4: The Kernel is deadlocked

Resolution: If it appears that DFS is either deadlocked or bogged down in some way, the only safe thing to do is to reboot the system. While it may be possible to free the system using **dmmean**, that is recommended only as a last resort because it invariably causes the DFS and HPSS name and/or data spaces to become inconsistent.

9.4.4.9 Messages are not written to the HDM log

Diagnosis 1: HDM has been told not to log messages.

Resolution: If **MsgFileDirectory** has been left blank, HDM will not produce a message file. In this case, if messages are written, they will be sent to stdout. Change this parameter to specify a message file directory.

Also, make sure the **LogRecordMask** configuration parameter has been set.

Diagnosis 2: There was a problem writing to the log file.

Resolution: If there is a problem writing to the log file (for example, the disk goes bad or fills up), HDM will revert to writing messages to stdout. To fix the problem, change **MsgFileDirectory** to specify the names of a directory where the log can be written, then restart HDM.

9.4.4.10 Files are destroyed slowly or not at all

Diagnosis 1: The destroy log is full.

Resolution: Check the log for messages indicating that the destroy log is full. If the destroy log is full, destroy operations will be delayed until space becomes available in the log. If necessary, increase the value of **DestroyLogSize**. To establish the proper value for **DestroyLogSize**, monitor the message log occasionally, to determine the high water for the number of destroys. If the number grows close to **DestroyLogSize**, increase the size of the log. If the number stays well short of **DestroyLogSize**, decrease the size.

Diagnosis 2: There are too many entries in the destroy log.

Resolution: Check the log for messages that read "destroy log is filled with undeletable files". If this happens, it indicates users are running programs that open files, unlink them, and continue to use the files for long periods of time without calling exit or closing the files. These files must remain in the destroy log until the user programs exit. This is a known problem. The only way to fix this problem is to increase the value of **DestroyLogSize**, or persuade users to change their programs.

Diagnosis 3: The system is busy destroying archived files.

Resolution: The HDM gives preference to destroying archived files. If the **DestroyLogSize** parameter is too big, HDM will have to spend a long time destroying archived files before it has a chance to destroy mirrored files. To fix this problem, use a smaller value for **DestroyLogSize**.

9.4.4.11 Cannot access files after a gateway is moved to a new host

Diagnosis 1: The DMAP Gateway host machine has been changed.

Resolution: To fix this problem, edit **gateways.dat** and **filesys.dat** to use the new host names, then restart HDM.

9.4.4.12 An HDM sub-process will not run because shared memory is corrupt

Diagnosis 1: Some other program is using the HDM shared memory.

Resolution: Use the **ipcs** command to determine if another process is using the same shared memory. If so, use **ipcs** to find a key that is not already in use and change **SharedMemoryKey** to that value.

9.4.4.13 End user reports numerous I/O (EIO) error messages

Diagnosis 1: An internal error has occurred in HDM.

Resolution: In most cases, an **EIO** error indicates an internal error has occurred in HDM. This may indicate that a configuration file is set up incorrectly, or that there as a programming error in HDM. Examine the message log closely to determine the cause of the problem. If necessary, increase the level of message logging. This can be done temporarily with **hdm_admin report**, and permanently by changing the **LogRecordMask** parameter in the configuration file.

Diagnosis 2: A DFS SMT routine failed in an unexpected way.

Resolution: Check the message log to determine the cause of the problem. If the problem continues, call software support.

Diagnosis 3: An I/O error has occurred.

Resolution: This would be rare, but to be thorough, check the message log.

9.4.4.14 End user reports numerous EAGAIN or EBUSY messages

Diagnosis 1: The system is overloaded.

Resolution: If the user complains about too many “resource busy” or “resource temporarily unavailable” messages, it could indicate that the HDM and/or DFS systems are overloaded. It may be possible to fix this problem by increasing the **NumNamespProcesses** and/or **NumDatProcesses** parameters in the configuration file -- assuming there are currently not too many processes. Ultimately, it may be necessary to spread the load over more processors.

Diagnosis 2: The XDSM kernel extension is configured incorrectly.

Resolution: Make sure the **delay** parameter on the **cfgdmepi** command line is set to 1. If a large value is used, the kernel will delay some operations so long the cache manager gives up. The following command line should appear in **rc.dfs**:

/usr/sbin/cfgdmepi -delay 1 -a /usr/lib/drivers/dmlfs.ext

9.4.4.15 Cannot create a fileset because of lack of space

Diagnosis 1: The HDM needs to be reconfigured.

Resolution: If the SSM Fileset Create screen returns an **ENOSPC** error, it may mean the HDM fileset table is not big enough. Edit **filesys.dat** to increase **MaxFilesets**.

9.4.4.16 The end user cannot use a new fileset

Diagnosis 1: The fileset has not been completely configured.

Resolution: Immediately after creating a fileset with **fts create**, the fileset is unavailable to the user. Use the SSM fileset create window to finish creating the fileset. Check both the HDM and HPSS logs to verify the fileset was created correctly. Check **filesys.dat** to determine if the fileset entry is complete.

Diagnosis 2: The fileset has been configured incorrectly.

Resolution: Check **filesys.dat** to ensure that the global and local mount points for the fileset are correct. In particular, make sure that a mount point for one fileset has not inadvertently been used for another fileset.

Diagnosis 3: The HPSS fileset was created while HDM was down.

Resolution: Check **filesys.dat** to ensure that the fileset has been properly configured. If not, and, if the fileset has already been created on HPSS, it will be necessary to delete the HPSS portion of the fileset first. Before creating the fileset again, make sure HDM is running.

9.4.4.17 The operating system crashes

Diagnosis 1: There is a problem with the XDSM kernel extensions.

Resolution: Try to get a system dump to send to software support. If the problem occurs often, use the **dfstrace** diagnostic tool to get occasional dumps in anticipation of a problem. (Unfortunately **dfstrace** cannot be used after a crash.) To get the most useful set of diagnostics, issue this command once after the system restarts:

```
dfstrace setset -set fx fshost dmops dmab xops episode/anode \ episode/vnopsVerbose
episode/vnopsBasic krpc cm zlc \ -active
```

and then issue this command whenever a trace dump is wanted:

```
ffstrace dump -file some.file.name
```

9.4.4.18 The system performs poorly

Diagnosis 1: The XSDM kernel extension is configured incorrectly.

Resolution 1: Make sure the **delay** parameter on the **cfgdmepi** command line is set to 1. If a large value is used, the kernel will delay some operations so long that the cache manager gives up. The following command line should appear in **rc.dfs**:

/usr/sbin/cfgdmepi -delay 1 -a /usr/lib/drivers/dmlfs.ext

Diagnosis 2: The number of HDM event handlers is too small.

Resolution: **NumNamespProcesses** and / or **NumDataProcesses** could be increased so more operations can be performed in parallel. However, if too many processes are created, the performance of the system will deteriorate.

Diagnosis 3: The size of a data structures needs to be changed.

Resolution: The **EventQueueSize** configuration parameter may need to be increased so busy aggregates do not bottleneck relatively inactive ones. The **DestroyLogSize** may need to be increased so destroy process does not bottleneck destroy operations.

Diagnosis 4: Performance on a mirrored fileset is slow.

Resolution: There is more overhead using mirrored filesets. If it is not essential to have the name space mirrored in HPSS, consider using the backup option for the fileset.

9.4.4.19 Files cannot be migrated and purged

Diagnosis 1: The Episode file system is full.

Resolution: In some cases, the Episode file system may become so full, it is impossible to save the information needed to migrate files. This is, of course, a difficult problem to solve. If possible, get the users to delete some files, then migrate and purge the file system. As a long range solution, make sure he migrate and purge processes are run often enough to keep ahead of space shortages.

9.4.4.20 Cannot create filesets because file system is unknown

Diagnosis 1: The HDM has not been told to manage a file system.

Resolution: The standard procedure for creating a file system calls for running **create_fsys**. Possibly, this step was overlooked. To determine if the file system is known to HDM, check **filesys.dat**. If ever **create_fsys** is not used to create a file system, edit **filesys.dat** to include the new file system, then restart the HDM.

9.4.4.21 EINVAL error returned when creating a fileset

Diagnosis 1: The file system was identified incorrectly.

Resolution: The SSM Create Fileset screen calls for the administrator to enter the file system name. If the fileset name is used instead, an **EINVAL** error will result. A typical file system name is **/dev/tardis_aggr1**.

9.4.4.22 The log has messages with error numbers -50 and/or -10000

Diagnosis 1: There is a problem with HPSS.

Resolution: An HPSS server, usually, the DMAP Gateway, is unable to communicate with other servers. Try restarting the DMAP Gateway and/or other HPSS servers.

Appendix A

Glossary of Terms and Acronyms

ACI	Automatic Media Library Client Interface
ACL	Access Control List
ACSLS	Automated Cartridge System Library Software (Science Technology Corporation)
ADIC	Advanced Digital Information Corporation
accounting	A log record message type used to log information to be used by the HPSS Accounting process. This message type is not currently used.
aggregate	A disk partition that has been modified to provide support for DFS filesets and access control lists.
AIX	Advanced Interactive Executive
alarm	A log record message type used to log high-level error conditions. The default logging policy is to log alarms and send alarms to the Storage System Management (SSM) to be displayed in the Alarm and Event window.
AML	Automated Media Library
AMS	Archive Management Unit
ANSI	American National Standards Institute
API	Application Program Interface
Archive	One or more interconnected storage systems of the same architecture.
archived fileset	A DFS fileset whose files are archived on HPSS but do not appear in the HPSS name space. Users can access these files from DFS but not from HPSS.
attribute	When referring to a managed object, an attribute is one discrete piece of information, or set of related information, within that object.
attribute change	When referring to a managed object, an attribute change is the modification of an object attribute. This event may result in a notification being sent to SSM, if SSM is currently registered for that attribute.

audit (security)	An operation that produces lists of HPSS log messages whose record type is SECURITY. A security audit is used to provide a trail of security-relevant activity in HPSS.
Bar code	An array of rectangular bars and spaces in a predetermined pattern (e.g., UPC symbol)
BFS	Bitfile Server
bitfile	A logical string of bits unrestricted in size or internal structure. HPSS imposes a size limitation in 8-bit bytes based upon the maximum size in bytes that can be represented by a 64-bit unsigned integer.
bitfile segment	An internal metadata structure, not normally visible, used by the Bitfile Server to map contiguous pieces of a bitfile to underlying storage provided by a Storage Server.
Bitfile Server	An HPSS server that provides a logical abstraction of bitfiles to its clients.
BMUX	Block Multiplexer Channel
bytes between tape marks	The number of data bytes that are written to a tape virtual volume before the Tape Storage Server requires a tape mark on the physical media.
CAP	Cartridge Access Port
cartridge	A physical media container, such as a tape reel or cassette, capable of being mounted on and dismounted from a drive. A fixed disk is technically considered to be a cartridge because it meets this definition and can be logically mounted and dismounted.
CDS	Cell Directory Service
central log	The main repository of logged messages from all HPSS servers enabled to send messages to the Log Daemon.
Class of Service	A set of storage system characteristics used to group bitfiles with similar logical characteristics and performance requirements together. A Class of Service is supported by an underlying hierarchy of storage classes.
configuration	The process of initializing or modifying various parameters affecting the behavior of an HPSS server or infrastructure service.
configuration file	An Encina Structured File Server (SFS) file that stores information defining HPSS server operating parameters, storage characteristics, policies, devices and drives, and other information.
COS	Class of Service
daemon	A UNIX program that runs continuously in the background, lying dormant until some condition is met.
Data Server	A Storage System Management (SSM) component that provides the bridge between the System Manager and the Graphical User Interface (GUI).
DCE	Distributed Computing Environment

debug	A log record message type used to log lower-level error conditions. The default logging policy is to log debug messages.
DEC	Digital Equipment Corporation.
delog	The process of extraction, formatting, and outputting HPSS central log records.
deregistration	The process of disabling notification to SSM for a particular attribute change.
descriptive name	A human-readable name for an HPSS server.
device	A physical piece of hardware, usually associated with a drive, that is capable of reading or writing data.
DFS	The Distributed File Service is a system that joins local files systems of File Server machines, making the file systems available globally.
DFS/HPSS fileset	A fileset that is represented in both DFS and HPSS.
directory	An HPSS object than can contain files, symbolic links, hard links, and other directories.
dismount	An operation in which a cartridge is either physically or logically removed from a device, rendering it unreadable and unwritable. In the case of tape cartridges, a dismount operation is a physical operation. In the case of a fixed disk unit, a dismount is a logical operation.
DMAP Gateway	A server that acts as a gateway between DFS and HPSS. The server relays requests between HPSS and the HPSS/DMAP server.
DMAPI	The Data Management APIs defined by the XDSM specification. These APIs allow a Data Management Application to monitor events on files, and provide special interfaces to manage the data in the files.
DMG	Shorthand for DMAP Gateway.
DMLFS	A DCE Local File System that has been modified to support XDSM Data Management APIs.
DNS	Domain Name Service
DOE	Department of Energy
drive	A physical piece of hardware capable of reading and/or writing mounted cartridges. The terms device and drive are often used interchangeably.
DTS	Distributed Time Service
Encina	A product from Transarc Corporation that serves as the HPSS transaction manager. The Encina Structured File Server (SFS) serves as the HPSS Metadata Manager.
ERA	Extended Registry Attribute
ESCON	Enterprise System Connection

event	A log record message type used to log informational messages (e.g., subsystem starting, subsystem terminating). The default logging policy is to log events and send events to SSM to be displayed in the Alarm and Event window.
export	An operation in which a cartridge and its associated storage space are removed from the HPSS system. An export translates into a removal of a cartridge's storage space from HPSS followed by an eject, which is the removal of the cartridge itself from its Physical Volume Repository.
FDDI	Fiber Distributed Data Interface.
file	An object than can be written to, read from, or both, with attributes including access permissions and type, as defined by POSIX (P1003.1-1990). HPSS supports only regular files.
file family	An attribute of an HPSS file that is used to group a set of files on a common set of tape virtual volumes.
file server	A machine that manages one or more DFS aggregates.
fileset	A collection of related files that are organized into a single easily managed unit. A fileset is a disjoint directory tree that can be mounted in some other directory tree to make it accessible to users.
fileset id	A 64-bit number that uniquely identifies a fileset.
fileset name	A name that uniquely identifies a fileset.
file system	Another term for a DFS aggregate.
file system id	A 32-bit number that uniquely identifies an aggregate.
FTP	File Transfer Protocol
GB	Gigabyte (2^{30})
GDA	Global Directory Agent
GDS	Global Directory Service
GECOS	The comment field in a UNIX password entry that can contain general information about a user, such as office or phone number.
GID	Group Identifier
global mount point	A path in the DFS name space where a fileset has been mounted. A global mount point typically starts with the characters '/:'.
GSS	Generic Security Service
GUI	Graphical User Interface
halt	A forced shutdown.
HDM	Shorthand for HPSS/DMAP.
hierarchy	See Storage Hierarchy.

HIMF	HPSS Interim Metadata Format
HiPPI	High Performance Parallel Interface
HPSS	High Performance Storage System
HPSS-only fileset	An HPSS fileset that has no counterpart in DFS.
HPSS/DMAP	A Data Management Application that monitors DFS activity in order to keep DFS and HPSS synchronized. The server relays requests between DFS and the DMAP Gateway.
IBM	International Business Machines Corporation
ID	Identifier
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
I/E	Import/Export
IETF	Internet Engineering Task Force
Imex	Import/Export
import	An operation in which a cartridge and its associated storage space are made available to the HPSS system. An import translates into an inject (the physical introduction of a cartridge to a Physical Volume Repository) followed by the addition of the cartridge's storage space to the HPSS system.
I/O	Input/Output
IOD/IOR	Input/Output Descriptor / Input/Output Reply
IP	Internet Protocol
IPI	Intelligent Peripheral Interface
IRIX	SGI's implementation of UNIX
IPI-3	Protocol for Intelligent Peripheral Interface
junction	A mount point for an HPSS fileset. The fileset may be in the either a local or remote HPSS system.
KB	Kilobyte (2^{10})
LAN	Local Area Network
LANL	Los Alamos National Laboratory
LARC	Langley Research Center

latency	For tape media, the average time in seconds between the start of a read or write request, and the time when the drive actually begins reading or writing the tape.
LCU	Library Control Unit
LFS	A DCE Local File System, which is a high performance log-based file system that supports the use of access control lists and multiple filesets within a single aggregate.
LLNL	Lawrence Livermore National Laboratory
LMCP	Library Manager Control Point
LMU	Library Management Unit
local log	An optional circular log maintained by a Log Client. The central log contains formatted messages from all enabled HPSS servers residing on the same node as the Log Client.
local mount point	The name of a directory in a Unix file system where a DFS fileset has been mounted.
location server	An HPSS server that is used to help clients locate the appropriate Bitfile Server, Name Server, and/or other HPSS server to use for a particular request.
Log Client	An HPSS server executing on each HPSS node that is responsible for sending log messages to the local log, to the Log Daemon for central logging, and to SSM to display messages in the Alarm and Event window.
Log Daemon	An HPSS server responsible for writing log messages to the central log.
log record	The records received and maintained in a central log by the HPSS Log Daemon.
log record type	An indicator of whether a message to be logged is an alarm, event, status, debug, request, security, or accounting record.
logging service	An HPSS infrastructure service consisting of a central Log Daemon, one or more Log Clients, and server-specific logging policies.
LRU	Least Recently Used
LS	Location Server
MAC	Mandatory Access Control
managed object	A programming data structure that represents an HPSS system resource. The resource can be monitored and controlled by operations on the managed object. Managed objects in HPSS are used to represent servers, drives, storage media, jobs, and other resources.
MB	Megabyte (2^{20})
metadata	Control information about the data stored under HPSS, such as location, access times, permissions, and storage policies.

Metadata Manager	The subsystem / component within HPSS responsible for the physical storage and management of HPSS metadata as well as the transactional mechanisms for manipulating HPSS meta data. The current Metadata Manager for HPSS is the Encina SFS product, together with a set of HPSS-developed application program interfaces (APIs) that provide a layer of abstraction on top of the Encina SFS data access methods.
Metadata Monitor	A type of HPSS server that is responsible for monitoring the space utilization of a single Encina SFS process. The Metadata Monitor calculates the overall amount of disk space being used by SFS and generates alarms to SSM as appropriate whenever various thresholds are exceeded.
migrate	To copy file data from a level in the file's hierarchy onto the next lower level in the hierarchy.
Migration/Purge Server	An HPSS server responsible for supervising the placement of data in the storage hierarchies based upon site-defined migration and purge policies.
mirrored fileset	A DFS fileset whose files are mirrored on HPSS. These files appear in both the DFS and HPSS name spaces, and so users can access the files from either place.
MM	Metadata Manager
MMON	Metadata Monitor
mount	An operation in which a cartridge is either physically or logically made readable and / or writable on a drive. In the case of tape cartridges, a mount operation is a physical operation. In the case of a fixed disk unit, a mount is a logical operation.
mount point	A place where a fileset is mounted in the DFS and / or HPSS name spaces.
Mount Daemon	An HPSS server object responsible for performing mount request operations for client systems accessing HPSS data through the HPSS Network File System (NFS) Daemon.
Mover	An HPSS server that provides control of storage devices and data transfers within HPSS.
MPS	Migration / Purge Server
MRA	Media Recovery Archive
MSSRM	Mass Storage System Reference Model
MVR	Mover
NASA	National Aeronautics and Space Administration
Name Server	An HPSS server that provides a mapping between names and machine-oriented identifiers. In addition, the Name Server performs access verification and provides the Portable Operating System Interface (POSIX).
name space	The set of name-object pairs managed by the HPSS Name Server.
NDCG	Non-DCE Client Gateway
NDAPI	Non-DCE Client Application Program Interface

NERSC	National Energy Research Supercomputer Center
Network File System	A protocol developed by Sun Microsystems that allows transparent access to files over a network.
NFS	Network File System
NFS Daemon	An HPSS server that provides access to HPSS name space objects and bitfile data for client systems through the Network File System (NFS) V2 protocol.
NLS	National Language Support
Non-DCE Client Gateway	An HPSS server that provides access to the user calls of the HPSS Client Application Program Interface for Non-DCE client applications.
Non-DCE Client Application Program Interface	An HPSS library that offers the user calls of the HPSS Client API for client applications running on platforms which do not support either DCE or Encina.
notification	A notice from one server to another about a noteworthy occurrence. HPSS notifications include notices sent from other servers to SSM of changes in managed object attributes, changes in tape mount information, and log messages that are alarm, event, and status log record message types.
NS	Name Server
NSL	National Storage Laboratory
object	See Managed Object.
ODM	Object Data Manager
OFD	Open File Descriptor
ONC	Online Network Computing
OSF	Open Software Foundation
OS/2	Operating System (multi-tasking, single user) used on the AMU controller PC
PB	Petabyte (2^{50})
PFTP	Parallel File Transfer Protocol
physical volume	An HPSS object managed jointly by the Storage Server and the Physical Volume Library that represents the portion of a cartridge that can be contiguously accessed when mounted. A single cartridge may contain multiple physical volumes.
Physical Volume Library	An HPSS server that manages mounts and dismounts of HPSS physical volumes.
Physical Volume Repository	An HPSS server that manages the robotic or human agent responsible for mounting and dismounting cartridges.

PIOFS	Parallel I/O File System
POSIX	Portable Operating System Interface (for computer environments)
purge	Deletion of file data from a level in the file's hierarchy after the data has been duplicated at lower levels in the hierarchy and is no longer needed at the deletion level.
purge lock	A lock applied to a bitfile which prohibits the bitfile from being purged.
PV	Physical Volume
PVL	Physical Volume Library
PVM	Physical Volume Manager
PVR	Physical Volume Repository
RAID	Redundant Array of Independent Disks
RAM	Random Access Memory
reclaim	The act of making virtual volumes that have no active data (i.e., empty) available for use in storing new data, so that data media can be reused.
registration	The process by which SSM requests notification of changes to specified attributes of a managed object.
reinitialization	An HPSS SSM administrative operation that directs an HPSS server to reread its latest configuration information, and to change its operating parameters to match that configuration, without going through a server shutdown and restart.
repack	The act of moving data from a virtual volume onto another virtual volume with the same characteristics (storage class) with the intention of freeing up all data references to that virtual volume. The act of emptying a virtual volume of all file data.
request	A log record message type used to log some action being performed by an HPSS server on behalf of a client. The default logging policy is to log request messages.
RISC	Reduced Instruction Set Computer/Cycles
RMS	Removable Media Service
RPC	Remote Procedure Call
Sammi	A commercial software product group from Kinesix Corporation that manages the graphical user interface to SSM in conjunction with the SSM Data Server.
SCSI	Small Computer Systems Interface
security	A log record message type used to log security related events (e.g., authorization failures). The default logging policy is to log security messages.
SFS	Structured File Server

SGI	Silicon Graphics
shelf tape	A cartridge which has been physically removed from a tape library, but whose file metadata still resides in HPSS.
shutdown	An HPSS SSM administrative operation that causes a server to stop its execution gracefully.
sink	The set of destinations to which data is sent during a data transfer (e.g., disk devices, memory buffers, network addresses).
SMIT	System Management Interface Tool
SNL	Sandia National Laboratories
SOID	Standard Object ID. An internal HPSS storage object identifier that uniquely identifies a storage resource.
source	The set of origins from which data is received during a data transfer (e.g., disk devices, memory buffers, network addresses).
SP	Scalable Processor
SS	Storage Server
SSA	Serial Storage Architecture
SSM	Storage System Management
SSM session	The environment in which an SSM user interacts with SSM to monitor and control HPSS through the SSM windows. SSM itself may be running without any sessions active. When an SSM user starts up Sammi and logs in, an SSM session begins and lasts until the user logs off. It is possible to have multiple sessions accessing the same SSM.
stage	To copy file data from a level in the file's hierarchy onto the top level in the hierarchy.
start-up	An HPSS SSM administrative operation that causes a server to begin execution.
status	A log record message type used to log processing results. This message type is being used to report status from the HPSS Accounting process. The default logging policy is to log the status messages and send them to SSM to be displayed in a pop-up window. Separate windows are displayed for each unique request. The message text within the pop-up window will update when additional messages are received for a particular request.
STK	Storage Technology Corporation
storage class	An HPSS object used to group storage media together to provide storage for HPSS data with specific characteristics. The characteristics are both physical and logical.
storage hierarchy	An ordered collection of storage classes. The hierarchy consists of a fixed number of storage levels numbered from level 1 to the number of levels in the hierarchy, with the maximum level being limited to 5 by HPSS. Each level is associated with a specific storage class. Migration and stage

	commands result in data being copied between different storage levels in the hierarchy. Each Class of Service has an associated hierarchy.
storage level	The relative position of a single storage class in a storage hierarchy. For example, if a storage class is at the top of a hierarchy, the storage level is 1.
storage map	An HPSS object managed by the Storage Server and used to keep track of allocated storage space.
storage segment	An HPSS object managed by the Storage Server and utilized by the Bitfile Server to provide storage for a bitfile or parts of a bitfile.
Storage Server	An HPSS object that provides control over a hierarchy of virtual and physical storage resources.
Storage System Management (SSM)	An HPSS component that provides monitoring and control of HPSS via a windowed operator interface. SSM has three components: (1) the System Manager, which communicates with all other HPSS components requiring monitoring or control, (2) the Data Server, which provides the bridge between the System Manager and the GUI, and (3) the GUI itself, which includes the Sammi Runtime Environment and the set of SSM windows.
stripe length	The number of bytes that must be written to span all the physical storage media (physical volumes) that are grouped together to form the logical storage media (virtual volume). The stripe length equals the virtual volume block size multiplied by the number of physical volumes in the stripe group (i.e., stripe width).
stripe width	The number of physical volumes grouped together to represent a virtual volume.
System Manager	A Storage System Management (SSM) server that communicates with all other HPSS components requiring monitoring or control.
TB	Terabyte (2^{40})
TCP/IP	Transmission Control Protocol / Internet Protocol
trace	A log record message type used to record entry / exit processing paths through HPSS server software. The default logging policy is to *not* log trace message types.
transaction	A programming construct that enables multiple data operations to possess the following properties: • All operations commit or abort / roll-back together such that they form a single, atomic unit of work. • All data modified as part of the same transaction are guaranteed to maintain a consistent state whether the transaction is aborted or committed. • Data modified from one transaction are isolated from other transactions until the transaction is either committed or aborted. • Once the transaction commits, all changes to data are guaranteed to be permanent.
TTY	Teletypewriter
UDP	User Datagram Protocol

UID	User Identifier
UUID	Universal Unique Identifier
virtual volume	An HPSS object managed by the Storage Server that is used to represent logical media. A virtual volume is made up of a group of physical storage media (a stripe group of physical volumes).
virtual volume block size	The size of the block of data bytes that is written to each physical volume of a striped virtual volume before switching to the next physical volume.
VV	Virtual Volume
XCT	Cross Cell Trust
XDSM	The Open Group's Data Storage Management standard. It defines APIs that use events to notify Data Management applications about operations on files.

Appendix B

References

1. *HPSS Error Messages Reference Manual*, February 2000, Release 4.1.1.
2. *HPSS Programmer's Reference Guide*, Volume 1, February 2000, Release 4.1.1.
3. *HPSS Programmer's Reference Guide*, Volume 2, February 2000, Release 4.1.1.
4. *HPSS User's Guide*, February 2000, Release 4.1.1.
5. *IBM AIX Version 4.3 Installation Guide*, SC23-4112-01
6. *AIX Versions 3.2 and 4.1 Performance Tuning Guide*, SC23-2365-04
7. *DCE for AIX, Version 2.2: Quick Beginnings*, SC 23-4188-00
8. *IBM DCE for AIX, Version 2.2: Introduction to DCE*
9. *IBM DCE for AIX, Version 2.2: Administration Guide -- Introduction*
10. *IBM DCE for AIX, Version 2.2: Administration Guide -- Core Components*
11. *IBM DCE for AIX, Version 2.2: Administration Commands Reference*
12. *IBM DCE for AIX, Version 2.2: DFS Administration Guide and Reference*
13. *Data Storage Management (XDSM)* API, ISBN 1-85912-190-X
14. *Installing, Managing, and Using the IBM AIX Parallel I/O File System*, SH34-6065-02
15. *Sammi Runtime Reference* (for Version 4.0)
16. *Sammi User's Guide* (for Version 3.0)
17. *Sammi System Administrator's Guide* (for Version 3.0)
18. *STK Automated Cartridge System Library Software (ACSLS) System Administrator's Guide*, PN 16716

19. *STK Automated Cartridge System Library Software Programmer's Guide*, PN 16718

20. *IBM 3494 Tape Library Dataserver Operator's Guide*, GA32-0280-02

21. *Parallel and ESCON Channel Tape Attachment/6000 Installation and User's Guide*, GA32-0311-02

22. *IBM 3495 Operator's Guide*, GA32-0235-02

23. *Parallel and ESCON Channel Tape Attachment/6000 Installation and User's Guide*, GA32-0311-02

24. *IBM SCSI Device Drivers: Installation and User's Guide*, GC35-0154-01

25. *Encina Administration Guide Volume 1: Introduction and Configuration*

26. *Encina Administration Guide Volume 2: Basic Administration*

27. *Encina Administration Guide Volume 3: Server Administration*

28. *Encina Administration Guide Volume 4: Advanced Administration*

29. *Ampex DST 800 Automated Cartridge Library—IPI system Administrator's Guide* [This document is cited in the online man pages for the Ampex DST, but may not be available.]

30. *OSF/DCE Application Development Reference Manual*, SR28-4995-00

31. J. Steiner, C. Neuman, and J. Schiller, *"Kerberos: An Authentication Service for Open Network Systems,"* USENIX 1988 Winter Conference Proceedings (1988).

32. R.W. Watson and R.A. Coyne, "*The Parallel I/O Architecture of the High-Performance Storage System (HPSS),*" from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.

33. T.W. Tyler and D.W. Fisher, "*Using Distributed OLTP Technology in a High-Performance Storage System*," from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.

34. J.K. Deutsch and M.R. Gary, "*Physical Volume Library Deadlock Avoidance in a Striped Media Environment*," from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.

35. R. Grossman, X. Qin, W. Xu, H. Hulen, and T. Tyler, "*An Architecture for a Scalable, High-Performance Digital Library,*" from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.

36. S. Louis and R.D. Burris, "*Management Issues for High-Performance Storage Systems,*" from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.

37. D. Fisher, J. Sobolewski, and T. Tyler, "*Distributed Metadata Management in the High Performance Storage System*," from the 1st IEEE Metadata Conference, April 16-18, 1996.

Appendix C

HPSS Worksheets

This appendix provides worksheets to allow the administrator to record the essential planning decisions for HPSS. It is highly recommended that the administrator fills out these worksheets during the planning process described in Chapter 2 before proceeding with the HPSS Installation, the HPSS Infrastructure Configuration and the HPSS Configuration procedures.

The worksheets are provided for the following planning tasks:

- HPSS Architecture Planning Worksheet (C.1)
- HPSS Installation Worksheet (C.2)
- HPSS Infrastructure Configuration Worksheets (C.3)

C.1 HPSS Architecture Planning Worksheet

To identify the HPSS architecture and resources, complete the following worksheet. This worksheet should be used as part of the HPSS configuration planning process defined in Section 2.1.2.

Installation Node Name: _____

Remote Node Names: _____

HPSS Tape Drives

Drive Name	Drive Type	Node/ Network	Cartridge Type/Size	# Drives	# of Cartridges

HPSS Disk Drives

Drive Name	Drive Type	Node/Network	Volume Name	Volume Size

C.1 HPSS Architecture Planning Worksheet (Cont.)

HPSS Servers

Server	# of Servers	Node Name(s)
NS	1	
BFS	1	
Disk SS		
Tape SS		
LS		
DMAP Gateway		
MPS		
PVL	1	
AML PVR		
STK PVR		
3494 PVR		
3495 PVR		
Operator PVR		
Mover		
Log Daemon	1	
Log Client		
Metadata Monitor		
NFS Daemon		
Mount Daemon		
Startup Daemon		
SSM SM	1	
SSM DS		
NDCG		

C.1 HPSS Architecture Planning Worksheet (Cont.)

HPSS User Interfaces

User Interface	# of Nodes	Node Name(s)
FTP / PFTP		
NFS		
DFS		
NDAPI		

C.2 HPSS Installation Worksheet

To plan for the HPSS installation, complete the following worksheet and use the information recorded in the worksheet during the installation process described in Section 3.3.

Installation Node Name: _____

INPUT Device / Directory for Software:

__ 8 mm; device name_____

__file; directory / file_name:_____

C.3 HPSS Infrastructure Configuration Worksheets

To plan for the HPSS infrastructure configuration, complete the Installation Node Infrastructure Configuration worksheet (Section C.3.1) for the installation node. If the HPSS system involves multiple nodes, complete the Remote Node Infrastructure Configuration worksheet (Section C.3.2) for each remote node. Use the information recorded in these worksheets during the infrastructure configuration process described in Section 4.4.

C.3.1 Installation Node Infrastructure Configuration Worksheet

Installation Node Name: _____

HPSS Servers To Be Executed On This Node:

__ NS	__ BFS	__ Disk SS	__ Tape SS
__ MPS	__ PVL	__ PVR	__ MVR
__ LOGD	__ LOGC	__ MMON	__ NFSD
__ LS	__ FTP	__ NDCG	__ DMG
__ HDM (AIX4.3)			

Configure HPSS With DCE: Yes

Configure HPSS With Encina: Yes / No

Setup FTP Daemon: Yes / No

Setup Startup Daemon: Yes

Install HPSS Subsystem(s) To Remote Node(s):

_____

_____

_____

_____

Add SSM Administrative User: Yes / No

Start SSM Administrative Session: Yes / No

C.3.2 Remote Node Infrastructure Configuration Worksheet

Remote Node Name: _____

HPSS Servers or Subsystem To Be Executed On This Node:

__ NS	__ BFS	__ Disk SS	__ Tape SS
__ MPS	__ PVL	__ PVR	__ MVR
__ LOGD	__ LOGC	__ MMON	__ NFSD
__ LS	__ FTP	__ NDCG	__ DMG
__ HDM (AIX4.3)			

Configure HPSS With DCE: Yes

Configure HPSS With Encina: Yes / No

Setup FTP Daemon: Yes / No

Setup Startup Daemon: Yes

Add SSM Administrative User: Yes / No

Start SSM Administrative Session: Yes / No

Exclusive HPSS Servers To Be Executed On This Node:

Note: If any one of the following servers is installed on a remote node, **no other server may be installed** *on the same remote node.*

__ HDM (AIX4.2)

-or-

__ SUN

-or-

__ NDCAIX

-or-

__ NDCSUN

-or-

__ NDCSGI

Appendix D

Accounting Examples

D.1 Introduction

This appendix describes how to set up the gathering of accounting data at a customer site. The accounting data is used by the customer to calculate charges for the use of HPSS resources. The accounting data represents a blurred snapshot of the system storage usage as it existed during the accounting run.

HPSS provides the capability of keeping duplicate files in some storage hierarchies. In the accumulation of accounting data, the storage for each duplicate file is reported; consequently, the user can expect to be charged for all duplicate copies.

D.2 Site Accounting Requirements

What are the accounting requirements for your site? The accounting department at each site should be consulted to determine what kind of information it will need attached to each accounting record. Based on this, an HPSS Account Index for a user account can be set up in an Account Map to point to the required information.

Do you need information per user? Per account code? Per group? Some sites will need accounting totaled on a per user, per Account Index basis. This type of accounting is called Site-style accounting. There are many sites that will be required to implement charging in this way.

What kind of reports will be generated? Many UNIX resources report usage by user ID (UID). Some sites will need to use the UNIX-style accounting, which gives accounting totaled on a per user basis only. The accounting department will provide a program that takes accounting information on a per user (UID) basis and applies percentages to split out charges for multiple project accounts per user.

D.3 Processing of HPSS Accounting Data

How should the HPSS accounting data be processed? Should the data be written to a flat file or put into a site data base?

The default accounting output routine for HPSS generates a text file that contains two types of lines. The first type, denoted by a zero (0) in the third column, gives the following summary information about the storage used by a particular HPSS Account Index (AcctId) in a particular Class Of Service (COS):

- The total amount of data stored (Length) under the Account Index in the Class Of Service.
- The total number of files (#Files) stored under the Account Index in the Class Of Service.
- The total number of file accesses (#Accesses) to files owned by the Account Index in the Class Of Service. Note that file accesses are counted against the owner of the account accessing the file, not the owner of the file itself.

The second type of line has a non-zero value in the Storage Class column (SClass). This type of line contains information about the storage used by a particular HPSS Account Index (AcctId) in a particular Storage Class (SClass) within a particular Class Of Service (COS). These line contain the following information:

- The total number of file accesses (#Accesses) in this particular Storage Class for the given Class Of Service. If a class of service is configured to stage bitfiles on open, then all file accesses will occur in the storage class at the top of the hierarchy. If a class of service is configure to not stage bitfiles, then file accesses will occur in storage classes which are not at the top of the hierarchy.
- The total amount of data transferred (Transferred) into or out of this Storage Class and Class Of Service for this particular Account Index. Note that data transferred is counted against the owner of account transferring the data, not the owner of the data itself.

Example Accounting Report File:

```
# Comment file first line
# Comment file last line
# HPSS Accounting Snapshot completed on Wed Jul 15 12:57:00 1998
# Storage Unit Size : 1
# Total Number of Rows : 5
# Total Number of Accounts : 2
# Total Storage Units for HPSS system : 15598533
#
# Entries with '0' in the SClass field are COS totals.
# Other entries apply to individual storage classes.
#
# AcctId  COS     0   #Accesses       #Files       Length  (COS)
# AcctId  COS  SClass  #Accesses     Transferred            (SClass)
# ---   -----  -----  ----------    ----------    ----------
  203     3      0        0           5          212895
  634     1      0       89          89          4168147
  634     1      1       89        4168147
  634     5      0      152          152         11217491
  634     5      9      152        11217491
```

The HPSS accounting file will be correlated with the Account Map to determine the appropriate accounting charges. This is a customer site function.

Sites may wish to write a module that will redirect the accounting data into a local accounting data base. This module would replace the default HPSS module, **acct_WriteReport()**, which writes out the HPSS accounting data to a flat text file.

Where should the accounting data be stored? The HPSS accounting file and a copy of the current Account Map should be named with the date and time and stored for future reference. Individual sites should write scripts to copy / archive the generated accounting report file from it's original location.

D.4 Site Accounting Table

What should be in the Account Map corresponding to the HPSS Account Index number? The HPSS Account Index will correspond to an Account Map entry that has the site information. The following are examples of possible Account Maps:

Site-style Account Map:

Acct	User	UID	Charge	Update_flag
12	dlk	3152	5A12x401	0
27	dlk	3152	5A12x501	0
341	dlk	3152	5A12x601	0
469	dpc	1478	7A14x401	0
470	dpc	1478	7A14x501	0
471	dmb	5674	5A12x401	0
7111	dmb	5674	7A14x501	0
...	...	...	...	...

UNIX-style Account Map

UID (Account Index)
1478
3152
5674
...

D.5 Account Apportionment Table

In UNIX-style accounting, the UID (as Account Index) maps only to a specific user. The mapping of a UID to various percentages of different project charge codes can be done by the site accounting department in an Account Apportionment Table as shown in the following example:

UID	% of (Project(s))	
1478	75(DND) 25(CBC)	
3152	45(DDI) 25(DND)	30(CBC)
5674	100(DDI)	
...	.....	

Note: The Account Apportionment Table and Account Maps can be created by the individual sites. They are not created or maintained by HPSS. Some sites may wish to add more information, such as department and text name, or include less information, such as only the UID.

D.6 Maintaining and/or Modifying the Account Map

What are the policies for the Account Map? How will the Account Map be maintained and modified? Each site must develop tools to maintain and modify its Account Map according to local accounting policies. Tools will be necessary to change, add, or delete an HPSS Account Index and its associated information. The site must implement their own policies on what to do when an account is deleted, a user moves to another project, or a user leaves the system.

UNIX-style accounting changes of this nature are handled through the normal utilities that set up and modify users and UIDs. A basic set of site-developed utilities are described in more detail below.

- *Add a user and account*. New entries can be made and the next available HPSS Account Index number will be assigned from the free-list. The free-list will most likely consist of the last assigned number plus one, but could include reclaimed index numbers if a site chooses to re-use Account Index numbers that were previously assigned and no longer referenced. It is not likely that a site will need to reclaim Account Index numbers, but it is an option.

- *Delete a user*. When a user is deleted from the Account Map, the HPSS files must be reassigned to another HPSS Account Index. This should be done from the HPSS client interface side. The **update_flag** should be set to **true** to indicate that this account index number can be reclaimed. The reclaiming tool should check for files using the account number before reclaiming it. When the Account Index is reclaimed, it can be put on the free-list to be re-used. It is important to keep a copy of the Account Map that corresponds to the HPSS accounting snapshot of storage space in use for that time period so that the proper site information can be matched.

- *Delete account*. When an account is deleted, it is handled in the same manner as when a user is deleted.

- *Modify account*. The entries in the Account Map can be modified to correlate the Account Index to different information, but care should be taken to keep a copy of the corresponding tables for past HPSS accounting runs.

D.7 Accounting Reports

What kind of reports will be needed for your site? Learning what kind of accounting reports your site will need to generate will help you determine how detailed the collected accounting information should be. A typical Account Map will allow reports to be generated for the following:

- Total file accesses, amount of data transferred, and total space used per account, per class of service, per storage class.

- Total file accesses, amount of data transferred, and total space used per user, per class of service, per storage class.

D.8 Accounting Intervals and Charges

How often should HPSS accounting be run? How much should be charged for each unit of data transferred and each unit of space used? The time between accounting runs and the charging policy for space usage should be developed after consulting with the site accounting department. The following are some guidelines to consider:

- Accounting should be run at a regular intervals, such as once per month.
- An accounting run may take several minutes, and the storage system will probably be active during the run. The resource usage reported for each user will reflect the resources used by that user at the point when the accounting run encounters that user. This is why accounting represents a blurred snapshot instead of a snapshot at a single point in time.
- Certain accounting information is kept in cache for several minutes after it has changed. For this reason, changes to a user's accounting data may not appear in an accounting report until this several-minute period has elapsed. Those changes which are still in cache when accounting runs will not appear on the current accounting report, but will appear on the next accounting report.
- The number of file accesses and the amount of data transferred can be taken to represent the activity level of a certain user account in the HPSS system. This activity consumes network and server resources which it may be desirable to charge for in addition to the amount of storage space used by this account.
- It may be useful to charge different rates for user data depending on the level in the hierarchy on which it resides. For example, a disk storage class at the top of a hierarchy may be more expensive to maintain than tape classes toward the bottom of a hierarchy. This may justify charging more for the data resident on disk than for data on tape.

.

Appendix E

Installation Examples

E.1 HPSS Installation Worksheet For AIX

Installation Node Name: **Install-node**_____

INPUT Device / Directory for Software:

X 8 mm; device name _____

__file; directory / file_name:_____

E.2 HPSS Installation Screen Display

1. Enter **smitty install** at command line:

```
% smitty install
```

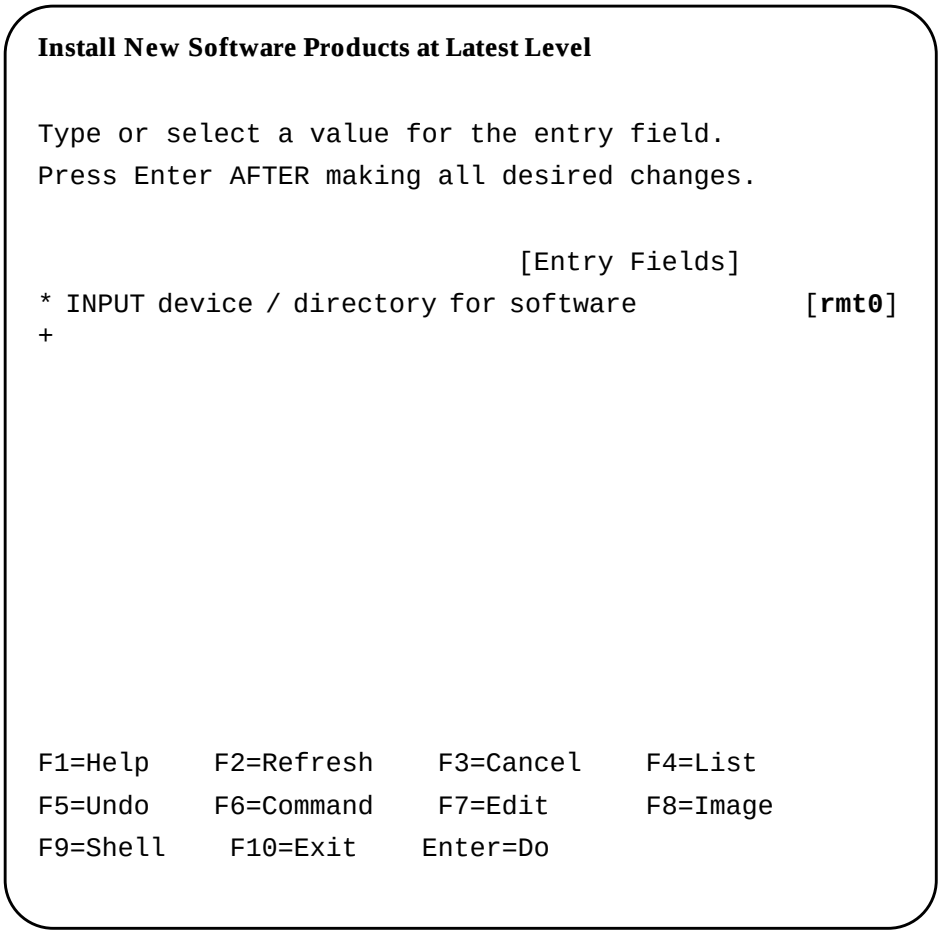
2. Select **Install and Update Software**

```
Software Installation & Maintenance

Move cursor to desired item and press Enter.

  Install and Update Software
  List Software and Related Information
  Software Maintenance and Utilities
  System Backup Manager









F1=Help    F2=Refresh    F3=Cancel    F8=Image
F9=Shell    F10=Exit    Enter=Do
```

3. Select **Install and Update Selectable Software (Custom Install)**

```
Install and Update Software

Move cursor to desired item and press Enter.

  Install Bundles of Software (Easy Install)
  Install and Update from LATEST Available Software
  Update Installed Software to Latest Level (Update All)
  Install and Update Software by Package Name (includes
devices and printers)
  Install Software Bundle (Easy Install)
  Update Software by Fix (APAR)
  Install and Update from ALL Available Software









F1=Help    F2=Refresh    F3=Cancel    F8=Image
F9=Shell    F10=Exit    Enter=Do
```

4. Enter **rmt0** as the installation media and press **Enter**.



5. Set values for entry fields.

Install and Update from LATEST Available Software

```
Type or select values in entry fields.
Press Enter AFTER making all desired changes.
                                          [Entry Fields]
* INPUT device / directory for software        rmt0
* SOFTWARE to install                  [all_licenced]  +
  PREVIEW only? (install operation will NOT occur)    no
  COMMIT software updates?                            no
  SAVE replaced files?                               yes
  AUTOMATICALLY install requisite software?           no
  EXTEND file systems if space needed?               yes
  OVERWRITE same or newer versions?                  yes
  VERIFY install and check file sizes?               yes
  Include corresponding LANGUAGE filesets?           yes
  DETAILED output?                                    no

F1=Help    F2=Refresh    F3=Cancel    F4=List
F5=Undo    F6=Command    F7=Edit      F8=Image
F9=Shell    F10=Exit    Enter=Do
```

6. Press **Enter** to continue.

```
Install and Update from LATEST Available Softwar
Type or select a value for the entry field.
Press Enter AFTER making all desired changes.

                                        [Entry Fields]
* INPUT device / directory for software        rmt0
* SOFTWARE to install                         [all_licensed]  +
  PREVIEW only? (install operation will NOT occur)    no
  COMMIT software updates?                           no
  SAVE replaced files?                               yes
  AUTOMATICALLY install requisite software?          no
  EXTEND file systems if space needed?               yes
  OVERWRITE same or newer versions?                  yes
  VERIFY install and check file sizes?               yes
  Include corresponding LANGUAGE filesets?           yes
  DETAILED output?                                   no

               |-----------------------------------------------------
               |          ARE YOU SURE?                              |
               |                                                     |
               |                                                     |
               | Continuing may delete information you may want      |
| to keep. This is your last chance to stop |
| before continuing. |
               |         Press Enter to continue.                    |
               |         Press Cancel to return to the application.  |
|              |
F1=Help    F2=Refresh    F3=Cancel    F4=List
```

7. Installation is completed and status is displayed.

```
                        COMMAND STATUS

Command: OK            stdout: yes           stderr: no

Before command completion, additional instructions may appear
below.

[TOP]

installp:  Performing requisite checking.
        (This may take several minutes.)

installp:  The following software products will be installed:
        hpss.ss at level 4.1.1.0
        hpss.pvr at level 4.1.1.0
        hpss.pvl at level 4.1.1.0
        hpss.log at level 4.1.1.0
        hpss.bfs at level 4.1.1.0
        hpss.inst at level 4.1.1.0
        hpss.cns at level 4.1.1.0
        hpss.ssm at level 4.1.1.0
        hpss.ftpd at level 4.1.1.0
[MORE...55]

F1=Help    F2=Refresh    F3=Cancel    F6=Command
F8=Image    F9=Shell    F10=Exit
```

.

.

Appendix F

Infrastructure Configuration Examples

F.1 Installation Node Infrastructure Configuration Worksheet

Installation Node Name: **mercury**

HPSS Servers To Be Executed On This Node:

X_ NS	X_ BFS	X_ Disk SS	X_ Tape SS
X_ MPS	X_ PVL	X_ PVR	X_ MVR
X_ LOGD	X_ LOGC	X_ MMON	X_ NFSD
X_ LS	X_ FTP	X_ NDCG	X_ DMG
X_ HJDM (AIX4.3			

Configure HPSS With DCE: Yes

Configure HPSS With Encina: Yes

Setup FTP Daemon: Yes

Setup Startup Daemon: Yes

Install HPSS Subsystem(s) To Remote Node(s):

sp2n06

Add SSM Administrative User: Yes

Start SSM Administrative Session: Yes

F.2 Remote Node Infrastructure Configuration Worksheet

Remote Node Name: **____sp2n06___**

HPSS Servers To Be Executed On This Node:

__ NS	__ BFS	__ Disk SS	__ Tape SS
__ MPS	__ PVL	__ PVR	__ MVR
__ LOGD	__ LOGC	__ MMON	__ NFSD
__ LS	__ FTP	__ NDCG	__ DMG

Configure HPSS With Encina: No

Setup FTP Daemon: No

Setup Startup Daemon: No

Add SSM Administrative User: No

Start SSM Administrative Session: No

Exclusive HPSS Subsystem To Be Installed On This Node: Yes

Note: If any one of the following subsystems is installed on a remote node, **no other subsystem or server may be installed** *on the same remote node.*

X_ HDM

-or-

__ NDC_AIX

-or-

__ NDC_SGI

F.3 Installation Node Infrastructure Configuration Screen Display

[root@mercury:/usr/lpp/hpss/config] mkhpss

<mkhpss> Running /usr/lpp/hpss/config/mkhpss!

```
                ====================================

<mkhpss>            Verify User ID
                ==============

<mkhpss> Status ==> User root; verified; continue...

<mkhpss>            Check HPSS Current Infrastructure Configuration
                ================================================

<mkhpss>            Perform HPSS Infrastructure Configuration:
                ========================================


<mkhpss> Status ==>   Platform: AIX
<mkhpss> Status ==>  Host Name: mercury
<mkhpss> Status ==> Start Time: Sun Sep 10 12:42:54 CDT 2000
<mkhpss> Status ==>  User Name: root


<mkhpss>            Create/Append /usr/lpp/hpss/config/hpss_unmake Shell Script
                ======================================================


<mkhpss>            Set up HPSS Special Directories and Links
                ========================================


<mkhpss>            Infrastructure Configuration Menu
                ================================

<mkhpss> Prompt ==>    Select Infrastructure Configuration Option:
<mkhpss>            [1] Configure HPSS with DCE
<mkhpss>            [2] Manage SFS Files
<mkhpss>            [3] Set Up FTP Daemon
<mkhpss>            [4] Set Up Startup Daemon
<mkhpss>            [5] Install HPSS Subsystem On Remote Node
<mkhpss>            [6] Add SSM Administrative User
<mkhpss>            [7] Start SSM Servers/User Session

<mkhpss>            [E] Re-run hpss_env()
<mkhpss>            [R] Re-configure HPSS
<mkhpss>            [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, R, X]):1
```

```
<mkhpss>          Perform /usr/lpp/hpss/config/hpss_dce_register
            ============================================


<mkhpss>          Verify DCE is Running
            ====================

<mkhpss> Status ==> DCE is running, continue...
<hpss_dce_register> Status => Running /usr/lpp/hpss/config/hpss_dce_register on mercury by
root on Sun Sep 10 12:43:47 CDT 2000

<hpss_dce_register>       Perform dce_login as cell_admin
                  ================================

<hpss_dce_register> Prompt => Perform DCE login as cell_admin; Please enter cell_admin
password
<hpss_dce_register> Reply => (cell_admin password:)
<hpss_dce_register> Status => DCE login successful...

<hpss_dce_register>       Create HPSS Server CDS directories
                  ==================================
Current site is: registry server at /.../mercury.ibm.com/subsys/dce/sec/master
bye.
<hpss_dce_register> Status => /usr/lpp/hpss/config/hpss_dce_register processing completed!
<mkhpss> Status ==> Configure HPSS with DCE completed, continue...


<mkhpss>          Infrastructure Configuration Menu
            ================================

<mkhpss> Prompt ==>   Select Infrastructure Configuration Option:
<mkhpss>          [1] Configure HPSS with DCE
<mkhpss>          [2] Manage SFS Files
<mkhpss>          [3] Set Up FTP Daemon
<mkhpss>          [4] Set Up Startup Daemon
<mkhpss>          [5] Install HPSS Subsystem On Remote Node
<mkhpss>          [6] Add SSM Administrative User
<mkhpss>          [7] Start SSM Servers/User Session

<mkhpss>          [E] Re-run hpss_env()
<mkhpss>          [R] Re-configure HPSS
<mkhpss>          [X] Exit
```

<mkhpss> Reply ===> (Select Option [1-7, E, R, X]):2

Enter SFS Server Name [/.:/encina/sfs/hpss]:

Available Operations:

1 - Create file
2 - Empty file
3 - Destroy file
4 - Query file
5 - List all files
6 - Change SFS server (currently "/.:/encina/sfs/hpss")
7 - Change SFS volume (currently "sfsVolhpss")
8 - Change SFS filename extension (currently <none>)

Select operation (<RETURN> to exit)> 1

Available Metadata Objects

1 - General Server Configurations
2 - Accounting Policies
3 - Account Log Records
4 - Account Summary Records
5 - Account Snapshot Records
6 - Migration Records
7 - Purge Records
8 - BFS Configurations
9 - Bitfiles
10 - Bitfile COS Changes
11 - Bitfile Tape Segments
12 - Bitfile Disk Segments
13 - Bitfile Disk Allocation Maps
14 - BFS Storage Segment Checkpoint
15 - BFS Storage Segment Unlinks
16 - Classes of Service
17 - DMAP Gateway Configurations
18 - DMAP Gateway File Sets
19 - File Families
20 - Hierarchies
21 - Non-DCE Gateway Configurations
22 - NS Configurations
23 - NS ACL Extensions
24 - NS Fileset Attrs
25 - NS Global Filesets
33 - Migration/Purge Configurations
34 - Migration Policies
35 - Purge Policies
36 - Migration/Purge Checkpoints
37 - Mount Daemon Configuration
38 - Mover Configurations
39 - Mover Devices
40 - NFS V2 Daemon Configuration
41 - PVL Configurations
42 - PVL Activities
43 - PVL Drives
44 - PVL Jobs
45 - PVL Physical Volumes
46 - PVR Configurations
47 - 3494 Cartridges
48 - 3495 Cartridges
49 - AML Cartridges
50 - Operator Cartridges
51 - STK Cartridges
52 - Remote Site Configurations
53 - Storage Server Configurations
54 - Storage Classes
55 - SS Disk Storage Maps
56 - SS Tape Storage Maps
57 - SS Disk Storage Segments

```
 26 -  NS Objects                58 -  SS Tape Storage Segments
 27 -  NS Text Extensions        59 -  SS Disk Physical Volumes
 28 -  Log Client Configurations 60 -  SS Tape Physical Volumes
 29 -  Log Daemon Configurations 61 -  SS Disk Virtual Volumes
 30 -  Log Policies              62 -  SS Tape Virtual Volumes
 31 -  Location Server Policies  63 -  <All of the above>
 32 -  Metadata Monitor Configuration


Select type of file to create (<RETURN> for Main Menu)> 63

Creating file serverconfig...done
Creating file accounting...done
Creating file bfmigrrec...done
Creating file bfpurgerec...done
Creating file bfs...done
Creating file bitfile...done
Creating file bfcoschange...done
Creating file bftapesegment...done
Creating file bfdisksegment...done
Creating file bfdiskallocrec...done
Creating file bfsssegchkpt...done
Creating file bfssunlink...done
Creating file cos...done
Creating file hierarchy...done
Creating file ndcg...done
Creating file cns...done
Creating file cnsobjects...done
Creating file cnstext...done
Creating file cnsacls...done
Creating file logclient...done
Creating file logdaemon...done
Creating file logpolicy...done
Creating file mmonitor...done
Creating file mps...done
Creating file migpolicy...done
Creating file purgepolicy...done
Creating file mpchkpt...done
Creating file mountd...done
Creating file mover...done
Creating file moverdevice...done
Creating file nfs2...done
Creating file pvl...done
Creating file pvlactivity...done
Creating file pvldrive...done
Creating file pvljob...done
```

```
Creating file pvlpv...done
Creating file pvr...done
Creating file cartridge_3494...done
Creating file cartridge_3495...done
Creating file cartridge_aml...done
Creating file cartridge_operator...done
Creating file cartridge_stk...done
Creating file ss...done
Creating file storageclass...done
Creating file storagemapdisk...done
Creating file storagemaptape...done
Creating file storagesegdisk...done
Creating file storagesegtape...done
Creating file sspvdisk...done
Creating file sspvtape...done
Creating file vvdisk...done
Creating file vvtape...done
```

```
1 - General Server Configurations    33 - Migration/Purge Configurations
  2 - Accounting Policies            34 - Migration Policies
  3 - Account Log Records            35 - Purge Policies
  4 - Account Summary Records        36 - Migration/Purge Checkpoints
  5 - Account Snapshot Records       37 - Mount Daemon Configuration
  6 - Migration Records              38 - Mover Configurations
  7 - Purge Records                  39 - Mover Devices
  8 - BFS Configurations             40 - NFS V2 Daemon Configuration
  9 - Bitfiles                       41 - PVL Configurations
 10 - Bitfile COS Changes            42 - PVL Activities
 11 - Bitfile Tape Segments          43 - PVL Drives
 12 - Bitfile Disk Segments          44 - PVL Jobs
 13 - Bitfile Disk Allocation Maps   45 - PVL Physical Volumes
 14 - BFS Storage Segment Checkpoint 46 - PVR Configurations
 15 - BFS Storage Segment Unlinks    47 - 3494 Cartridges
 16 - Classes of Service             48 - 3495 Cartridges
 17 - DMAP Gateway Configurations    49 - AML Cartridges
 18 - DMAP Gateway File Sets         50 - Operator Cartridges
 19 - File Families                  51 - STK Cartridges
 20 - Hierarchies                    52 - Remote Site Configurations
 21 - Non-DCE Gateway Configurations 53 - Storage Server Configurations
 22 - NS Configurations              54 - Storage Classes
 23 - NS ACL Extensions              55 - SS Disk Storage Maps
 24 - NS Fileset Attrs               56 - SS Tape Storage Maps
 25 - NS Global Filesets             57 - SS Disk Storage Segments
 26 - NS Objects                     58 - SS Tape Storage Segments
```

```
 27 -  NS Text Extensions            59 -  SS Disk Physical Volumes
 28 -  Log Client Configurations     60 -  SS Tape Physical Volumes
 29 -  Log Daemon Configurations     61 -  SS Disk Virtual Volumes
 30 -  Log Policies                  62 -  SS Tape Virtual Volumes
 31 -  Location Server Policies      63 -  <All of the above>
 32 -  Metadata Monitor Configuration


Select type of file to create (<RETURN> for Main Menu)>

Available Operations:

  1 - Create file
  2 - Empty file
  3 - Destroy file
  4 - Query file
  5 - List all files
  6 - Change SFS server (currently "/.:/encina/sfs/hpss")
  7 - Change SFS volume (currently "sfsVolhpss")
  8 - Change SFS filename extension (currently <none>)

Select operation (<RETURN> to exit)>
Current site is: registry server at /.../mercury.ibm.com/subsys/dce/sec/master
Domain changed to: group




bye.
<hpss_encina_register> Status => /usr/lpp/hpss/config/hpss_encina_register
processing completed!
<mkhpss> Status ==> Configure HPSS with Encina completed, continue...



<mkhpss>          Infrastructure Configuration Menu
                  ================================

<mkhpss> Prompt ==>   Select Infrastructure Configuration Option:
<mkhpss>          [1] Configure HPSS with DCE
<mkhpss>          [2] Manage SFS Files
<mkhpss>          [3] Set Up FTP Daemon
<mkhpss>          [4] Set Up Startup Daemon
<mkhpss>          [5] Install HPSS Subsystem On Remote Node
```

```
<mkhpss>          [6] Add SSM Administrative User
<mkhpss>          [7] Start SSM Servers/User Session

<mkhpss>          [E] Re-run hpss_env()
<mkhpss>          [R] Re-configure HPSS
<mkhpss>          [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, R, X]):3


<mkhpss>          Perform FTP Daemon Setup
            ========================



<mkhpss>          Verify User ID
==============

<mkhpss> Status ==> User root; verified; continue...
<hpss_ftpd_config> Status => Perform HPSS FTP Daemon setup
<hpss_ftpd_config> Prompt => Do you wish to use the default port id's?
                  (control port = 4021; data port = 4020)
<hpss_ftpd_config> Reply => (Y) y
0513-095 The request for subsystem refresh was completed successfully.

<hpss_ftpd_config> Status => Add HPSS FTP Daemon entry to /etc/services & inetd.conf
<mkhpss> Status ==> FTP Daemon setup completed, continue...



<mkhpss>          Infrastructure Configuration Menu
            ================================

<mkhpss> Prompt ==>    Select Infrastructure Configuration Option:
<mkhpss>          [1] Configure HPSS with DCE
<mkhpss>          [2] Manage SFS Files
<mkhpss>          [3] Set Up FTP Daemon
<mkhpss>          [4] Set Up Startup Daemon
<mkhpss>          [5] Install HPSS Subsystem On Remote Node
<mkhpss>          [6] Add SSM Administrative User
<mkhpss>          [7] Start SSM Servers/User Session

<mkhpss>          [E] Re-run hpss_env()
<mkhpss>          [R] Re-configure HPSS
```

```
<mkhpss>          [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, R, X]):4


<mkhpss>          Verify User ID
             ==============

<mkhpss> Status ==> User root; verified; continue...


<mkhpss>          Perform HPSS Startup Daemon Setup
             ================================

<mkhpss> Status ==> HPSS Startup Daemon will be invoked at system restart




<mkhpss>          Infrastructure Configuration Menu
             ================================

<mkhpss> Prompt ==>    Select Infrastructure Configuration Option:
<mkhpss>          [1] Configure HPSS with DCE
<mkhpss>          [2] Manage SFS Files
<mkhpss>          [3] Set Up FTP Daemon
<mkhpss>          [4] Set Up Startup Daemon
<mkhpss>          [5] Install HPSS Subsystem On Remote Node
<mkhpss>          [6] Add SSM Administrative User
<mkhpss>          [7] Start SSM Servers / User Session

<mkhpss>          [E] Re-run hpss_env()
<mkhpss>          [R] Re-configure HPSS
<mkhpss>          [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, R, X]):5


<mkhpss>          Perform Subsystem Remote Installation
             ===================================
```

```
<mkhpss>            Verify User ID
                ==============

<mkhpss> Status ==> User root; verified; continue...
<hpss_remote_install> Status => Running /usr/lpp/hpss/config/hpss_remote_install on
mercury
                    by root on Sun Sep 10 12:51:37 CDT 2000

<hpss_remote_install> Status => Perform subsystem tar file generation, this may take a while...

<hpss_remote_install> Error => Error occurred on subsystem tar file generation
<hpss_remote_install> Prompt => Do you wish to continue?
 <hpss_remote_install> Reply => (Y) y

<hpss_remote_install> Prompt => Enter target node name or EXIT to quit
<hpss_remote_install> Reply => (target node name or EXIT) sp2n06

<hpss_remote_install> Prompt => Enter root password for target node: sp2n06
<hpss_remote_install> Reply => (root password:)
<hpss_remote_install> Prompt => Select subsystem to be installed
<hpss_remote_install> Prompt => Valid subsystem: cns, bfs, mps, mvr, pvl, pvr
<hpss_remote_install> Prompt =>              logd, ss, ssm, ndcg, hdm, dmg,
<hpss_remote_install> Prompt =>              nfs2, ls, ftpd, ndcaix, ndcsgi

<hpss_remote_install> Reply => (subsystem name or EXIT) hdm

x ./usr/lpp/hpss/Makefile.macros, 18266 bytes, 36 tape blocks
x ./usr/lpp/hpss/Makefile.rules, 12790 bytes, 25 tape blocks
x ./usr/lpp/hpss/bin
x ./usr/lpp/hpss/bin/archivedump, 36712 bytes, 72 tape blocks
x ./usr/lpp/hpss/bin/create_fsys, 597129 bytes, 1167 tape blocks
x ./usr/lpp/hpss/bin/deldmattr, 36760 bytes, 72 tape blocks
x ./usr/lpp/hpss/bin/getattr, 35361 bytes, 70 tape blocks
x ./usr/lpp/hpss/bin/hdm_admin, 448398 bytes, 876 tape blocks
x ./usr/lpp/hpss/bin/hdmdump, 69384 bytes, 136 tape blocks
x ./usr/lpp/hpss/bin/hpss_hdm, 1009402 bytes, 1972 tape blocks
x ./usr/lpp/hpss/bin/hpss_hdm_dst, 866870 bytes, 1694 tape blocks
x ./usr/lpp/hpss/bin/hpss_hdm_evt, 996123 bytes, 1946 tape blocks
x ./usr/lpp/hpss/bin/hpss_hdm_han, 1261825 bytes, 2465 tape blocks
x ./usr/lpp/hpss/bin/hpss_hdm_mig, 978956 bytes, 1913 tape blocks
x ./usr/lpp/hpss/bin/hpss_hdm_pur, 185582 bytes, 363 tape blocks
x ./usr/lpp/hpss/bin/hpss_hdm_tcp, 1424351 bytes, 2782 tape blocks
x ./usr/lpp/hpss/bin/loadhpssid, 36022 bytes, 71 tape blocks
```

x ./usr/lpp/hpss/bin/loadtree, 89471 bytes, 175 tape blocks
x ./usr/lpp/hpss/bin/prtsess, 16237 bytes, 32 tape blocks
x ./usr/lpp/hpss/bin/set_ns_handle, 22376 bytes, 44 tape blocks
x ./usr/lpp/hpss/bin/setdmattr, 36986 bytes, 73 tape blocks
x ./usr/lpp/hpss/config
x ./usr/lpp/hpss/config/templates
x ./usr/lpp/hpss/config/templates/hdm_config.dat.template, 1122 bytes, 3 tape blocks
x ./usr/lpp/hpss/config/templates/hdm_filesys.dat.template, 375 bytes, 1 tape blocks
x ./usr/lpp/hpss/copyright, 1654 bytes, 4 tape blocks
x ./usr/lpp/hpss/include
x ./usr/lpp/hpss/include/Makefile, 16568 bytes, 33 tape blocks
x ./usr/lpp/hpss/include/Makefile.hdm, 10936 bytes, 22 tape blocks
x ./usr/lpp/hpss/include/acct_config.idl, 4704 bytes, 10 tape blocks
x ./usr/lpp/hpss/include/acct_hpss.idl, 2022 bytes, 4 tape blocks
x ./usr/lpp/hpss/include/acct_interface.h, 9252 bytes, 19 tape blocks
x ./usr/lpp/hpss/include/acct_log.idl, 2606 bytes, 6 tape blocks
x ./usr/lpp/hpss/include/acct_logging.h, 9164 bytes, 18 tape blocks
x ./usr/lpp/hpss/include/acct_metadata.h, 9583 bytes, 19 tape blocks
x ./usr/lpp/hpss/include/acct_summary.idl, 3081 bytes, 7 tape blocks
x ./usr/lpp/hpss/include/api_internal.h, 45274 bytes, 89 tape blocks
x ./usr/lpp/hpss/include/api_log.h, 4269 bytes, 9 tape blocks
x ./usr/lpp/hpss/include/auth_mgr.h, 2427 bytes, 5 tape blocks
x ./usr/lpp/hpss/include/auth_shmem.h, 6293 bytes, 13 tape blocks
x ./usr/lpp/hpss/include/bfs.h, 20405 bytes, 40 tape blocks
x ./usr/lpp/hpss/include/bfs_abort.h, 3874 bytes, 8 tape blocks
x ./usr/lpp/hpss/include/bfs_acct.h, 6993 bytes, 14 tape blocks
x ./usr/lpp/hpss/include/bfs_asynch.h, 2568 bytes, 6 tape blocks
x ./usr/lpp/hpss/include/bfs_cache.h, 20402 bytes, 40 tape blocks
x ./usr/lpp/hpss/include/bfs_config.idl, 5223 bytes, 11 tape blocks
x ./usr/lpp/hpss/include/bfs_control.h, 9539 bytes, 19 tape blocks
x ./usr/lpp/hpss/include/bfs_cos.h, 4837 bytes, 10 tape blocks
x ./usr/lpp/hpss/include/bfs_exlock.h, 3153 bytes, 7 tape blocks
x ./usr/lpp/hpss/include/bfs_funcs.h, 29636 bytes, 58 tape blocks
x ./usr/lpp/hpss/include/bfs_interface.tacf, 1417 bytes, 3 tape blocks
x ./usr/lpp/hpss/include/bfs_interface.tidl, 13365 bytes, 27 tape blocks
x ./usr/lpp/hpss/include/bfs_interface_def.idl, 25533 bytes, 50 tape blocks
x ./usr/lpp/hpss/include/bfs_lock.h, 2801 bytes, 6 tape blocks
x ./usr/lpp/hpss/include/bfs_logging.h, 35097 bytes, 69 tape blocks
x ./usr/lpp/hpss/include/bfs_metadata.h, 29614 bytes, 58 tape blocks
x ./usr/lpp/hpss/include/bfs_ssm.h, 2038 bytes, 4 tape blocks
x ./usr/lpp/hpss/include/cns_Attrs.h, 8271 bytes, 17 tape blocks
x ./usr/lpp/hpss/include/cns_Blocks.h, 15696 bytes, 31 tape blocks
x ./usr/lpp/hpss/include/cns_Compress.h, 3764 bytes, 8 tape blocks
x ./usr/lpp/hpss/include/cns_Config.h, 2448 bytes, 5 tape blocks
x ./usr/lpp/hpss/include/cns_Config.idl, 10232 bytes, 20 tape blocks
x ./usr/lpp/hpss/include/cns_Connect.h, 4062 bytes, 8 tape blocks

x ./usr/lpp/hpss/inclufrx ./usr/lpp/hpss/Makefile.macros, 18266 bytes, 36 tape blocks
x ./usr/lpp/hpss/Makefile.rules, 12790 bytes, 25 tape blocks
x ./usr/lpp/hpss/bin
x ./usr/lpp/hpss/bin/archivedump, 36712 bytes, 72 tape blocks
x ./usr/lpp/hpss/bin/create_fsys, 597129 bytes, 1167 tape blocks
x ./usr/lpp/hpss/bin/deldmattr, 36760 bytes, 72 tape blocks
x ./usr/lpp/hpss/bin/getattr, 35361 bytes, 70 tape blocks
x ./usr/lpp/hpss/bin/hdm_admin, 448398 bytes, 876 tape blocks
x ./usr/lpp/hpss/bin/hdmdump, 69384 bytes, 136 tape blocks
x ./usr/lpp/hpss/bin/hpss_hdm, 1009402 bytes, 1972 tape blocks
x ./usr/lpp/hpss/bin/hpss_hdm_dst, 866870 bytes, 1694 tape blocks
x ./usr/lpp/hpss/bin/hpss_hdm_evt, 996123 bytes, 1946 tape blocks
x ./usr/lpp/hpss/bin/hpss_hdm_han, 1261825 bytes, 2465 tape blocks
x ./usr/lpp/hpss/bin/hpss_hdm_mig, 978956 bytes, 1913 tape blocks
x ./usr/lpp/hpss/bin/hpss_hdm_pur, 185582 bytes, 363 tape blocks
x ./usr/lpp/hpss/bin/hpss_hdm_tcp, 1424351 bytes, 2782 tape blocks
x ./usr/lpp/hpss/bin/loadhpssid, 36022 bytes, 71 tape blocks
x ./usr/lpp/hpss/bin/loadtree, 89471 bytes, 175 tape blocks
x ./usr/lpp/hpss/bin/prtsess, 16237 bytes, 32 tape blocks
x ./usr/lpp/hpss/bin/set_ns_handle, 22376 bytes, 44 tape blocks
x ./usr/lpp/hpss/bin/setdmattr, 36986 bytes, 73 tape blocks
x ./usr/lpp/hpss/config
x ./usr/lpp/hpss/config/templates
x ./usr/lpp/hpss/config/templates/hdm_config.dat.template, 1122 bytes, 3 tape blocks
x ./usr/lpp/hpss/config/templates/hdm_filesys.dat.template, 375 bytes, 1 tape blocks
x ./usr/lpp/hpss/copyright, 1654 bytes, 4 tape blocks
x ./usr/lpp/hpss/include
x ./usr/lpp/hpss/include/Makefile, 16568 bytes, 33 tape blocks
x ./usr/lpp/hpss/include/Makefile.hdm, 10936 bytes, 22 tape blocks
x ./usr/lpp/hpss/include/acct_config.idl, 4704 bytes, 10 tape blocks
x ./usr/lpp/hpss/include/acct_hpss.idl, 2022 bytes, 4 tape blocks
x ./usr/lpp/hpss/include/acct_interface.h, 9252 bytes, 19 tape blocks
x ./usr/lpp/hpss/include/acct_log.idl, 2606 bytes, 6 tape blocks
x ./usr/lpp/hpss/include/acct_logging.h, 9164 bytes, 18 tape blocks
x ./usr/lpp/hpss/include/acct_metadata.h, 9583 bytes, 19 tape blocks
x ./usr/lpp/hpss/include/acct_summary.idl, 3081 bytes, 7 tape blocks
x ./usr/lpp/hpss/include/api_internal.h, 45274 bytes, 89 tape blocks
x ./usr/lpp/hpss/include/api_log.h, 4269 bytes, 9 tape blocks
x ./usr/lpp/hpss/include/auth_mgr.h, 2427 bytes, 5 tape blocks
x ./usr/lpp/hpss/include/auth_shmem.h, 6293 bytes, 13 tape blocks
x ./usr/lpp/hpss/include/bfs.h, 20405 bytes, 40 tape blocks
x ./usr/lpp/hpss/include/bfs_abort.h, 3874 bytes, 8 tape blocks
x ./usr/lpp/hpss/include/bfs_acct.h, 6993 bytes, 14 tape blocks
x ./usr/lpp/hpss/include/bfs_asynch.h, 2568 bytes, 6 tape blocks
x ./usr/lpp/hpss/include/bfs_cache.h, 20402 bytes, 40 tape blocks

x ./usr/lpp/hpss/include/bfs_config.idl, 5223 bytes, 11 tape blocks
x ./usr/lpp/hpss/include/bfs_control.h, 9539 bytes, 19 tape blocks
x ./usr/lpp/hpss/include/bfs_cos.h, 4837 bytes, 10 tape blocks
x ./usr/lpp/hpss/include/bfs_exlock.h, 3153 bytes, 7 tape blocks
x ./usr/lpp/hpss/include/bfs_funcs.h, 29636 bytes, 58 tape blocks
x ./usr/lpp/hpss/include/bfs_interface.tacf, 1417 bytes, 3 tape blocks
x ./usr/lpp/hpss/include/bfs_interface.tidl, 13365 bytes, 27 tape blocks
x ./usr/lpp/hpss/include/bfs_interface_def.idl, 25533 bytes, 50 tape blocks
x ./usr/lpp/hpss/include/bfs_lock.h, 2801 bytes, 6 tape blocks
x ./usr/lpp/hpss/include/bfs_logging.h, 35097 bytes, 69 tape blocks
x ./usr/lpp/hpss/include/bfs_metadata.h, 29614 bytes, 58 tape blocks
x ./usr/lpp/hpss/include/bfs_ssm.h, 2038 bytes, 4 tape blocks
x ./usr/lpp/hpss/include/cns_Attrs.h, 8271 bytes, 17 tape blocks
x ./usr/lpp/hpss/include/cns_Blocks.h, 15696 bytes, 31 tape blocks
x ./usr/lpp/hpss/include/cns_Compress.h, 3764 bytes, 8 tape blocks
x ./usr/lpp/hpss/include/cns_Config.h, 2448 bytes, 5 tape blocks
x ./usr/lpp/hpss/include/cns_Config.idl, 10232 bytes, 20 tape blocks
x ./usr/lpp/hpss/include/cns_Connect.h, 4062 bytes, 8 tape blocks
x ./usr/lpp/hpss/include....
....
x ./usr/lpp/hpss/include/util_stack.h, 3604 bytes, 8 tape blocks
x ./usr/lpp/hpss/lib
x ./usr/lpp/hpss/lib/Makefile.hdm, 4923 bytes, 10 tape blocks
x ./usr/lpp/hpss/src
x ./usr/lpp/hpss/src/Makefile.hdm, 3830 bytes, 8 tape blocks
x ./usr/lpp/hpss/src/cs
x ./usr/lpp/hpss/src/cs/Makefile.hdm, 6975 bytes, 14 tape blocks
x ./usr/lpp/hpss/src/cs/interop
x ./usr/lpp/hpss/src/cs/interop/Makefile.hdm, 6711 bytes, 14 tape blocks
x ./usr/lpp/hpss/src/cs/interop/hpss_interop.c, 16978 bytes, 34 tape blocks
x ./usr/lpp/hpss/src/cs/pdata
x ./usr/lpp/hpss/src/cs/pdata/Makefile.hdm, 6012 bytes, 12 tape blocks
x ./usr/lpp/hpss/src/cs/pdata/mvrprotocol.c, 33823 bytes, 67 tape blocks
x ./usr/lpp/hpss/src/cs/pdata/pdata.c, 18316 bytes, 36 tape blocks
x ./usr/lpp/hpss/src/cs/u_signed64.c, 21601 bytes, 43 tape blocks
x ./usr/lpp/hpss/src/cs/xdr
x ./usr/lpp/hpss/src/cs/xdr/Makefile.hdm, 19627 bytes, 39 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/dmapi
x ./usr/lpp/hpss/src/cs/xdr/dmapi/Makefile, 6581 bytes, 13 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/dmapi/dmg_dm_xdrcstub.c, 100733 bytes, 197 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/dmapi/dmg_dm_xdrsstub.c, 71222 bytes, 140 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/dmapi/hdm_hp_xdrcstub.c, 101721 bytes, 199 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/dmapi/hdm_hp_xdrsstub.c, 85409 bytes, 167 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/dmapi/tcpmsg.c, 14584 bytes, 29 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/dmapi/tcpmsg_proto.h, 4236 bytes, 9 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/dmg_msg.x, 33789 bytes, 66 tape blocks

x ./usr/lpp/hpss/src/cs/xdr/hdm_defs.x, 18115 bytes, 36 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/hdr_to_x.pl, 14517 bytes, 29 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/hpss_dmap_attrs.x, 2389 bytes, 5 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/hpss_iod_xdr.c, 24089 bytes, 48 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/idl_to_x.pl, 21555 bytes, 43 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/mvr_shmem.x, 3953 bytes, 8 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/ss_pvlist_xdr.c, 3608 bytes, 8 tape blocks
x ./usr/lpp/hpss/src/dmapi
x ./usr/lpp/hpss/src/dmapi/Makefile.hdm, 4875 bytes, 10 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm
x ./usr/lpp/hpss/src/dmapi/hdm/Makefile, 12119 bytes, 24 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Acl.c, 85709 bytes, 168 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_DFSApis.c, 28122 bytes, 55 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_DFSAuth.c, 17377 bytes, 34 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_DFSMain.c, 44930 bytes, 88 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_DstProg.c, 44103 bytes, 87 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Epi.c, 163168 bytes, 319 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_EvtProg.c, 33032 bytes, 65 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_EvtRecover.c, 50027 bytes, 98 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_HanProg.c, 59961 bytes, 118 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_IO.c, 127771 bytes, 250 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Log.c, 66457 bytes, 130 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_MainProg.c, 140810 bytes, 276 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_MigMain.c, 11970 bytes, 24 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_MigSocket.c, 10192 bytes, 20 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Migrate.c, 122486 bytes, 240 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Msg.c, 30912 bytes, 61 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Ptran.c, 63632 bytes, 125 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Purge.c, 54596 bytes, 107 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_PurgeMain.c, 12010 bytes, 24 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Sem.c, 11157 bytes, 22 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Shmem.c, 15836 bytes, 31 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Support.c, 85771 bytes, 168 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_TcpProg.c, 37492 bytes, 74 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_adminPackage.c, 43589 bytes, 86 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_checkreg.c, 6460 bytes, 13 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_dfsID_IF.acf, 1123 bytes, 3 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_hpPackage.c, 130210 bytes, 255 tape blocks
x ./usr/lpp/hpss/src/dmapi/sec
x ./usr/lpp/hpss/src/dmapi/sec/Makefile, 7860 bytes, 16 tape blocks
x ./usr/lpp/hpss/src/dmapi/sec/gsz_dmap.c, 7360 bytes, 15 tape blocks
x ./usr/lpp/hpss/src/dmapi/sec/gsz_dmapAuth.c, 21697 bytes, 43 tape blocks
x ./usr/lpp/hpss/src/dmapi/sec/gsz_dmap_dmg.c, 30062 bytes, 59 tape blocks
x ./usr/lpp/hpss/src/dmapi/sec/gsz_dmap_hdm.c, 32099 bytes, 63 tape blocks
x ./usr/lpp/hpss/src/dmapi/sec/gsz_support.c, 17265 bytes, 34 tape blocks

x ./usr/lpp/hpss/src/dmapi/sec/gszapi.c, 28334 bytes, 56 tape blocks
x ./usr/lpp/hpss/tools
x ./usr/lpp/hpss/tools/Makefile.hdm, 4264 bytes, 9 tape blocks
x ./usr/lpp/hpss/tools/dmapi
x ./usr/lpp/hpss/tools/dmapi/Makefile.hdm, 3922 bytes, 8 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools
x ./usr/lpp/hpss/tools/dmapi/dmapitools/Makefile, 3840 bytes, 8 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/archivedump
x ./usr/lpp/hpss/tools/dmapi/dmapitools/archivedump/Makefile, 6702 bytes, 14 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/archivedump/archivedump.c, 30930 bytes, 61 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/common
x ./usr/lpp/hpss/tools/dmapi/dmapitools/common/Makefile, 5160 bytes, 11 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/common/handle.c, 11995 bytes, 24 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/common/parse.c, 2469 bytes, 5 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/create_fsys
x ./usr/lpp/hpss/tools/dmapi/dmapitools/create_fsys/Makefile, 7346 bytes, 15 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/create_fsys/create_fsys.c, 12394 bytes, 25 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/deldmattr
x ./usr/lpp/hpss/tools/dmapi/dmapitools/deldmattr/Makefile, 6702 bytes, 14 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/deldmattr/deldmattr.c, 30370 bytes, 60 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/getattr
x ./usr/lpp/hpss/tools/dmapi/dmapitools/getattr/Makefile, 6640 bytes, 13 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/getattr/getattr.c, 29710 bytes, 59 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/hdmdump
x ./usr/lpp/hpss/tools/dmapi/dmapitools/hdmdump/Makefile, 6649 bytes, 13 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/hdmdump/hdmdump.c, 64091 bytes, 126 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/loadhpssid
x ./usr/lpp/hpss/tools/dmapi/dmapitools/loadhpssid/Makefile, 6927 bytes, 14 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/loadhpssid/loadhpssid.c, 20555 bytes, 41 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/loadtree
x ./usr/lpp/hpss/tools/dmapi/dmapitools/loadtree/Makefile, 6923 bytes, 14 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/loadtree/loadtree.c, 68954 bytes, 135 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/prtsess
x ./usr/lpp/hpss/tools/dmapi/dmapitools/prtsess/Makefile, 6662 bytes, 14 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/prtsess/prtsess.c, 9383 bytes, 19 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/set_ns_handle
x ./usr/lpp/hpss/tools/dmapi/dmapitools/set_ns_handle/Makefile, 6829 bytes, 14 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/set_ns_handle/set_ns_handle.c, 9917 bytes, 20 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/setdmattr
x ./usr/lpp/hpss/tools/dmapi/dmapitools/setdmattr/Makefile, 6727 bytes, 14 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/setdmattr/setdmattr.c, 30588 bytes, 60 tape blocks
x ./usr/lpp/hpss/tools/dmapi/hdm_admin
x ./usr/lpp/hpss/tools/dmapi/hdm_admin/Makefile, 7253 bytes, 15 tape blocks

```
x ./usr/lpp/hpss/tools/dmapi/hdm_admin/hdm_admin.c, 119014 bytes, 233 tape blocks
x ./usr/lpp/hpss/tools/make
x ./usr/lpp/hpss/tools/make/Makefile, 5663 bytes, 12 tape blocks
x ./usr/lpp/hpss/tools/make/make_remote_tar, 6845 bytes, 14 tape blocks
x ./usr/lpp/hpss/tools/make/makedepend, 48974 bytes, 96 tape blocks
x ./usr/lpp/hpss/tools/make/mkhpssdepends, 656 bytes, 2 tape blocks
x ./usr/lpp/hpss/tools/make/post_depend_script, 1078 bytes, 3 tape blocks
<hpss_remote_install> Status => HPSS subsystem: hdm installed on target node: rosebud
ftpGet:NLST .rhosts.hpss; return -1

x ./usr/lpp/hpss/include/util_stack.h, 3604 bytes, 8 tape blocks
x ./usr/lpp/hpss/lib
x ./usr/lpp/hpss/lib/Makefile.hdm, 4923 bytes, 10 tape blocks
x ./usr/lpp/hpss/src
x ./usr/lpp/hpss/src/Makefile.hdm, 3830 bytes, 8 tape blocks
x ./usr/lpp/hpss/src/cs
x ./usr/lpp/hpss/src/cs/Makefile.hdm, 6975 bytes, 14 tape blocks
x ./usr/lpp/hpss/src/cs/interop
x ./usr/lpp/hpss/src/cs/interop/Makefile.hdm, 6711 bytes, 14 tape blocks
x ./usr/lpp/hpss/src/cs/interop/hpss_interop.c, 16978 bytes, 34 tape blocks
x ./usr/lpp/hpss/src/cs/pdata
x ./usr/lpp/hpss/src/cs/pdata/Makefile.hdm, 6012 bytes, 12 tape blocks
x ./usr/lpp/hpss/src/cs/pdata/mvrprotocol.c, 33823 bytes, 67 tape blocks
x ./usr/lpp/hpss/src/cs/pdata/pdata.c, 18316 bytes, 36 tape blocks
x ./usr/lpp/hpss/src/cs/u_signed64.c, 21601 bytes, 43 tape blocks
x ./usr/lpp/hpss/src/cs/xdr
x ./usr/lpp/hpss/src/cs/xdr/Makefile.hdm, 19627 bytes, 39 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/dmapi
x ./usr/lpp/hpss/src/cs/xdr/dmapi/Makefile, 6581 bytes, 13 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/dmapi/dmg_dm_xdrcstub.c, 100733 bytes, 197 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/dmapi/dmg_dm_xdrsstub.c, 71222 bytes, 140 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/dmapi/hdm_hp_xdrcstub.c, 101721 bytes, 199 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/dmapi/hdm_hp_xdrsstub.c, 85409 bytes, 167 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/dmapi/tcpmsg.c, 14584 bytes, 29 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/dmapi/tcpmsg_proto.h, 4236 bytes, 9 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/dmg_msg.x, 33789 bytes, 66 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/hdm_defs.x, 18115 bytes, 36 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/hdr_to_x.pl, 14517 bytes, 29 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/hpss_dmap_attrs.x, 2389 bytes, 5 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/hpss_iod_xdr.c, 24089 bytes, 48 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/idl_to_x.pl, 21555 bytes, 43 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/mvr_shmem.x, 3953 bytes, 8 tape blocks
x ./usr/lpp/hpss/src/cs/xdr/ss_pvlist_xdr.c, 3608 bytes, 8 tape blocks
x ./usr/lpp/hpss/src/dmapi
x ./usr/lpp/hpss/src/dmapi/Makefile.hdm, 4875 bytes, 10 tape blocks
```

x ./usr/lpp/hpss/src/dmapi/hdm
x ./usr/lpp/hpss/src/dmapi/hdm/Makefile, 12119 bytes, 24 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Acl.c, 85709 bytes, 168 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_DFSApis.c, 28122 bytes, 55 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_DFSAuth.c, 17377 bytes, 34 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_DFSMain.c, 44930 bytes, 88 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_DstProg.c, 44103 bytes, 87 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Epi.c, 163168 bytes, 319 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_EvtProg.c, 33032 bytes, 65 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_EvtRecover.c, 50027 bytes, 98 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_HanProg.c, 59961 bytes, 118 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_IO.c, 127771 bytes, 250 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Log.c, 66457 bytes, 130 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_MainProg.c, 140810 bytes, 276 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_MigMain.c, 11970 bytes, 24 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_MigSocket.c, 10192 bytes, 20 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Migrate.c, 122486 bytes, 240 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Msg.c, 30912 bytes, 61 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Ptran.c, 63632 bytes, 125 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Purge.c, 54596 bytes, 107 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_PurgeMain.c, 12010 bytes, 24 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Sem.c, 11157 bytes, 22 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Shmem.c, 15836 bytes, 31 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_Support.c, 85771 bytes, 168 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_TcpProg.c, 37492 bytes, 74 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_adminPackage.c, 43589 bytes, 86 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_checkreg.c, 6460 bytes, 13 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_dfsID_IF.acf, 1123 bytes, 3 tape blocks
x ./usr/lpp/hpss/src/dmapi/hdm/hdm_hpPackage.c, 130210 bytes, 255 tape blocks
x ./usr/lpp/hpss/src/dmapi/sec
x ./usr/lpp/hpss/src/dmapi/sec/Makefile, 7860 bytes, 16 tape blocks
x ./usr/lpp/hpss/src/dmapi/sec/gsz_dmap.c, 7360 bytes, 15 tape blocks
x ./usr/lpp/hpss/src/dmapi/sec/gsz_dmapAuth.c, 21697 bytes, 43 tape blocks
x ./usr/lpp/hpss/src/dmapi/sec/gsz_dmap_dmg.c, 30062 bytes, 59 tape blocks
x ./usr/lpp/hpss/src/dmapi/sec/gsz_dmap_hdm.c, 32099 bytes, 63 tape blocks
x ./usr/lpp/hpss/src/dmapi/sec/gsz_support.c, 17265 bytes, 34 tape blocks
x ./usr/lpp/hpss/src/dmapi/sec/gszapi.c, 28334 bytes, 56 tape blocks
x ./usr/lpp/hpss/tools
x ./usr/lpp/hpss/tools/Makefile.hdm, 4264 bytes, 9 tape blocks
x ./usr/lpp/hpss/tools/dmapi
x ./usr/lpp/hpss/tools/dmapi/Makefile.hdm, 3922 bytes, 8 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools
x ./usr/lpp/hpss/tools/dmapi/dmapitools/Makefile, 3840 bytes, 8 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/archivedump
x ./usr/lpp/hpss/tools/dmapi/dmapitools/archivedump/Makefile, 6702 bytes, 14 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/archivedump/archivedump.c, 30930 bytes, 61 tape

blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/common
x ./usr/lpp/hpss/tools/dmapi/dmapitools/common/Makefile, 5160 bytes, 11 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/common/handle.c, 11995 bytes, 24 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/common/parse.c, 2469 bytes, 5 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/create_fsys
x ./usr/lpp/hpss/tools/dmapi/dmapitools/create_fsys/Makefile, 7346 bytes, 15 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/create_fsys/create_fsys.c, 12394 bytes, 25 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/deldmattr
x ./usr/lpp/hpss/tools/dmapi/dmapitools/deldmattr/Makefile, 6702 bytes, 14 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/deldmattr/deldmattr.c, 30370 bytes, 60 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/getattr
x ./usr/lpp/hpss/tools/dmapi/dmapitools/getattr/Makefile, 6640 bytes, 13 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/getattr/getattr.c, 29710 bytes, 59 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/hdmdump
x ./usr/lpp/hpss/tools/dmapi/dmapitools/hdmdump/Makefile, 6649 bytes, 13 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/hdmdump/hdmdump.c, 64091 bytes, 126 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/loadhpssid
x ./usr/lpp/hpss/tools/dmapi/dmapitools/loadhpssid/Makefile, 6927 bytes, 14 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/loadhpssid/loadhpssid.c, 20555 bytes, 41 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/loadtree
x ./usr/lpp/hpss/tools/dmapi/dmapitools/loadtree/Makefile, 6923 bytes, 14 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/loadtree/loadtree.c, 68954 bytes, 135 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/prtsess
x ./usr/lpp/hpss/tools/dmapi/dmapitools/prtsess/Makefile, 6662 bytes, 14 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/prtsess/prtsess.c, 9383 bytes, 19 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/set_ns_handle
x ./usr/lpp/hpss/tools/dmapi/dmapitools/set_ns_handle/Makefile, 6829 bytes, 14 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/set_ns_handle/set_ns_handle.c, 9917 bytes, 20 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/setdmattr
x ./usr/lpp/hpss/tools/dmapi/dmapitools/setdmattr/Makefile, 6727 bytes, 14 tape blocks
x ./usr/lpp/hpss/tools/dmapi/dmapitools/setdmattr/setdmattr.c, 30588 bytes, 60 tape blocks
x ./usr/lpp/hpss/tools/dmapi/hdm_admin
x ./usr/lpp/hpss/tools/dmapi/hdm_admin/Makefile, 7253 bytes, 15 tape blocks
x ./usr/lpp/hpss/tools/dmapi/hdm_admin/hdm_admin.c, 119014 bytes, 233 tape blocks
x ./usr/lpp/hpss/tools/make
x ./usr/lpp/hpss/tools/make/Makefile, 5663 bytes, 12 tape blocks
x ./usr/lpp/hpss/tools/make/make_remote_tar, 6845 bytes, 14 tape blocks
x ./usr/lpp/hpss/tools/make/makedepend, 48974 bytes, 96 tape blocks
x ./usr/lpp/hpss/tools/make/mkhpssdepends, 656 bytes, 2 tape blocks
x ./usr/lpp/hpss/tools/make/post_depend_script, 1078 bytes, 3 tape blocks
<hpss_remote_install> Status => HPSS subsystem: hdm installed on target node: rosebud
ftpGet:NLST .rhosts.hpss; return -1

```
<hpss_remote_install> Prompt => Select subsystem to be installed
<hpss_remote_install> Prompt => Valid subsystem: cns,  bfs,  mps,  mvr,  pvl, pvr
<hpss_remote_install> Prompt =>              logd, ss,  ssm, ndcg, hdm, dmg,
<hpss_remote_install> Prompt =>              nfs2,  ls, ftpd, ndcaix, ndcsgi
<hpss_remote_install> Prompt =>              ndcsun,  sun
<hpss_remote_install> Reply => (subsystem name or EXIT) exit

<hpss_remote_install> Prompt => Enter target node name or EXIT to quit
<hpss_remote_install> Reply => (target node name or EXIT) exit

<hpss_remote_install> Prompt => Do you wish to keep the generated tar files?
bfs.tar.Z      gen.tar.Z      mps.tar.Z      piofs_ie.tar.Z  ss.tar.Z
cns.tar.Z      inst.tar.Z     mvr.tar.Z      pvl.tar.Z       ssm.tar.Z
ftpd.tar.Z     logd.tar.Z     nfs2.tar.Z     pvr.tar.Z       ndcg.tar.Z
dmg.tar.Z      ls.tar..Z      hdm.tar.Z    ndcaix.tar.Z      ndcsgi.tar.Z
<hpss_remote_install> Reply => (Y) y

<hpss_remote_install> Status => /usr/lpp/hpss/config/hpss_remote_install
processing completed!
<mkhpss> Status ==> Subsystem Remote installation completed...



<mkhpss>            Infrastructure Configuration Menu
                 ================================

<mkhpss> Prompt ==>    Select Infrastructure Configuration Option:
<mkhpss>            [1] Configure HPSS with DCE
<mkhpss>            [2] Manage SFS Files
<mkhpss>            [3] Set Up FTP Daemon
<mkhpss>            [4] Set Up Startup Daemon
<mkhpss>            [5] Install HPSS Subsystem On Remote Node
<mkhpss>            [6] Add SSM Administrative User
<mkhpss>            [7] Start SSM Servers/User Session

<mkhpss>            [E] Re-run hpss_env()
<mkhpss>            [R] Re-configure HPSS
<mkhpss>            [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, R, X]):6


<mkhpss>            Add SSM User
                 ============
```

```
<mkhpss>          Verify DCE is Running
            ====================

<mkhpss> Status ==> DCE is running, continue...
<mkhpss> Prompt ==> Perform DCE login as cell_admin; Please enter cell_admin password
<mkhpss> Reply ===> (cell_admin password:)
<mkhpss> Prompt ==> Enter SSM user id
<mkhpss> Reply ===> (user id:)ssm_adm


DCE: Adding DCE User ‘ssm_adm’ ...

Enter cell_admin password :

Enter User’s Full Name [John Nguyen]: A. Smith
Enter User’s Password [ssm_admin]: ssm_pass
Enter UID [999]: 789
Enter Group [hpss]:
Enter Organization [hpss]:

DCE: Principal ‘ssm_adm’ added
DCE: Account information for ‘ssm_adm’ added
DCE: User ‘ssm_adm’ (A. Smith) added to DCE Registry

AIX: Adding AIX User ‘ssm_adm’ ...
Enter Group [hpss]:

SSM: Adding SSM User ‘ssm_adm’ ...
Enter Hostname (where SAMMI Runtime resides) [mercury]:
Select SAMMI Security Level :
  1. User
  2. Privileged User
  3. Operator
  4. Admin
Enter Security Level [4]: 4
SSM: User ‘ssm_adm’ added.
SSM: The Data Server needs to be recycled for the changes to take effect.


<mkhpss>          Infrastructure Configuration Menu
            ================================
```

```
<mkhpss> Prompt ==>    Select Infrastructure Configuration Option:
<mkhpss>          [1] Configure HPSS with DCE
<mkhpss>          [2] Manage SFS Files
<mkhpss>          [3] Set Up FTP Daemon
<mkhpss>          [4] Set Up Startup Daemon
<mkhpss>          [5] Install HPSS Subsystem On Remote Node
<mkhpss>          [6] Add SSM Administrative User
<mkhpss>          [7] Start SSM Servers / User Session

<mkhpss>          [E] Re-run hpss_env()
<mkhpss>          [R] Re-configure HPSS
<mkhpss>          [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, R, X]): 7

<mkhpss>          Start SSM Session
             =================




<mkhpss>          Verify User ID
             ==============

<mkhpss> Status ==> User root; verified; continue...
<mkhpss> Prompt ==> Do you wish to start SSM servers under user id hpss?
<mkhpss> Reply ===> (Y)

<mkhpss> Prompt ==> Do you wish to start SSM session under default user id? (ssm_adm)
<mkhpss> Reply ===> (Y)

<mkhpss> Prompt ==> Enter SAM2_DISPLAY for displaying SSM session
<mkhpss> Reply ===> (display:chang220:0.0)


Enter SAM2_DISPLAY variable [chang220:0.0]:
Starting SAMMI Session for user 'ssm_adm' ...



<mkhpss>          Infrastructure Configuration Menu
             ================================

<mkhpss> Prompt ==>    Select Infrastructure Configuration Option:
<mkhpss>          [1] Configure HPSS with DCE
<mkhpss>          [2] Manage SFS Files
```

```
<mkhpss>          [3] Set Up FTP Daemon
<mkhpss>          [4] Set Up Startup Daemon
<mkhpss>          [5] Install HPSS Subsystem On Remote Node
<mkhpss>          [6] Add SSM Administrative User
<mkhpss>          [7] Start SSM Servers / User Session


<mkhpss>          [E] Re-run hpss_env()
<mkhpss>          [R] Re-configure HPSS
<mkhpss>          [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, R, X]):x

<mkhpss> Status ==> HPSS infrastructure configuration completed
<mkhpss> Status ==> End Time: Sun Sep 10 12:58:20 CDT 2000

 = = = = = = = = = = = = = = = = = = = = =

 = = = = = = = = = = = = = = = = = = = = =

 = = = = = = = = = = = = = = = = = = = = =

 = = = = = = = = = = = = = = = = = = = = =

 = = = = = = = = = = = = = = = = = = = = =



[root@mercury: / usr / lpp / hpss / config]
```

Appendix G

DCE Cross Cell Administration

G.1 HPSS Trusted Cross Cell Administration

Both DCE and Kerberos support the concept of accepting authentication credentials from "foreign" (non-local) authentication domains (DCE Cells or Kerberos realms.) This feature provides enhanced scalability while also providing the foundation for applications to provide strong authentication without exposing the users password. The mechanism involves establishing a bi-directional trust between cooperative sites. While the details of establishing Cross Cell Trust (XCT) are beyond the scope of this document, a "*DCE Cross Cell Cookbook*" is provided at the end of this section to assist in this endeavor.

The DCE Cross Cell Cookbook *is not intended as a detailed implementation guide. Establishing XCT will involve both the DCE and the Domain Name Server (DNS) Administrators at the cooperative locations. Additional details may be obtained in the DCE and DNS documentation.*

HPSS Release 4.1.1 provides / utilizes limited XCT features to clients using the Generic Security Services (GSS) Parallel FTP Client ("**krb5_gss_pftp_client**") communicating with the GSS HPSS Parallel FTP Daemon (**hpss_pftp_amgr** and the **auth_krb5gss** Authentication Manager).

The Kerberos and DCE GSS Interfaces are not exportable outside the U.S. due to U.S. export restrictions. The HPSS GSS PFTP/D features are not provided with the standard distributed HPSS software; however, are available by special request - pursuant to restrictions.

Clients possessing credentials from a either a Kerberos Key Distribution Center (KDC) or a DCE Registry Service may present their authentication credentials to the GSS PFTPD. These credentials are used to authenticate the end client in the local or "foreign" DCE Registry and to subsequently authorize appropriate file access to HPSS. End user information required by the HPSS PFTP Daemon will be obtained from the originating (local or "foreign") DCE Registry. File access will be subject to the appropriate Access Control Lists (ACLs); e.g., foreign_user, foreign_group, or foreign_other, etc.

In Release 4.1.1, the HPSS Administrator will need to use the insif utility to establish the initial ACL conditions on the end users home directory. Refer to Section I.25 in Appendix I for more information on the insif utility.

A DCE Cell Identifier is stored in the HPSS Name Server metadata for each name objects. Since UNIX does not have any concept of DCE / Kerberos Cells, files and directories may appear (when accessed by some applications; e.g., NFS) to be owned by a person in the local cell since the UNIX Identifier (uid) associated with a file or directory may be assigned to a local user); however, if the

file / directory belongs to an individual in another DCE Cell, access will be properly limited to the real owner as specified by the ACLs in the HPSS Name Server metadata. Future use of the XCT features may be implemented to allow HPSS servers in one DCE Cell to communicate and inter-operate with HPSS servers in other cells (XCT Distributed HPSS.)

Each DCE Registry contains account entries of the format "krbtgt / {cellname}" for its own cell and for each cell with which it has established an XCT relationship; e.g., krbtgt / dce.sandia.gov, krbtgt / lanl.gov. In order for HPSS to utilize XCT, a DCE Extended Registry Attribute (ERA) named HPSS.CellId must be created in each of the cooperating DCE cells. Note that this process requires DCE Registry modification privileges for the respective cells and should be performed by the appropriate DCE Administrative personnel at each site.

The ERA creation command is as follows:

```
dcecp -c xattrschema create /.:/sec/xattrschema/HPSS.CellId \
    -aclmgr {principal r m r m} -annotation {HPSS Trusted Cell Identifier} \
    -applydefs no -encoding integer -intercell reject  - multivalued no \
    -reserved no -unique yes
```

The back slashes (\ ) implies line continuations (Korn shell) and must be omitted if the command is entered on a single line or appropriately substituted by the appropriate line continuation character for the administrators shell. It is recommended that each HPSS site create this ERA even if XCT is not to be established with any other cells. Refer to DCE documentation for detailed explanations of each of the options specified.

For each "**krbtgt/{cellname**}" entry in the local DCE Registry which will accepted as a Trusted HPSS Cross Cell, it is necessary to then "populate" the ERA with a identifier (integer). This number must be unique and, due to the difficulties associated with cell name changes, should be registered with the HPSS reseller prior to assignment of the number. This registration process is not currently implemented. Also currently, the following numbers have been assigned and / or reserved (Contact the HPSS Support Representative for available numbers):

100000	dce.sandia.gov
100001	lanl.gov
100002	spectrum.llnl.gov
100003	Reserved for Oak Ridge National Laboratory
100004	Reserved for Lawrence Berkeley

To populate each "**krbtgt/{cellname}**", the HPSS Administrator (with appropriate DCE Registry Modification privileges) for the local cell should enter the following command for each **krbtgt** entry in the local cell:

```
dcecp -c principal modify krbtgt/{cellname} -add {HPSS.CellId ######}
```

The trusted cells may be determined by entering the following command:

```
dcecp -c principal catalog | grep krbtgt
```

The braces ({}) surrounding the word cellname imply the substitution of the appropriate cellname for each **krbtgt** entry; e.g., dce.sandia.gov. The braces surrounding the "**HPSS.CellId ######**" are part of the "**dcecp**" command and must be typed with the command. Substitute the appropriate DCE Cell Identifier for the **######**.

To determine your local cellname, enter the command "**getcellname**". This command, provided as part of the DCE commands, is usually located in the /**opt/dcelocal/bin** or **/usr/lpp/dce/bin** directory. Your "cellname" is the name that follows the "/.../" in the reply.

Every HPSS managed server builds a "Trusted Cell Table" during initialization. This table consists of one (local cell) or more **cell_map_entry** structures for each "**krbtgt/{cellname}**' in the local DCE Registry. If new XCT entries are added, it is necessary to restart every HPSS managed server to refresh this table prior to accepting XCT requests. Failure, to restart every server may result in HSEC_ERGY messages (or other mapped messages) and/or unexpected/unpredictable results.

The **cell_map_entry** structure is defined as follows:

```
typedef struct cell_map_entry {
    uuid_t              cell_id;
    unsigned32          uid;
    unsigned32          hpss_cell_id;      /* From Extended Attribute */
    unsigned32          hpss_cell_id_alias;
    sec_rgy_handle_t    query_handle;
    sec_rgy_name_t      *cellname;
} cell_map_entry_t;
```

The hpss_cell_id field will be filled with the HPSS.CellId ERA value for each foreign XCT. The "**krbtgt/{local_cell}**" entry which will always be 0. For the local cell entry, the "hpss_cell_id_alias" field will contain the appropriate value specified in the HPSS.CellId ERA. This allows future applications to provide file listings which always indicate the HPSS cell identifier.

The Trusted Cell Table is used to determine which DCE Cell to contact to obtain information regarding HPSS clients (users), to build ACLs for files/directories, and in the future MAY be used to obtain information/communicate with HPSS servers in other DCE cells.

If a remote DCE cell is unavailable (communication disruption or remote cell termination) when a client request (FTP command) is issued, an RPC "time-out" may be observed by the end client and an error will be returned for the request. The request will not be queued; however, the appropriate response(s) should be obtained for subsequent requests after communications has been reestablished with the remote cell. HPSS servers do not need to be restarted.

Create a DCE group by the name **hpss_cross_cell_members** using the command:

```
dcecp -c group create hpss_cross_cell_members
```

Add each trusted cell principal to the group **hpss_cross_cell_members** using the commands:

```
dcecp -c group add hpss_cross_cell_members -add krbtgt/"local_cell"
dcecp -c group add hpss_cross_cell_members -add krbtgt/"other_cell"
...
```

With the exception of the local DCE cell, HPSS Trusted Cells *must* have the HPSS.CellId Extended Registry Attribute specified, and each Remote Cell *must* be in the group **hpss_cross_cell_members**.

G.2 ESNet Cross Cell cookbook

The procedures documented in the ESNet Cross Cell cookbook provides a template for the establishment of a Cross Cell Trust. These procedures describe the procedures used by the ESNet community to establish cross cell trust between two cells. The procedures assume that both cells are running DCE 2.1, and are based on standard **dcecp** and **rgy_edit** commands.

A shell script, **x_cell.sh**, developed by Tom Harper at PNNL to automate these procedures is included at the end of this section as an example.

The procedures in this cookbook are divided into four sections:

1. DNS requirements (for both sites)
2. Site A setup
3. Site A account creation procedures,
4. Site B cell connect procedures.

Cell Admin authority is required for both site A and site B participants. The participants must have a means of securely agreeing on an account password.

DNS Requirements

The GDA servers of both cells must be running, and must have access to DNS. The GDA service is handled by one or more **gdad** processes that might be running anywhere in your cell. You can test that your cell has a running **gdad** by using the command "**dcecp -c dir show /.:/**" and looking for the entry **CDS_GDAPointers**, which will show you which server is running **gdad** services. You can double check by logging into that server and grepping "**gdad**" from the **ps** output.

Systems must have an **/etc/resolv.conf** entry pointing to a DNS name server. To ensure that your cell can find the other cell, make sure that you can run commands such as "**nslookup -type=txt dce.es.net"** from the machine that is running **gdad.**

In response to this command you should see a list of text records for **dce.es.net**.

Your own cell must also be registered in DNS. The hostnames of your CDS servers must be registered in DNS, and a special TXT record for at least your master CDS must be added. To get the TXT record for your cell, use the command:

cdscp show cell /.:/ as dns

For example, if the command produces the result:

TXT = 1 67dd7158-3572-11cf-bb37-0800094f9b2e Master /.../dce.sandia.gov/ sass1621_ch 2a6b0532-bd95-11cf-ba57-84af6d0aaa77

Then your DNS TXT record for dce.sandia.gov would be:

TXT = 1 67dd7158-3572-11cf-bb37-0800094f9b2e Master /.../dce.sandia.gov/ sass1621_ch 2a6b0532-bd95-11cf-ba57-84af6d0aaa77 sass1621.sandia.gov

Note that the hostname of the server was appended and the entire string (which is really one long string without line breaks) is enclosed in double quotes. If you want read-only CDS replicas visible to outside cells you would add those entries to the text record as well.

Details specifying how these TXT records are added to your DNS server is beyond the scope of this cookbook. Contact the DNS administrator at your site for assistance, if necessary.

Site A Setup Procedures

Site A will need to establish a special group with appropriate ACLs for performing cross cell trust as follows:

1. dce_login as cell_admin or equivalent.
2. Create group cross_cell_admin:

   dcecp -c group create cross_cell_admin -inprojlist yes
3. Add this group with appropriate permissions to **krbtgt**

   dcecp -c acl mod /.:/sec/principal/krbtgt -add {group cross_cell_admin i}
4. Give acct_admin permission to add users to this group (optional):

   dcecp -c acl mod /.:/sec/group/cross_cell_admin -add {group acct_admin rtM}
5. Set the default permission for this group on keys added to **ktbtgt**:

   dcecp -c acl mod /.:/sec/principal/krbtgt -io -add {group cross_cell_admin rmaug}
6. Add this group to group/none and org/none with rt and M permission:

   dcecp -c acl mod /.:/sec/group/none -add {group cross_cell_admin rtM}

   dcecp -c acl mod /.:/sec/org/none -add {group cross_cell_admin rtM}

Site A Account Creation Procedures

Site A will also need to create an account for the individual(s) at site B that will be establishing and maintaining this trust relationship. These procedures give the individual from site B limited privileges on the site A system. The administrator at site A has the option of further limiting the account after cross-cell trust is established. This procedure does it the hard way - you may want to accomplish the same using a GUI tool or using the "**dcecp -c user create"** command.

1. dce_login as cell_admin or equivalent.

   Note that you might be able to use **account_admin** or some lesser privileged account if the acl's on the group **cross-cell-admin** give you **rtM** permissions.

2. Create the user's principal name:

   dcecp princ create <name> -quota 8 -fullname "<Full Name>"

   Note: Setting the quota to 2 is the minimum required to get the job done, but may not be expedient. If the connect fails you might have to reset the quota before trying again. A value of 8 is reasonable.

3. Add the user to the primary group and organization:

   dcecp -c group add <group> -member <name>

   dcecp -c org add none -member <name>

You probably don't want to make cross_cell_admin this user's primary group, and it may be prudent to make their primary group something less privileged than 'none.' At Sandia, for example, all users belong to the same primary group called "all" that is never deleted and has no privileges.

4. Create the account for this user:

   dcecp -c account create <name> -org none -group <group> -password <passwd>

   Enter Your Password: (enter your login password from step 1)

   Be sure to specify the primary groupname for <group> that you used in step 3.

5. Add the user to the cross_cell_admin group.

   dcecp -c group add cross_cell_admin -member <name>

Now site A is ready to inform site B that everything is ready for performing the site B procedures below. After the site B procedures successfully complete, site A can optionally remove the user from the cross_cell_admin group and set the account as invalid:

dcecp -c account modify <name> -acctvalid no

dcecp -c group remove cross_cell_admin -member <name>

Site B Cell Connect Procedures

Use **rgy_edit** rather than **dcecp** to perform these procedures because of potential implementation problem in the **dcecp registry connect** command. The dcecp command seems to have a hardwired check for membership in **cell_admin** or **account_admin** rather than using the appropriate acls.

1. dce_login as cell_admin or equivalent.

2. Run **rgy_edit cell** command and respond to prompts as below:

   rgy_edit

   rgy_edit> cell /.../<Site A cellname>

   rgy_edit> Enter group name of the local account for the foreign cell: none

   rgy_edit> Enter group name of the foreign account for the local cell: none

   rgy_edit> Enter org name of the local account for the foreign cell: none

   rgy_edit> Enter group name of the foreign account for the local cell: none

   rgy_edit> Enter your password: <your dce_login password from step 1>

   rgy_edit> Enter password for foreign account: <your site B account password>

   rgy_edit> Enter expiration date [yy/mm/dd or 'none']: none

   Principals and Accounts have been created

   rgy_edit> exit

   bye

What this account does for you is to create **/.:/sec/principal/krbtgt/<site A cellname>** in your local cell (using your cell_admin privileges), and creates an equivalent principal and account (**/.../<site A cell>/sec/principal/krbtgt/<SiteB cellname>**) in the foreign cell. The entries in the foreign cell are created using privileges and quotas for your user account in site A.

Each side should now be able to do a:

dcecp -c account show krbtgt/<foreign_cell_name>

to verify that the cross cell trust accounts have been created.

A more conclusive test would verify that local authentication gives the user access to objects in the foreign cell. If the foreign cell has a DFS service running with a directory that has no access for "unauthenticated", but read and execute for "any_other" a simple test is to do a dce_login to your local cell, and cd into the foreign cell's directory:

cd /.../<foreign_cell_name>/fs/<pathname>

ls -l

If this command succeeds, and if the ACLs on the directory specified in <pathname> don't permit unauthenticated read, you can assume that your local credentials are valid in the foreign cell.

Acknowledgments

The initial procedures were based on Neil Fraser's paper, Connecting DCE Cells, presented at the OSF DCE User and Developer Conference in San Jose, CA, April 11, 1995.

Melissa Myerly at Sandia revised and tested the procedures to use **dcecp** and the **cross_cell_admin** group.

Tom Harper and Troy Thompson (PNNL), Joa Ramus (NERSC), Doug Engert (ANL) participated in developing and testing the cookbook.

Cross Cell Script

Following is the x_cel.sh script written by Tom Harper at PNNL that automates the entire process of establishing a Cross Cell Trust.

x_cell.sh

```
#!/bin/sh -

#<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
#
#Cross Cell Authentication Shell Script(csa.sh)
#
#This script attempts to automate portions of the ESnet
#Cross Cell Cookbook and steals freely from both the
#commands and text of that document.  For more info on the
#Cookbook, contact Pat Moore at Sandia National Laboratories
#(pcmoore@sandia.gov).
#
#---------------------------------------------------------------------------
#V 0.0  (corresponds to Cookbook Rev 2.1)
#Tom Harper, ta_harper@pnl.gov
#October 18, 1996
#---------------------------------------------------------------------------
#V 0.1  Fix dcecp account create, remove no authentication option,
#             add a path statement so SGI's and HP's can find "which",
#       test for pre-existing cross-cell acount.
#Tom Harper, ta_harper@pnl.gov
#December 4, 1996
#---------------------------------------------------------------------------

## Set DEBUG and/or VERBOSE to "1" for diagnostic output.
## These can also be set on the command line via -d -v

DEBUG=
VERBOSE=

PATH=/bin:/usr/bin:/usr/ucb/bin:/usr/sbin:/usr/5bin:/usr/local/bin
export PATH
```

```
DCECP=`which dcecp`
RGY_EDIT=`which rgy_edit`
NSLOOKUP=`which nslookup`
HOSTNAME=`hostname`

#<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
#
#  Perform Sanity Checks
#
#-------------------------------------------------------------------------------

perform_sanity_checks()
{
 if [ $DEBUG ]; then echo "\n#D# perform_sanity_checks()";  fi

 if [ ! -x $DCECP ]; then
  echo " This machine does not appear to have the dcecp binary.  Exiting."
  exit 1
 elif [ $DEBUG ]; then
  echo "#D# Using $DCECP"
 fi

 if [ ! -x $RGY_EDIT ]; then
  echo " This machine does not appear to have the rgy_edit binary. Exiting."
  exit 1
 elif [ $DEBUG ]; then
  echo "#D# Using $RGY_EDIT"
 fi

 if [ ! -x $NSLOOKUP ]; then
  echo " This machine does not appear to have nslookup.  Exiting."
  exit 1
 elif [ $DEBUG ]; then
  echo "#D# Using $NSLOOKUP";
 fi

 CELL=`$DCECP -c cell show /.: | grep "\.\.\." | cut -d"/" -f3 | head -1`
 if [ ! $CELL ]; then
  echo ""
  echo " $HOSTNAME does not appear to be part of a DCE cell.  Exiting"
  exit 1
 elif [ $DEBUG ]; then
  echo "#D# Local DCE cell: $CELL"
 fi

 GDAs=`$DCECP -c dir show /.: | grep _gda | cut -d"/" -f4 | cut -d_ -f1`
 if [ ! "$GDAs" ]; then
  echo ""
  echo " Cell $CELL does not appear to have any GDAs running.  Exiting."
  exit 1
 elif [ $VERBOSE ]; then
  echo "#V# GDAs running on: "
  echo "$GDAs"
```

```
  echo ""
 fi

 DNS=`$NSLOOKUP -type=txt $CELL | grep "\.\.\."`
 if [ ! "$DNS" ]; then
  echo ""
  echo " Cell $CELL does not appear to be registered in DNS.  Exiting."
  exit 1
 elif [ $VERBOSE ]; then
  echo "#V# DNS entry for $CELL :"
  echo "$DNS"
  echo ""
 fi

 return 0
}

#<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
#
#  Choose A or B
#
#-------------------------------------------------------------------------
choose_a_or_b()
{
 if [ $DEBUG ]; then echo "\n#D# choose_a_or_b()"; fi

 while true
 do
  echo ""
  echo " Perform procedure for DCE site a or b? [b] \c" >&$STDOUT
  read AorB
  if [ -z "$AorB" ]; then
    AorB="b"
    break
  fi
  case "$AorB" in
     [Aa])
        AorB="a"
        break
        ;;
     [Bb])
        AorB="b"
        break
        ;;
     *)
        echo " Please enter either a or b "
        ;;
  esac
 done

 return 0
}

#<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
```

```
#
#  Util Chkstr :: from the util dce_module by vanryzin
#
#---------------------------------------------------------------------------
util_ckstr()
{
   prompt="$1"
   help="$2"
   default="${3-}"

   if [ $DEBUG ]; then echo "\n#D# util_ckstr()"; fi

   while true
   do
      if [ -z "$default" ]
      then
         echo "${prompt}: (<string>, q, ?) \c" >&$STDOUT
      else
         echo "${prompt}: (<string>, q, ?) [$default] \c" >&$STDOUT
      fi
      read ans
      if [ -z "$ans" ]
      then
         ans="$default"
      fi
      case "$ans" in
         'q')
            echo 'Prompting aborted.'
            return 1
            ;;
         '?')
            echo "$help"
            ;;
         *)
            return 0
            ;;
      esac
   done
}

#<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
#
#  Get Password
#
#---------------------------------------------------------------------------
get_password()
{
USER="$1"

  if [ $DEBUG ]; then echo "\n#D# get_password()"; fi

  echo " Enter Password for $USER"
  trap 'stty echo; exit' 1 2
  stty -echo
```

```
  read ans
  echo " >&$STDOUT
  stty echo
  trap 1 2

}

#<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
#
#   Check DCECP command status
#
#---------------------------------------------------------------------------
check_dcecp_command_status()
{
CMD_STATUS=$1

  if [ $DEBUG ]; then echo "\n#D# check_dcecp_command_status()"; fi

  if [ $CMD_STATUS -ne 0 ]; then
    echo ""
    echo "  The dcecp command did not complete successfully."
    echo "  Please contact your DCE system administrator for additional"
    echo "  assistance.  Exiting."
    echo ""
    exit 1
  fi
}

#<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
#
#   A Site Procedure
#
#---------------------------------------------------------------------------
a_site_procedure()
{
  if [ $DEBUG ]; then echo "\n#D# a_site_procedure()"; fi

  a_site_setup
  a_site_account_creation
}

#<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
#
#  A Site Setup
#
#---------------------------------------------------------------------------
a_site_setup()
{
  if [ $DEBUG ]; then echo "\n#D# a_site_setup()"; fi

  echo " Create group cross_cell_admin"
  RESULT=`$DCECP -c group catalog | grep cross_cell_admin`
  if [ -n "$RESULT" ]; then
    echo "    The group cross_cell_admin exists.  Skipping group create."
```

```
else
  $DCECP -c group create cross_cell_admin -inprojlist yes
  check_dcecp_command_status $?
fi

echo " Add group cross_cell_admin to krbtgt "
RESULT=`$DCECP -c acl show /.:/sec/principal/krbtgt|grep cross_cell_admin`
if [ -n "$RESULT" ]; then
  echo "    The principal is already on the krbtgt ACL."
else
  $DCECP -c acl mod /.:/sec/principal/krbtgt -add {group cross_cell_admin i}
  check_dcecp_command_status $?
fi

echo " Give acct-admin permission to add users to this group "
RESULT=`$DCECP -c acl show /.:/sec/group/cross_cell_admin|grep acct-admin`
if [  -n "$RESULT" ]; then
  echo "    The account is already on the ACL."
else
  $DCECP -c acl mod /.:/sec/group/cross_cell_admin -add \
                                    {group acct-admin rtM}
  check_dcecp_command_status $?
fi

echo " Set the default permission for this group on keys added to krbtgt"
RESULT=`$DCECP -c acl show /.:/sec/principal/krbtgt|grep cross_cell_admin`
if [ -n "$RESULT" ]; then
  echo "    The group is already on the krbtgt ACL."
else
  $DCECP -c acl mod /.:/sec/principal/krbtgt -add \
                                  {group cross_cell_admin rmaug}
  check_dcecp_command_status $?
fi

echo " Add the group cross_cell_admin to group none"
RESULT=`$DCECP -c acl show /.:/sec/group/none|grep cross_cell_admin`
if [ -n "$RESULT" ]; then
  echo "    The group is already on the ACL."
else
  $DCECP -c acl mod /.:/sec/group/none -add {group cross_cell_admin rtM}
  check_dcecp_command_status $?
fi

echo " Add the group cross_cell_admin to org none"
RESULT=`$DCECP -c acl show /.:/sec/org/none|grep cross_cell_admin`
if [ -n "$RESULT" ]; then
  echo "    The group is already on the ACL."
else
  $DCECP -c acl mod /.:/sec/org/none -add {group cross_cell_admin rtM}
  check_dcecp_command_status $?
fi

}
```

```
#<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
#
#  A Site Account Creation
#
#-----------------------------------------------------------------------------
a_site_account_creation()
{

  if [ $DEBUG ]; then echo "\n#D# a_site_account_creation()"; fi

## Gather needed information ##

  prompt=" Cross cell user's principal name"
  help="
 This is the name of the principal which will be used to establish
 cross cell trust between your site and site B.  The actual name used
 is not important as long as both the A and B sites agree to use the
 same name."
  util_ckstr "$prompt" "$help" || exit 1
  NAME="$ans"

  prompt=" Full name for principal $NAME (remember quotes)"
  help="
 This is a text field to store human readable information about $NAME"
  util_ckstr "$prompt" "$help" || exit 1
  FULL_NAME="$ans"

  prompt=" CDS creation quota for $NAME"
  help="
 This quota controls how many entries you can insert into the CDS
 namespace.  A minimum of 2 is required for the cross cell authentication. "
  util_ckstr "$prompt" "$help" "10" || exit 1
  QUOTA="$ans"

  prompt=" Primary group for $NAME"
  help="
 You probably don't want to make cross_cell-admin the primary group
 for this user, and it may be prudent to make the primary group something
 less privileged than group none."
  util_ckstr "$prompt" "$help" "none" || exit 1
  GROUP="$ans"

  get_password $NAME;
  U_PASS="$ans"

## Take action in the registry ##

  echo "  Create principal $NAME"
  RESULT=`$DCECP -c prin catalog | grep $NAME`
  if [ -n "$RESULT" ]; then
    echo "    The principal $NAME exists.  Skipping principal create."
  else
    if [ $DEBUG ]; then
      echo "
```

```
    $DCECP -c principal create $NAME -quota $QUOTA -fullname $FULL_NAME"
   fi
   $DCECP -c principal create $NAME -quota $QUOTA -fullname "$FULL_NAME"
   check_dcecp_command_status $?
  fi


  echo "  Add $NAME to group $GROUP"
  RESULT=`$DCECP -c group list $GROUP | grep $NAME`
  if [ -n "$RESULT" ]; then
   echo "    The principal $NAME is already in group $GROUP."
  else
   if [ $DEBUG ]; then
     echo "
     $DCECP -c group add $GROUP -member $NAME"
   fi
   $DCECP -c group add $GROUP -member $NAME
   check_dcecp_command_status $?
  fi


  echo "  Add $NAME to organization none"
  RESULT=`$DCECP -c org list none | grep $NAME`
  if [ -n "$RESULT" ]; then
   echo "    The principal $NAME is already in organization none."
  else
   if [ $DEBUG ]; then
     echo "
     $DCECP -c org add none -member $NAME"
   fi
   $DCECP -c org add none -member $NAME
   check_dcecp_command_status $?
  fi


  echo "  Create the account for $NAME"
  RESULT=`$DCECP -c account cata | grep $NAME`
  if [ -n "$RESULT" ]; then
   echo "    The account $NAME already exists."
  else
   if [ $DEBUG ]; then
     echo "
     $DCECP -c account create $NAME -org none -group $GROUP -password ******"
   fi

 ## V0.1 Kludge - TAH
    rm /tmp/dcecp_command_$$ > /dev/null 2>&1
    echo " account create $NAME -org none -group $GROUP -password $U_PASS " \
        > /tmp/dcecp_command_$$

    $DCECP /tmp/dcecp_command_$$
    rm /tmp/dcecp_command_$$
    check_dcecp_command_status $?
  fi
```

```
  echo "  Add $NAME to the cross_cell_admin group"
  RESULT=`$DCECP -c group list cross_cell_admin | grep $NAME`
  if [ -n "$RESULT" ]; then
    echo "    The account $NAME is already in group cross_cell_admin."
  else
    if [ $DEBUG ]; then
      echo "
      $DCECP -c group add cross_cell_admin -member $NAME"
    fi

    $DCECP -c group add cross_cell_admin -member $NAME
    check_dcecp_command_status $?
 fi

  echo "

Congratulations!  Now, inform site B that you are ready
to proceed with establishing cross-cell trust! "

}

#<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
#
#  Authenticate as cell_admin
#
#----------------------------------------------------------------------------
authenticate_as_admin()
{

  if [ $DEBUG ]; then echo "\n#D# authenticate_as_admin()"; fi

  if [ $VERBOSE ]; then
    echo ""
    echo " Identity information of the principal running this script:"

    if [ -x /krb5/bin/klist ]; then
      IDENT=`/krb5/bin/klist | grep Default`
    elif [ -x /bin/kinfo ]; then
      IDENT=`/bin/kinfo | grep Identity`
    fi
    echo " $IDENT"
  fi

  util_ckstr " Administrative account name" "No help available" \
           "cell_admin" || exit 1
  ADMIN_ACCOUNT="$ans"

  get_password  "$ADMIN_ACCOUNT"
  PASS="$ans"

#
#  Insert code here to mail admin passwd's to 3l33t-1@phrack.net  ;)
```

```
#

  ln -s /bin/dce_login /tmp/dce_login_noexec

  KRB5CCNAME=`/tmp/dce_login_noexec cell_admin $PASS 2>/dev/null` \
  export KRB5CCNAME

  if [ $? -ne 0 ] ; then
    echo ""
    echo " dce_login  for $ADMIN_ACCOUNT failed.  Exiting."
    exit 1
  elif [ $VERBOSE ]; then
    echo " dce_login as $ADMIN_ACCOUNT succeeded."
  fi

  rm /tmp/dce_login_noexec

  if [ $VERBOSE ]; then
    echo ""
    echo " New DCE Identity of the principal running this script:"

    if [ -x /krb5/bin/klist ]; then
      IDENT=`/krb5/bin/klist | grep Default`
    elif [ -x /bin/kinfo ]; then
      IDENT=`/bin/kinfo | grep Identity`
    fi
    echo " $IDENT"
  fi

}

#<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
#
#  b_site_procedure()
#
#---------------------------------------------------------------------------
b_site_procedure()
{

  if [ $DEBUG ]; then echo "\n#D# b_site_procedure()"; fi

  prompt=" Fully qualified name of remote cell:"
  help="
 This is the name of the other cell you are attempting to setup cross
 cell trust with.  Remember to use the fully qualified cell name which
 begins with /.../<cell name>"
  util_ckstr "$prompt" "$help" "/.../dce.es.net" || exit 1
  FOREIGN_CELL="$ans"

  prompt=" Account name to use in remote cell:"
  help="
 The name of the account you agreed to use with the site doing part A
 of this cookbook."
  util_ckstr "$prompt" "$help" "ramus" || exit 1
```

```
  FOREIGN_ACCOUNT="$ans"

  get_password $FOREIGN_ACCOUNT
  FOREIGN_PASSWORD="$ans"

  prompt=" Admin account name to use in local cell:"
  help="
 The name of the administrative account you are using"
  util_ckstr "$prompt" "$help" "cell_admin" || exit 1
  ADMIN_ACCOUNT="$ans"

  get_password $ADMIN_ACCOUNT
  ADMIN_PASSWORD="$ans"

  if [ $DEBUG ]; then
    echo " Calling rgy_edit with :
    cell $FOREIGN_CELL -gl none -ol none -gf none -of none
     -mp ********** -fa $FOREIGN_ACCOUNT -fp ********** -x none"
  fi

  $RGY_EDIT -up <<- EOT
    cell $FOREIGN_CELL -gl none -ol none -gf none -of none \
     -mp $ADMIN_PASSWORD -fa $FOREIGN_ACCOUNT -fp $FOREIGN_PASSWORD -x none
EOT


}
#<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
#
#  Main
#
#-------------------------------------------------------------------------------

USAGE="\n
  $0 [ d | D ] [ v | V ] [ n | N ] \n
  d: debug\t\t enable debugging output \n
  v: verbose\t\t enable verbose informational output \n"

  while getopts dDvV cmdline_args
  do
    case $cmdline_args in
      d | D)   DEBUG=1;;
      v | V)   VERBOSE=1;;
      \?)      echo $USAGE; exit 2;;
    esac
  done
  shift `expr $OPTIND - 1`

  if [ $DEBUG ]; then echo "\n#D# $0 : main()"; fi

  perform_sanity_checks
  choose_a_or_b
  authenticate_as_admin
  eval ${AorB}_site_procedure
```

exit 0

G.3 HPSS Extended Attribute Administration

Three HPSS Extended Registry Attributes (ERAs) have been defined: **HPSS.CellId**, **HPSS.homedir**, and **HPSS.gecos**. These ERAs must be created and then appropriately filled for use with HPSS. Refer to the section G.1 for details about the HPSS.CellId ERA. The other two HPSS ERA fields are used to specify the locations of each HPSS clients (users) HPSS home directory and **gecos** (Accounting and/or Full Name Information.). The following processes must be performed by a person with appropriate DCE Registry administrative privileges. The command for creation of these fields follows:

```
dcecp -c xattrschema create /.:/sec/xattrschema/HPSS.homedir \
    -aclmgr {principal r m r m} -annotation {HPSS Home Directory Specifier} \
    -applydefs no -encoding printstring -intercell reject  -multivalued no \
    -reserved no -unique no

dcecp -c xattrschema create /.:/sec/xattrschema/HPSS.gecos \
    -aclmgr {principal r m r m} -annotation {HPSS Accounting Information} \
    -applydefs no -encoding printstring -intercell reject  -multivalued no \
    -reserved no -unique no
```

These Extended Registry Attributes may then be filled for each HPSS user using the following commands:

```
dcecp -c principal modify USERNAME -add {HPSS.homedir PATH}

dcecp -c principal modify USERNAME -add {HPSS.gecos AA=123456}
```

The USERNAME, PATH, and AA=123456 terms above should be substituted with the appropriate information. The "AA=123456" may be substituted with information containing spaces by using quotes (") around the information.

Modification of these fields MUST be restricted to authorized personnel. Currently, if a customer is given permission to modify these fields, he/she is allowed to modify all these ERAs for all customers. If an HPSS site wishes to allow a customer to change these ERA(s), it will be necessary to create a "Trusted" client utility to perform this function. This utility is not provided with HPSS.

The ERA information for a customer may be changed (by authorized personnel) by substituting "-change" for "-add" in the preceding commands.

The HPSS Parallel FTP Daemon accesses these fields from the appropriate DCE Registry if the "USE_EXTENDED_ATTRIBUTES" flag (-X) is specified on the initiation line for the Daemon (See

FTP sections of the HPSS Administrative Guide for FTP Daemon options.) If this option is specified, the HPSS Parallel FTP Daemon will first obtain the standard (non-ERA) DCE information for the home directory and/or **gecos** of the customer then attempt to read the appropriate HPSS ERA fields if available. If the ERAs are not filled, the non-ERA fields from the DCE Registry will be used.

G.4 Using insif to Establish ACLs for Foreign Customers

Foreign customers may authenticate to the HPSS Parallel FTP Daemon using credentials obtained from a trusted remote DCE Registry Server. The HPSS Name Server establishes and regulates access requirements to the HPSS directories and files. It is necessary to establish "**foreign_other**" access ACLs to upper level HPSS directories and "**foreign_user**" ACLs to customer directories and files. The ACLs for files are typically established by the customer at the time of file creation. Currently the only method for establishing **foreign_other** ACLs on HPSS directories is using the HPSS **insif** utility.

The **insif** utility should be limited to HPSS Administrators only. Instructions for establishing ACLs are given below:

- Establish the **insif** environment. This is easiest if a file; e.g., insif.env, can be "sourced". This file contains environment variables required by the **insif** utility.
- Initiate **insif**
- If the site uses a UID other than 0 as the HPSS Root account, enter the command **uc UID=######** where **######** represents the uid associated with the HPSS Root account.

  1. cd to the desired directory.
  2. Issue the command **getacl**. This will list the ACLs associated with the current HPSS directory.
  3. Issue the command **updacl**.
  4. Answer the first question with the "a" (no option)
  5. Choose the type of ACL to be set; e.g., i ==> **foreign_other**.
  6. Choose the permission bits to be set; e.g., **rx**
  7. Enter a 0 for the expiration date.
  8. Enter a 0 for the entry id if the type is other or the specific UID from the remote cell if the type is either **foreign_user** or **foreign_group**.
  9. Enter the appropriate remote HPSS Cell Id for the Location. (e.g., 100000 ==> Sandia National Laboratories, 100001 ==> Los Alamos National Laboratories, 100002 ==> Lawrence Livermore National Laboratories, 100003 ==> Oak Ridge National Laboratories. See discussion on establishing Trusted Cross Cell HPSS for details on the HPSS Cell Id.

Repeat steps 1 - 9 as needed for all directories/files.

Foreign_other with permissions **rx** should be set on the root directory for each trusted cell. Similarly customer upper level directories should have foreign_other ACLs to allow customers to be able to traverse the HPSS tree. Private customer directories should have appropriate permissions set for the **foreign_user**. Customers desiring to allow other foreign customers access to their home directory and/or files will need to have the HPSS administrator establish the appropriate ACLs for their directory and/or files.

The first **updacl** on a directory/file will establish the **mask_object** ACL with the permissions specified by the first ACL added. It may be necessary to modify the permissions on the **mask_obj** ACL after the first ACL is added.

Appendix H

Additional SSM Information

H.1 Using the SSM Windows

Before using the SSM windows, it is helpful to be aware of some of the conventions used by SSM and by Sammi (on which SSM is based). While the following list does not cover all features of all windows, it does describe the most important points.

- Almost all mouse actions should be performed with the left button. One exception is opening help windows (see Section H.2). Some windows (very few) may use other buttons for special purposes. For example, the HPSS Alarms and Events window uses the left mouse button to select a message for detailed viewing, while the right mouse button is for acknowledging alarms. Such exceptions are documented in the online help.

- Colors are used consistently from window to window. While it is possible for these colors to be customized, the default colors will be used in descriptions below.

- All buttons are goldenrod (a pale orange) in color. Plain buttons perform a single action when clicked with a mouse. Toggle buttons have a small square indicator on the left, and are used to select one of two possible states. When the indicator is red, the button is **ON**; otherwise the button is **OFF**.

  Buttons may be "desensitized". In this state, a button is visible, but its label text is grayed out, and clicking it has no effect. This happens when the current state of the window does not allow the button to be clicked. It also happens when your SSM security level is such that you are not permitted to do any more than view the button (see Section H.7).

- Some buttons are labelled only with three periods, "**...**". Clicking such a button opens a window giving more information on the item next to it. For example, if a "**...**" button is to the right of a field displaying a storage class name, clicking it will open the Storage Class Configuration window giving complete details on that storage class.

- A "text" field is any field which displays alphanumeric text or number data. (This does not include "static" text painted on the window background, or labels on things like buttons.) Text fields may be single or multiple line, and they may be "enterable" (you can alter the displayed data) or "non-enterable" (you cannot change the displayed data directly).

- Most non-enterable text fields have gray backgrounds slightly lighter than the window background, and no borders. Some multi-line fields have the same background color, but use borders to help set them off from their surroundings. Some special fields display a fixed set of text strings, and use different background colors for different strings. These are mostly used to mark status conditions with different colors. For example, the status fields on the Health and Status window display text strings like "**Normal**", "**Warning**", "**Critical**", and so on, with green, yellow, and red background colors to match the severity of the condition.

- Enterable text fields have lighter backgrounds than non-enterable fields, and are framed within borders. To change the contents of a field, move the cursor into the field (with the mouse or the **Tab** key) and type in the desired value. As soon as you type the first character, the field colors change to white text on a blue background. This indicates that the field is being modified. When you are finished typing, press Tab to move to the next field, or Enter. This changes the field back to its normal colors.

SSM does not record what you have entered until you press the Tab or Enter key. If you fill in a configuration window, for example, and click the "Add" button while some fields are still displaying the white-on-blue colors, those fields will not be set correctly in the new configuration record.

  Some enterable text fields are wider than they appear. If you type up to the right edge of such a field, the text automatically scrolls to the left and allows you to keep typing up to the actual size of the field. You can scroll back and forth within the field using the left and right cursor keys. "Scroll indicators" (graphics which look like something between parentheses and square brackets) appear on either end of the field if there is any data beyond the visible edge of the field.

  Enterable fields may be "desensitized". On configuration windows, for example, some fields are enterable when creating a new record, but may not be changed when updating an existing record. A desensitized field has a grayed-out look, and you cannot type anything in it.

  Your SSM security level (see Section H.7) may not be high enough to allow you to change what looks like an enterable field. In this case, the field will not be grayed out, but you will be unable to type anything in it.

- Special pairs of text fields are used for entering "byte count" data. These consist of an enterable field on the left, a non-enterable field on the right, and a ">" character between them to mark the relation between the two. In the enterable field you can enter a number of bytes, or you can enter a number followed by "KB", "MB", "GB", or "TB" to specify kilobytes, megabytes, gigabytes, or terabytes, respectively. (Any of these may be abbreviated to the first letter, and case is ignored.) No matter how the value is entered, it is translated to bytes for display in the non-enterable part of the field.

- An "option list" field is a non-enterable text field surrounded by a goldenrod box. It looks like a button, but it has a small dash-shaped graphic (similar to a Motif option menu widget) on the right end. Clicking one of these fields pops up a list of options, also in goldenrod. You may select one of the displayed options (in which case it will replace the displayed contents of the field), or you can dismiss the list to cancel the change. Note that if there is only one possible option, the popup list will not appear; instead, the only choice will be automatically entered into the field.

  The popup lists are of two types. Where practical, a popup "menu" is used, with all possible options displayed at once. This type can be dismissed by clicking the mouse anywhere outside of the popup. For potentially longer option lists, a "selection list" is used. This type uses scroll-bars, if necessary, to display all the option data, and it has a "**Cancel**" button at the bottom of the list. You must click the "**Cancel**" button to dismiss this type of popup.

- Popup selection lists are used in other places besides being part of option lists. This type of popup is gray in color, but they work the same way as the option list variety.

- A "status line" field is a long non-enterable text field along the bottom of some windows. Status lines display messages which show important information about the state of the window, the progress of some operation started from the window, etc. In most cases, a status line message will be erased as soon as you do something to affect the window (click a button, type something into a field, etc.).

- Each window always has one item which has "input focus". This item is surrounded by a red highlight border. If an enterable text field has input focus, typing anything on your keyboard will enter characters into the field. If a button has input focus, pressing the space bar will have the same effect as clicking the button with the mouse.

- You can do Motif select / cut / paste operations on enterable text fields, but not on non-enterable fields.

H.2 SSM On-line Help

Each SSM window has a help file which describes the features and fields on that window. To access the help for a window, position the mouse cursor to any blank spot on the window, hold down the **Shift** key, and click the right mouse button. A help browser window will open. When finished, click the "**DONE**" button at the bottom of the help browser to dismiss it.

Note that the mouse cursor must be in a blank spot when the button is clicked. If the cursor is over a button, field, or some other dynamic feature, an empty help browser will appear. This is because Sammi thinks you are requesting help on the field, not on the window, and SSM does not include any field-level help. The one exception is the main menu on the Health and Status window. Shift-clicking the right mouse button within this menu bar will display help for the main menu.

The on-line help provides fairly detailed information on all fields, buttons, and other features of each window. It does not, however, provide the detailed planning information provided in this Administration Guide. For example, the on-line help for the Storage Class Configuration window has a description of each field on the window, but it does not include the overall planning advice on how to design storage classes for the most efficient use of resources at your site.

H.3 Viewing SSM Session Information

The **Show Sammi Environment** menu option, under the **Session** heading on the SSM main menu, opens a window which shows the settings of all Sammi environment variables for the current SSM session. This information can be useful in diagnosing Sammi problems.

The **View Sammi Errorlog** menu option opens a window which displays the Sammi error log file for the current SSM session. The error log display is difficult to interpret and usually not very informative, but it sometimes can be useful in diagnosing SSM or Sammi problems.

The **About Sammi** menu option opens a window which displays information about Sammi and about Kinesix, the Sammi developer. Among other items, it shows the current version of the Sammi

runtime, the host operating system and operating system version, and the hostname where Sammi is running. Again, this information can sometimes be useful in diagnosing problems.

The **SSM Consoles** menu option, under the **Monitor** heading on the SSM main menu, opens the SSM Console Information window. A "console" refers to one Sammi session which has connected to SSM. When first opened, the window shows information on the local (i.e., your) console. If other consoles are connected to the same SSM Data Server, you can use the "<<" and ">>" buttons to view information on other consoles.

Once a console has connected to an SSM Data Server, the Data Server continues to maintain information on that console in its internal console list, even after its SSM session has completely shut itself down. Seeing an inactive console in the list which is marked "Down" and "Logged Off" is perfectly normal. Console information is re-initialized only when the Data Server is shut down and restarted.

The window shows information such as console ID, connection state (Up or Down), last uptime and downtime, logon state, last logged-on user, and so forth (see the window help file for details). Sometimes, a Sammi session will crash, which can generate a lot of SSM error messages and cause problems for the remaining consoles. In such a situation, this window can be useful in tracking down which console may be causing the problem. Many times such console problems can be cleared up by restarting the crashed SSM session.

H.4 Customizing SSM and Sammi

This section lists a mixed bag of activities which can be performed to provide some site customization of SSM and Sammi. Some of these activities may be of interest to individual users; others only to the HPSS administrator.

- The file **/usr/lpp/sammi/data/timzones.dat** defines time zones and the start/end dates for daylight savings time in a particular year. It should be edited by the HPSS administrator at the beginning of each calendar year. It should also be edited immediately after HPSS is installed, to make sure that the installed file is up to date, and that it specifies the correct local time zone. The format of the file is explained by comments within the file.

- The **Print SSM Window** menu option, under the **Session** heading on the SSM main menu, allows you to print a hard copy of an SSM window on a PostScript printer. Sammi uses the environment variable **SAM2_SPOOL** as the command which sends the hard copy to the printer. The **/usr/lpp/hpss/bin/start_ssm_session** script defines **SAM2_SPOOL** as "**lpr**".

  With the "**lpr**" command, output will ordinarily go to the default printer on the host where Sammi is executing. Users can direct output to a different printer by defining the **PRINTER** environment variable before running **start_ssm_session**.

  The HPSS administrator may want to edit **start_ssm_session** to change the print command defined by **SAM2_SPOOL**, or to add options to the "**lpr**" command.

- The file /**usr/lpp/hpss/sammi/hpss_ssm/hpss.def** contains Sammi "defaults" settings for SSM. Sammi automatically loads **/usr/lpp/sammi/data/s2_defaults.def** when it starts, and **hpss.def** is used to override the settings in **s2_defaults.def**. Some of the user-preference features which can be set in a defaults file are:

1. Whether or not popup items on windows cause a beep tone.

2. The volume of the beep which is issued when the user tries to type beyond the end of a field.

3. The text entry foreground and background colors.

4. The blink rate for blinking fields.

Again, consult the comments for more information. For a complete list of default options, consult s2_defaults.def, or see the documentation on the set-default command in Chapter 4 of the Sammi Version 4.0 Runtime Reference.

The HPSS administrator may, if desired, modify **hpss.def** to change the site defaults. Alternatively, new defaults files may be created for individual users or groups of users. This would also involve the user authorization file and login command files (see below). The names of new defaults files must end in "**.def**", and must reside in **/usr/lpp/hpss/sammi/hpss_ssm** directory.

Any changes to defaults beyond the most innocuous user-preference items should be done with great care.

- Each user that logs into SSM has a Sammi command file executed automatically. The command file for each user is specified in **/usr/lpp/hpss/sammi/hpss_ssm/user_authorization.dat**, as the fifth field in each user entry. The default is **hpss.cmds**, which is in the **/usr/lpp/sammi/bin** directory. All command files are expected to be in that directory, but they can be placed elsewhere by specifying the full path name of the file in **user_authorization.dat**.

  The default command file loads the SSM "defaults" file (described above). If desired, the HPSS administrator can modify **hpss.cmds**, or create new command files for individual users or groups of users. A new command file might be created, for example, to load an alternate defaults file for a special group of users. Or a "**redefine-keys**" command might be added to load an alternate keyboard definition file for a particular user (see below).

  If a new command file is created, **user_authorization.dat** will need to be edited to assign the file to selected SSM users.

  Sammi command files must contain nothing but valid Sammi commands, as documented in the Sammi Version 4.0 Runtime Reference.

- The file **SAMMI** contains X defaults data important to the correct execution of Sammi. SSM users need a copy of this file in their home directories on all hosts where they will run Sammi (which may not be the same as the hosts where they display the windows). Normally, the **hpssuser** utility copies a **SAMMI** file to the home directory of each new SSM user which it creates, but that applies only to the host where **hpssuser** is executed. The template copy of **SAMMI** is in **/usr/lpp/hpss/config/templates/SAMMI.template.**

  The **SAMMI** file is customizable to some extent, and each SSM user is free to modify his own copy. These modifications should be kept to a minimum, however, or there may be severely adverse effects on the appearance of the SSM windows.

- The file **.motifbind** is read by the Motif window manager when it starts up, and is used to set key bindings to make Sammi work correctly. There are six versions of this file in **/usr/lpp//sammi/data**:

  .motifbind.dec

  .motifbind.hp

.motifbind.ibm

.motifbind.sgi

.motifbind.sun_mit

.motifbind.sun_news

In theory, all SSM users need **.motifbind** files in their home directories on the hosts where they work with SSM and view the windows (which may not be the same as the hosts where they are running Sammi). The Sammi documentation states that the proper file for the platform should be copied to each home directory and renamed to "**.motifbind**".

In practice, on AIX systems, the file is not required because the settings in **.motifbind.ibm** are the same as the defaults built into Motif. On other platforms, you may want to experiment to see if using one of these files solves any keyboard problems you may be having.

- The file **/usr/lpp/sammi/data/keydefs.dat** also affects Sammi keyboard behavior. This file is loaded into Sammi whenever it starts. By default, it contains keyboard data appropriate to the platform where Sammi is installed.

  A user who views SSM windows on a different platform may need to load a different **keydefs.dat** file. The alternatives in **/usr/lpp/sammi/data** are:

  keydefs.dat.ALPHA

  keydefs.dat.HP

  keydefs.dat.NIGHT

  keydefs.dat.RS6000

  keydefs.dat.SCO

  keydefs.dat.SGI

  keydefs.dat.SPARC_SOLARIS

  keydefs.dat.SUN

  Each user can load a keydefs.dat file by using the Set Keyboard option under the Session heading on the SSM main menu. This change lasts as long as the current SSM session. Alternatively, the HPSS administrator can permanently change the keydefs.dat file for a user by creating a Sammi login command file which includes a redefine-keys command specifying the appropriate keyboard file. Once the login command file is created, the administrator will have to edit the Sammi user authorization file to assign the login command file to the users who need it (see above).

- The SSM logon window, **hpss_Logon**, contains an HPSS logo graphic which uses 42 colors. This sometimes causes SSM start-up problems for workstations which have a limited number of available colors, or those with very slow network links (such as home computers running via PPP over a modem).

  There are two alternate logon windows which help with these problems. **hpss_LogonNoColor** has the same logo, but in only two colors (black and white). **hpss_LogonNoLogo** does not use the graphic logo at all.

To use one of the alternate windows, the user must edit /**usr/lpp/sammi/ssmuser/<user>/ssm_console.dat**, where <user> is the user's actual SSM username. Locate the string "**hpss_Logon**" and replace it with the name of the logon window to be used.

- Each user is allowed to set "preferences" to customize selected windows in SSM (see Chapter 6). These preferences may be saved in disk files which SSM can automatically load each time the user logs into an SSM session. Preferences are saved in disk files in each user's SSM work area, **/usr/lpp/hpss/sammi/ssm_user/<user>**, where <user> is an actual SSM username.

  While each SSM work area is owned by an SSM user, the SSM Data Server process is the entity which actually writes and reads the preferences files. This means that in most cases, the Data Server will be performing file operations in a directory which does not belong to it. This introduces permissions problems which can interfere with the user's ability to save and reload SSM preferences. The best way to avoid this problem is to have all SSM users in the same Unix group as the SSM processes, and to set group "rwx" permissions on the SSM work area directories.

H.5 Detailed Information on Setting Up an SSM User

The HPSS User Management Utility, **hpssuser**, must be run as **root** to set up each SSM user. Use the **-aix**, **-dce**, and **-ssm** flags to add a new user as an AIX user, a DCE principal, and an SSM user, respectively. Use whichever of these flags are appropriate for each new SSM user, but do not use a flag that is not needed. Do not, for example, use **-aix** if the user already has an AIX account on the SSM host. Use the **-h** flag to view a short summary of how to use **hpssuser**. Refer to the manual page for the **hpssuser** utility in Appendix I for more usage information.

Operations performed by **hpssuser** are summarized below:

1. If the **-dce** flag is specified, **hpssuser** creates a DCE principal entry and a DCE account entry for the new user in the DCE security registry. You will need to know the **cell_admin** password for your DCE cell in order to do this.

2. If the **-aix** flag is specified, **hpssuser** adds the new user as an AIX user.

3. If the **-ssm** flag is specified, you will be prompted for the hostname on which SSM will run and the user security level. The **hpssuser** utility will setup the SSM user as follows:

   - The Sammi API configuration file (**/usr/lpp/hpss/bin/api_config.dat**) is searched for existing "console ID" numbers. Each Sammi user requires a unique console ID number. After searching through the API configuration file, **hpssuser** creates a new console ID that is not yet in use.
   - Sun ONC RPC addresses are created for the three vital Sammi processes (**s2_evtsvr**, **s2_stream**, and **s2_mstalm**) by using the new console ID as part of each address. This results in a unique set of RPC addresses for each console ID.
   - Four new lines are added to the API configuration file. The first line is a comment line that includes the user's name. The last three lines begin with **logical_server**, and define the RPC address, RPC version number (equal to the console ID), and hostname (the one you entered at the prompt earlier) for each of the three vital Sammi processes. This information is read by the SSM Data Server process at start-

up, and is used by the Data Server to contact the Sammi processes being run by a specific SSM user.

- A new entry for the user is added to the Sammi User Authorization file **/usr/lpp/hpss/sammi/hpss_ssm/user_authorization.dat**.
- SSM work areas for the new user are created. These work areas are **/usr/lpp/hpss/sammi/ssmuser/<user>** (where **<user>** is the actual ID of the new user), and **/usr/lpp/hpss/sammi/ssmuser/<user>/mapfiles**.
- A template Sammi configuration file (**ssm_console.dat**) is copied to the new user's work area and modified to be user-specific. This involves editing the console ID number, process hostnames, and the RPC addresses and version numbers for the three vital Sammi processes.
- An X resources file named **SAMMI** is copied to the user's home directory.

H.6 Non-standard SSM Configurations

A number of SSM configurations are possible that do not get set up by the default HPSS installation or SSM user configuration. Using these configurations involves performing some procedures manually, instead of accepting the automated defaults. The information below can help in setting up these non-standard configurations.

You should not attempt to set up non-standard configurations unless you are familiar with how SSM works, and only when there is a real need to do so. These configurations are more difficult to set up and maintain, and most of them will have negative impacts on SSM performance.

Multiple Data Servers. The defaults assume only one SSM Data Server, running on the same host as the other SSM components. You may want multiple Data Servers, with some or all of them running on different hosts. For each SSM user with a non-default Data Server, you must edit the user's Sammi configuration file (**ssm_console.dat**, described above in Section H.2). Near the bottom of this file is a **logical_server** line for **ssm_ds**, which is the Data Server. Edit the RPC address and the hostname on this line as appropriate for the non-default Data Server. Note that all concurrently running Data Servers must have unique RPC addresses (although it is not mandatory that they run on different hosts). Make sure that any RPC addresses you select are not being used by any other processes on that host. Also note that multiple Data Servers must have unique CDS entries, as is true for any HPSS server.

If any of the above changes are made, the user must create a customized version of the **start_ssm** script which reflects the non-default RPC address for the Data Server and the non-default CDS names. If the Data Server and System Manager are to be executed on different hosts, invoke the **start_ssm** script with the **-s** option (to start the System Manager only) or **-d** option (to start up the Data Server only) on the appropriate host. If more than one Data Server is to be run on the same host, the logic in the script that disallows duplicate data servers must be removed.

Non-standard Sammi Runtime. The defaults assume that all SSM users are running Sammi on the same host where you executed the **hpssuser** script. If your site has multiple Sammi licenses, an SSM user may be running Sammi on a workstation remote from the main SSM processes. For any Sammi console ID not on the default host, you must edit the API configuration file (for that console) to correct the hostname for the three Sammi processes. You must also edit the user's Sammi configuration file (**ssm_console.dat**) to change the same three hostnames. You must then transfer the user's Sammi work area (see above) from the default host to the actual host. Finally, you will

likely have to provide the user a customized copy of the **start_ssm_session** script to deal with the non-standard Sammi runtime.

Multiple SSM Sessions. The defaults assume that each user will run only one SSM session at a time. If one user must run multiple SSM sessions, the easiest way to configure this is to create multiple user names for that user with **hpssuser**. If you choose not to do this, you must create completely separate Sammi execution environments for each concurrent session that a user may want to run. This involves separate Sammi work areas, hand-customized Sammi configuration files and API configuration file, and customized **start_ssm_session** scripts. Each concurrent session will have to use a different Sammi console ID and different sets of Sammi process RPC addresses.

H.7 SSM Security

SSM supports four types of users for security purposes: user, privileged, operator, and admin.

Security is applied both at the window level and the field level. A user must have permission to open a window to do anything with it at all. If the user does succeed in opening a window, all items on that window may be viewed. Field level security then determines if the user can modify fields, push buttons, and so on.

1. **user**. This security level is normally assigned to a user who may have a need to monitor some HPSS functions. This user may open the following windows, but cannot perform any SSM control functions:

   hpss_BitfileMO

   hpss_HealthStatus

   hpss_IdentifyBitfile

   hpss_IdentifyCartVol

   hpss_PvlVolumeMO

   hpss_PvrCartridgeMO

   hpss_Shutdown

2. **privileged**. This security level is normally assigned to a user such as an HPSS system analyst. This user may open and view almost all of the SSM windows, but cannot perform any SSM control functions. All SSM windows except for the following may be opened:

   hpss_ConfirmShutHPSS

   hpss_ConfirmShutSSM

   hpss_ConfirmShutSSMDS

   hpss_CreateDFSFileset

   hpss_CreateHPSSFileset

   HPSSCreateJunction

hpss_CreateSSRes

hpss_DeleteJunction

hpss_DeleteSSRes

hpss_ExportVolumes

hpss_ImportVolumes

hpss_MoveCartridge

hpss_ReclaimStart

hpss_RepackStart

3. **operator**. This security level is normally assigned to an HPSS operator. This user may open all SSM windows, and can perform all SSM control functions except for HPSS configuration.

4. **admin**. This security level is normally assigned to an HPSS administrator. This user may open all SSM windows and perform all control functions provided by SSM.

The file **/usr/lpp/hpss/sammi/hpss_ssm/user_authorization.dat** contains the assigned security level for each SSM user, the third field in each entry. The initial security level for a user is normally assigned when the SSM user is created by **hpssuser**.

The SSM security levels are defined in user_authorization.dat and security_class.dat, both in /usr/lpp/hpss/sammi/hpss_ssm. These files should be owned by root, and permit read-write access to root and read-only access to all other SSM users. Despite these protection measures, you should be aware that it is impossible to completely enforce the use of the SSM security levels. It is possible for any SSM user to get around the levels if so desired by taking advantage of weaknesses in Sammi security mechanisms. Do not rely on the SSM security levels to keep SSM users from maliciously accessing certain windows or fields.

Appendix I

HPSS Utility Manual Pages

This Appendix contains verbatim, unedited copies of the HPSS Man Pages that are in the HPSS **/usr/lpp/hpss/man** directory. The only changes are the addition of section headings.

archivecomp
archivedel
archivedump
archivelist
archiverec
auth_manager
backman
chacl
create_fset
create_fsys
crt_junction
del_junction
dump_sspvs
dumpbf
dumphpssfs
dumppv_pvl
dumppv_pvr
gen_reclaim_list
getdmattr
hdm_admin
hdmdump
hpss.clean
hpss_delog
hpssuser
insif
loadhpssdmid
loadhpssfs
loadhpssid
loadtree
lsacl
lsfilesets
lshpss
lsjunctions
lsrb
lsvol
managesfs
metadata_info
mps_reporter
multinoded
nfsmap
nsde
pftp_client
pftpd
plu
reclaim
reclaim.ksh
recover
remove
repack
retire
scrub
setdmattr
settapestats
sfsbackup
shelf_tape

I.1 archivecmp — Determine filesets inconsistencies

```
NAME
     archivecmp - Checked for archived fileset inconsistencies.

SYNTAX
     archivecmp <Sorted archivedump output> <[Sorted archivelist output>

DESCRIPTION
     archivecmp will compare sorted output from the utilities
     archivedump and archivelist and attempt to find any inconsistencies
     between DFS and HPSS archived filesets.

     In general, this utility will be used in conjunction with the
     utilities, archivedump and archivelist to check for archive fileset
     consistency between DFS and HPSS. The output from archivecmp
     will describe an inconsistencies and will make suggestions on
     how best to repair the inconsistencies.

     See “HPSS Administration Guide: DFS Configuration and Management”
     for more details.

PARAMETERS

     Sorted archivedump output - Output from the utility archivedump that
                                 has been sorted by the UNIX “sort”
                                 command to order the output by HPSS file
                                 name.

     Sorted archivelist output - Output from the utility archivelist that
                                 has been sorted by the UNIX “sort”
                                 command to order the output by HPSS file
                                 name.

EXAMPLE
     The following command will compare output from archivelist and
     archivedump, where the output from each utility has been sorted
     and placed in the files “archdump.sort.out” and “archlist.sort.out”.

     archivecmp archdump.sort.out archlist.sort.out

FILES


SEE ALSO
     archivedump(7), archivelist(7), setdmattr(7), getdmattr(7)

SCCS
     man/archivecmp.7, gen, 4va 11/20/98 15:19:16
```

I.2 archivedel — Delete old files from an HPSS archived fileset

```
NAME
     archivedel - Delete old files from an HPSS archive filese

SYNTAX
     archivelist [Fileset Name] [Access age in days]

DESCRIPTION
     archivedel must be used periodically to remove old unused files from
     an HPSS archived fileset that is using the archive/rename option.
     Files in filesets of this type do not automatically remove HPSS
     files when the DFS files are deleted, the files are simply renamed
     with a “DEL.” prefix. This allows for recovery of these files in
     case of DFS fileset loss.

     Since files are renamed it may become necessary to periodically
     remove old “DEL.” files to free up HPSS resources. Only “DEL.” files
     that are older than the last full backup should be removed, or it
     may be impossible to recover from a DFS disk failure.

     See “HPSS Administration Guide: DFS Configuration and Management”
     for more details.

PARAMETERS

     Fileset Name       - The HPSS name of the archived fileset.

     Access age in days - Remove any “DEL.” files from the archived
                          fileset that have not been accessed for this
                          many days.

EXAMPLE
     The following command will remove all “DEL.” files from the
     archived fileset test.fileset that have not been accessed for
     10 days.

     archivedel test.fileset 10

FILES


SEE ALSO
     archiverec(7)

SCCS
     man/archivedel.7, gen, 4va 11/20/98 15:19:44
```

I.3 archivedump — List all objects in a DFS archived fileset

```
NAME
     archivedump - List out all the objects in a DFS archived fileset.

SYNTAX
     archivedump <Fileset ID> <Fileset Name> <Fileset Local Mount> <Min Size>

DESCRIPTION
     archivedump must be run on the machine that holds a DFS archived
     fileset and will dump to stdout a list of files that reside on
     the fileset. archivedump must be run as root. For each file in
     the fileset, a line of output will be sent to stdout in the
     following format:

        [hpss file name] [file age] [file state] [Episode path]

     hpss file name - The name of the file stored in HPSS
     file age       - time since last file access
     file state     - HPSS_BACKED_DATA_VALID, file has been backed into
                        HPSS and the data on HPSS is in sync with the
                        DFS data.

                      HPSS_BACKED_DATA_INVALID, file stored in HPSS in
                        the past, but file modified on DFS after being
                        backed into HPSS. This state should change after
                        the next migrate cycle.

                      NOT_YET_MIGRATED, DFS file never migrated into HPSS.
                        If access time is old then migration may have a
                        problem. This state should change during the
                        next migrate cycle.

                      ERROR_NO_DMATTRS, DFS file is in a bad state. The
                        utility “setdmattr” should be used to set the
                        MIGRATE DM attribute for the file to a value of
                        1.
     Episode Path   - Path to the file relative to the root of the
                        fileset.

     In general, this utility will be used in conjunction with the
     utilities, archivelist and archivecmp to check for archive fileset
     consistency between DFS and HPSS.

     See “HPSS Administration Guide: DFS Configuration and Management”
     for more details.

PARAMETERS

     Fileset ID          - Numeric fileset identification.
                           (e.g 0.4, 0.234, etc.)
```

```
     Fileset Name        - Character name for the fileset.

     Fileset Local Mount - Path to where the fileset is mounted on the
                           local UNIX file system.

     Min Size            - Do not generate output for any file less than this
                           size. Since archived filesets now have options that
                           stop the migration of small files into HPSS, these
                           files should not generate any output when using this
                           tool to check archived fileset consistency.


EXAMPLE
     The following command will dump output to the file “archdump.out”
     for fileset 0.4 whose name is test.fileset and which is currently
     locally mounted at
     “/var/hpss/hdm/hdm1/aggr/test.aggr/test.fileset”. It will only generate
     output for files whose size is less than 65536.

     archivedump 0.4 test.fileset \
        /var/hpss/hdm/hdm1/aggr/test.aggr/test.fileset 65536 > archdump.out

FILES


SEE ALSO
     archivecmp(7), archivelist(7), setdmattr(7), getdmattr(7)

SCCS
     man/archivedump.7, gen, 4.1.1, 4.1.1.1 5/13/99 17:26:53
```

I.4 archivelist — List all objects in an HPSS archived fileset

```
NAME
     archivelist - List out all the objects in an HPSS archived fileset.

SYNTAX
     archivelist <Fileset Name>

DESCRIPTION
     archivelist prints out a list to stdout of all the files that
     reside in an HPSS archived fileset. The tool should be run on a
     machine that can access the Name Server. The client should
     dce_login as principal “hpss_dmg”.

     This tool will generally be used in conjunction with other tools
     (archivedump and archivecmp) to check archived fileset consistency.

     This tool will generate a line of data for each file in an HPSS
     archived fileset. If redirecting the output to a file be sure there
     is enough space for all the output. The amount of space required will
     depend on the number of entries in the archived fileset.

     See “HPSS Administration Guide: DFS Configuration and Management”
     for more details.

PARAMETERS

     Fileset Name - The HPSS name of the archived fileset.

EXAMPLE
     The following command will list all the files in the archived
     fileset “test.fileset” and put the output in the file “arch.out”.

     archivelist test.fileset > arch.out

FILES


SEE ALSO
     archivedump(7), archivecmp(7), setdmattr(7), getdmattr(7)

SCCS
     man/archivelist.7, gen, 4va 11/20/98 15:20:41
```

I.5 archiverec — Recover files from an HPSS archived fileset

```
NAME
     archiverec - Recover any files from an HPSS archive fileset.

SYNTAX
     archiverec <Fileset Name> <Access age in days>

DESCRIPTION
     archiverec must be used to recover any recently deleted files for
     an HPSS archived fileset that is using the archive/rename option.
     Files in filesets of this type do not automatically remove HPSS
     files when the DFS file is deleted, the files are simply renamed
     with a “DEL.” prefix. This allows for recovery of these files in
     case of DFS fileset loss.

     If a DFS fileset has to be restored because of a disk loss,
     archiverec must be run to rename any recently deleted files so the
     DFS can retrieve data from these files.

     See “HPSS Administration Guide: DFS Configuration and Management”
     for more details.

PARAMETERS

     Fileset Name       - The HPSS name of the archived fileset.

     Access age in days - Recover any “DEL.” files from the archived
                          fileset that have been deleted within the
                          time frame. The amount of time must go beyond
                          the last fill “fts dump” for the fileset on
                          DFS.

EXAMPLE
     The following command will recover all “DEL.” files from the
     archived fileset test.fileset that have been been deleted in
     the last 10 days.

     archiverec test.fileset 10

FILES


SEE ALSO
     archivedel(7)

SCCS
     man/archiverec.7, gen, 4va 11/20/98 15:21:01
```

I.6 auth_manager — Authentication mechanisms for the HPSS PFTP server process

NAME

```
auth_generic           - Generic Authentication Module
                                (NOT intended for use).
auth_dcegss            - DCE GSS Authentication Module.
auth_krb5gss           - Kerberos GSS Authentication Module.
auth_dcemitgss_compat  - DCE and MIT V5B6 Cmpatable Authentication Module.
auth_ident             - IDENT Authentication Module.
auth_krb5_and_ident    - Kerberos GSS and/or IDENT Authentication Module.
```

The appropriate Authentication Module is executed by the hpss_pftpd_amgr initiated with the -G flag and obtaining the name of the Authentication module from the “hpss_option AMGR path/name” record in the ftpaccess file.

PURPOSE

The Authentication Modules are mechanisms for implementing “password-less” FTP services. The appropriate module is executed by the hpss_pftpd_amgr and uses shared memory to communicate between the hpss_pftpd_amgr and the Authentication module.

SYNTAX

Note: The Authentication module is started by the hpss_pftpd_amgr daemon.

DESCRIPTION

The Authentication Module provides alternate authentication mechanisms for the HPSS PFTP server process.

The Kerberos GSS-based (Generic Security Service) versions of the Authentication modules support Cross Cell Authentication.

FLAGS

FILES

RELATED INFORMATION

pftp_client(7), multinoded(7), pftpd(7), syslogd.

HPSS System Administration Guide

SCCS

```
        @(#)94   1.2   man/auth_manager.7, gen, 4va   12/2/98   08:52:36
```

I.7 backman — HPSS Backup Manager

```
NAME
     backman - HPSS backup manager

SYNTAX
     backman

DESCRIPTION
     This menu-based utility helps HPSS administrators implement important
     backup procedures on AIX platforms.  Mechanisms are available to:

        - Log into DCE
        - Backup Encina log archive files
        - Backup Encina data volumes
        - Backup DCE Security and CDS servers
        - Backup Encina Server configuration data
        - Backup HPSS PFTP/NFS daemon configuration data
        - Copy existing tar file to tape
        - Query Encina SFS backup status
        - Retain Encina SFS backups
        - Review backup log for this node

     IMPORTANT NOTE:  When specifying the tar backup device, the tape must
                      already be positioned for writing (i.e., it must be at
                      the end of the tape).  If you plan on leaving the tape
                      in the drive and/or backing up multiple items to the
                      tape, you MUST specify the proper device name that will
                      not cause the tape to automatically rewind after each
                      operation.  Normally, this is the .1 file associated
                      with the device (e.g., for device /dev/rmt0, specifying
                      /dev/rmt0.1 will not rewind the device after each
                      operation).  If, for example, you back up to /dev/rmt0,
                      each backup operation will write to the beginning of
                      tape and all previous data written to the tape will be
                      lost!

     Individual menu options are discussed below.

     1 - Log into DCE

     In order to perform certain operations (namely creating Encina and DCE
     backups as well as querying and retaining Encina backups), proper DCE
     credentials are required.  DCE backups typically require “cell_admin”
     credentials while Encina operations typically require “encina_admin”
     credentials.  DCE credentials acquired during the backman session are
     destroyed when you exit from the tool.

     2 - Backup Encina log archive files

     If Encina SFS media archiving is enabled, log archive (or LA) files are
     generated automatically, usually in
```

```
     /opt/encinalocal/encina/sfs/hpss/archives

(for SFS server "/.:/encina/sfs/hpss") for Encina 2.1 or in

     /opt/encinalocal/encina/sfs/server/hpss/logArchive

for Encina 2.2.  These files must routinely be moved to offline storage
(tape).  This option will tell you how many LA files exist on disk and
allow you to tar the files to a device.  Subsequently, the tool will allow
you to delete the LA files from disk.

3 - Backup Encina data volumes

This option generates backup files for one or all SFS data volumes.  It is
simply a front-end to the HPSS "sfsbackup" utility.  You will be prompted
for the SFS volume to be backed up (the default is "all" volumes, which
is recommended).  It also prompts for the number and size of backup
files.  Note that one(1) kilobyte of space must be added to the size
parameter for overhead.  For example, if your SFS volume was 2 GB and you
wanted 64 backup files to constitute a complete backup, then each backup
file must be 32769k, which is ( 2 GB / 1024 / 64 ) + 1.  Note that the
size parameter should be given in "kilobytes" with a trailing "k" on the
value.

The backup files are stored, by default, in
/archive/sfs_backups/<volname>.  A separate directory <volname> is
created for each SFS volume.  Once the files have been generated, the
tool allows you to tar them to a device.

Note that if you have multiple SFS servers defined on the same node,
the archive directory must be unique for each server.  The archive
directory must also be writable by the SFS process (for Encina 2.2,
Transaction Server for AIX, the directory should be owned by user
"encina").

4 - Backup DCE Security and CDS servers

Since disks used by DCE can become corrupted, it is important to backup
certain DCE files critical to recovering DCE.  This option will tar
together the following local DCE directories:

   /opt/dcelocal/var
   /opt/dcelocal/etc
   /krb5

This option should be executed on the node where the Security Server
and/or CDS is configured.  Prior to the backup, the Security Server is
placed in "maintenance" mode while CDS is stopped.  After the backup, the
Security Server is restored to "service" mode and CDS is restarted.
Since this backup operation completes very quickly, this activity should
not adversely affect executing DCE programs.  Refer to the current DCE
documentation for other potential side affects as well as what other
```

files may need to be backed up.

5 - Backup Encina Server configuration data

This option allow the Encina configuration and restart files to be backed up. The following files are included:

For Encina 2.1 for AIX:
/opt/encinalocal/encina/sfs/*/restart*
/opt/encinalocal/encina/sfs/*/keyfile
/var/encina/

For Encina 2.2 (Transaction Server) for AIX:
/opt/encinalocal/encina/node
/opt/encinalocal/encina/ecm
/opt/encinalocal/encina/*/*/restart*
/opt/encinalocal/encina/*/*/keyfile

This option should be run on each node where an Encina SFS server is defined.

6 - Backup HPSS PFTP/NFS daemon configuration data

This option will generate a tar file of existing HPSS Parallel FTP (PFTP) and NFS daemon configuration information. By default, these files and directories are backed up:

PFTP Daemon Directory
/var/hpss/ftp/etc

NFS Daemon Files
/var/hpss/nfs/exports

This option should be run on each node containing either a PFTP daemon or NFS daemon configuration.

7 - Copy existing tar file to tape

Since options 4-6 may generate tar files that must be transferred to a machine with a tape device, this option allows an existing tar file to be dumped to a device.

8 - Query Encina SFS backup status

This option displays backup status information for each SFS volume.

9 - Retain Encina SFS backups

Since SFS maintains some amount of information on generated backup files, older backups should be deleted from time to time in order to allow SFS to free up those resources. After numerous, complete backups have been generated for your SFS data volumes, this option can be used to instruct Encina to keep information only on the latest X number of complete

backups (e.g., 5). This option should be performed at least once per week.

r - Review backup log for this node

This option will display the contents of the local backup log file on this node (default is /var/hpss/backup.log). The user is prompted to enter the number of lines to view at the bottom of the file, since the file may become large over time. Entering a value of “all” (the default) will show the entire contents of the file.

EXAMPLE

The following session shows how all LA files are backed up to tape and 8 backup files are generated for each SFS data volume and moved to tape.

```
H P S S   B a c k u p   M a n a g e r

Today’s Specials:

   1 - Log into DCE
   2 - Backup Encina log archive files
   3 - Backup Encina data volumes
   4 - Backup DCE Security and CDS servers
   5 - Backup Encina Server configuration data
   6 - Backup HPSS PFTP/NFS daemon configuration data
   7 - Copy existing tar file to tape
   8 - Query Encina SFS backup status
   9 - Retain Encina SFS backups
   r - Review backup log for this node
   q - Quit

Your choice: 1
Current DCE principal is “/.../hpss.mhpcc.edu/hosts/popolana.mhpcc.edu/self”
Do you want to log in as a different DCE principal? [y]:

Enter DCE principal [encina_admin]:
Enter Password:

Today’s Specials:

   1 - Log into DCE
   2 - Backup Encina log archive files
   3 - Backup Encina data volumes
   4 - Backup DCE Security and CDS servers
   5 - Backup Encina Server configuration data
   6 - Backup HPSS PFTP/NFS daemon configuration data
   7 - Copy existing tar file to tape
   8 - Query Encina SFS backup status
   9 - Retain Encina SFS backups
   r - Review backup log for this node
   q - Quit
```

```
Your choice: 2

A total of 25 log archive files are available to move to offline storage
Oldest LA file is /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2346
Latest LA file is /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2370

Would you like to tar these to some device or file? [y]:
Enter backup device/file [/dev/rmt0.1]:
The backup device should already be positioned for writing.
Ok to begin writing at current position? [n]: y
Enter label for the tape you are about to write to: HPSS0002

Creating tar backup to "/dev/rmt0.1"...
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2346 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2347 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2348 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2349 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2350 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2351 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2352 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2353 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2354 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2355 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2356 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2357 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2358 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2359 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2360 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2361 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2362 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2363 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2364 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2365 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2366 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2367 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2368 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2369 2048 blocks.
a /opt/encinalocal/encina/sfs/hpss/archives/logVol1.LA.2370 2048 blocks.

Of 25 log archive files, how many would you like to keep on disk? [25]: 0

First file to be deleted will be /opt/encinalocal/encina/sfs/hpss/archives/
logVol1.LA.2346
 Last file to be deleted will be /opt/encinalocal/encina/sfs/hpss/archives/
logVol1.LA.2370

Ok to delete these 25 log archive files? [n]: y
Deleting files...done

Today's Specials:

   1 - Log into DCE
   2 - Backup Encina log archive files
```

```
   3 - Backup Encina data volumes
   4 - Backup DCE Security and CDS servers
   5 - Backup Encina Server configuration data
   6 - Backup HPSS PFTP/NFS daemon configuration data
   7 - Copy existing tar file to tape
   8 - Query Encina SFS backup status
   9 - Retain Encina SFS backups
   r - Review backup log for this node
   q - Quit

Your choice: 3

Enter backup root directory [/archive/sfs_backups]:
Do you want to generate SFS data volume backup files? [y]:
Enter number of backup files to generate per volume [8]:
Enter data volume to back up [all]:
Enter size of each backup file (use k for kb) [32769k]:

You are about to generate 8 backup files for all volumes
on server "/.:/encina/sfs/hpss" into "/archive/sfs_backups"

Ok to proceed? [n]: y

Backing up sfsVol1 from /.:/encina/sfs/hpss at Wed Jul 10 15:21:47 HST 1996
Backing up sfsVol2 from /.:/encina/sfs/hpss at Wed Jul 10 15:23:52 HST 1996

Would you like to tar these to some device or file? [y]: y
Enter backup device/file [/dev/rmt0.1]:
The backup device should already be positioned for writing.
Ok to begin writing at current position? [n]: y
Enter label for the tape you are about to write to [HPSS0002]:

Creating tar backup to "/dev/rmt0.1"...
a /archive/sfs_backups/sfsVol1/sfsVol1.TRB.000144 65537 blocks.
a /archive/sfs_backups/sfsVol1/sfsVol1.TRB.000145 65537 blocks.
a /archive/sfs_backups/sfsVol1/sfsVol1.TRB.000146 65537 blocks.
a /archive/sfs_backups/sfsVol1/sfsVol1.TRB.000147 65537 blocks.
a /archive/sfs_backups/sfsVol1/sfsVol1.TRB.000148 65537 blocks.
a /archive/sfs_backups/sfsVol1/sfsVol1.TRB.000149 65537 blocks.
a /archive/sfs_backups/sfsVol1/sfsVol1.TRB.000150 65537 blocks.
a /archive/sfs_backups/sfsVol1/sfsVol1.TRB.000151 65537 blocks.
a /archive/sfs_backups/sfsVol2/sfsVol2.TRB.000144 65537 blocks.
a /archive/sfs_backups/sfsVol2/sfsVol2.TRB.000145 65537 blocks.
a /archive/sfs_backups/sfsVol2/sfsVol2.TRB.000146 65537 blocks.
a /archive/sfs_backups/sfsVol2/sfsVol2.TRB.000147 65537 blocks.
a /archive/sfs_backups/sfsVol2/sfsVol2.TRB.000148 65537 blocks.
a /archive/sfs_backups/sfsVol2/sfsVol2.TRB.000149 65537 blocks.
a /archive/sfs_backups/sfsVol2/sfsVol2.TRB.000150 65537 blocks.
a /archive/sfs_backups/sfsVol2/sfsVol2.TRB.000151 65537 blocks.

Would you like to delete these backup files on disk? [n]: y

WARNING:  Do not delete these files unless they were successfully
```

```
          moved to offline storage!

Ok to delete the backup files? [n]: y

Deleting files...done

Today’s Specials:

   1 - Log into DCE
   2 - Backup Encina log archive files
   3 - Backup Encina data volumes
   4 - Backup DCE Security and CDS servers
   5 - Backup Encina Server configuration data
   6 - Backup HPSS PFTP/NFS daemon configuration data
   7 - Copy existing tar file to tape
   8 - Query Encina SFS backup status
   9 - Retain Encina SFS backups
   r - Review backup log for this node
   q - Quit

Your choice: r

How many lines at the end of the log to review [all]: 10

Press ENTER to view “/var/hpss/backup.log”:

07/08/96:15:42:11 HPSS0001 TRB backup sfsVol1 000096-000103, sfsVol2 000096-
000103
07/08/96:16:10:36 HPSS0001 TRB backup sfsVol1 000104-000111, sfsVol2 000104-
000111
07/08/96:16:40:31 HPSS0002 TRB backup sfsVol1 000112-000119, sfsVol2 000112-
000119
07/08/96:17:09:55 HPSS0002 TRB backup sfsVol1 000120-000127, sfsVol2 000120-
000127
07/09/96:14:30:10 HPSS0002 LA backup logVol1.LA.1952-logVol1.LA.2231
07/09/96:16:19:07 HPSS0002 TRB backup sfsVol1 000128-000135, sfsVol2 000128-
000135
07/09/96:17:10:39 HPSS0002 TRB backup sfsVol1 000136-000143, sfsVol2 000136-
000143
07/10/96:11:37:12 HPSS0002 LA backup logVol1.LA.2232-logVol1.LA.2345
07/10/96:15:22:49 HPSS0002 LA backup logVol1.LA.2346-logVol1.LA.2370
07/10/96:15:39:20 HPSS0002 TRB backup sfsVol1 000144-000151, sfsVol2 000144-
000151

Today’s Specials:

   1 - Log into DCE
   2 - Backup Encina log archive files
   3 - Backup Encina data volumes
   4 - Backup DCE Security and CDS servers
   5 - Backup Encina Server configuration data
   6 - Backup HPSS PFTP/NFS daemon configuration data
   7 - Copy existing tar file to tape
```

```
   8 - Query Encina SFS backup status
   9 - Retain Encina SFS backups
   r - Review backup log for this node
   q - Quit

Your choice: q

Acquired DCE credentials have been destroyed


FILES
     /var/hpss/backup.log

          Contains a log of backup operations performed by backman.

     /opt/encinalocal/encina/*/<sfsServer>/archives

          Encina directory for log archive files

     /archive/sfs_backups

          Default location for SFS data volume backup directories and files


SEE ALSO
     sfsbackup
     tar
     dd

SCCS
     man/backman.7, gen, 4va   8/6/97   09:21:41
```

I.8 chacl — HPSS Change ACL Utility

```
NAME
     chacl - HPSS Change ACL Utility

SYNTAX
     chacl [ -r -u -b ] < FILE

DESCRIPTION
     "chacl" is a HPSS User Utility.  This utility is used to change
     discretionary access control information associated with a
     specified object. The output from lsacl is in the correct
     format for input to chacl.  The input to chacl for an ACL consists
     of two sections: the comments section and the ACL entries section.
     The comments section contains three lines: name of the object,
     the object owner, and the object owning group.  The ACL entries
     section contains lines with three fields seperated by a colon.
     The first field contains the ACL entry tag type.  The second
     field contains the ACL entry qualifier.  The third field contains
     the access permissions.

  Output Format:

     #file:<filename>
     #owner:<uid>
     #group:<gid>
     user::<perm>
     mask::<perm>
     user:<uid>:<perm>
     user:<uid2>:<perm>
     group::<perm>
     group:<gid>:<perm>
     group:<gid2>:<perm>
     other::<perm>

The types of permissions are:
        r - read
        w - write
        x - execute
        i - insert
        d - delete
        c - control

        The utility chacl is initiated from the command line.
        The utility differs from some other HPSS utilities in
        that it uses the client api library.  The user must
        have a dce_login and two environment variables must
        be set.  The environment variables for the CDS names
        of the NameServer and the Bitfile Server are:
        HPSS_HPNS_NAME and HPSS_BFS_NAME.
```

```
OPTIONS
     -r     Remove the specified ACL entries. Base
            entries are not affected.

     -u     Update the specified ACL entries.  The
            entries are added if they do not already
            exist.

     -b     Remove all entries except the three base
            entries, user::, group::, and other::.

     FILE   The file containing the object to change.

EXAMPLE
     The following steps are used to add the user with id 7004
     with read permission to the ACL list of the file
     /ampex/test.txt.

     dce_login hpss_client1

     export HPSS_LS_NAME=/.:/hpss/ls

     lsacl -f /ampex/test.txt > acl_format

     cat acl_format
     #file: /ampex/test.txt
     #owner:1006
     #group:system
     user::rwx---
     mask::r-x---
     group::r-x---

     edit acl_format { add to the list “user:7004:r-----” }

     cat acl_format
     #file: /ampex/test.txt
     #owner:1006
     #group:system
     user::rwx---
     user:7004:r------
     mask::r-x---
     group::r-x---

     chacl -u < acl_format


FILES


SEE ALSO
     lsacl(7)
```

```
SCCS
     man/chacl.7, gen, 4va   11/23/98   14:06:35
```

I.9 create_fset — Create the HPSS counterpart of a DFS Fileset

NAME

create_fset - create the HPSS counterpart of a DFS fileset

SYNTAX

```
create_fset dmg_name dmg_host dmg_port hdm_host hdm_port \
    ftid fsid fsname global local family [cos]
```

PARAMETERS

Name	Use	Examples
dmg_name	CDS name for DMAP gateway	/.:/hpss/dmg
dmg_host	Host name for DMAP gateway	htibm2.sandia.gov
dmg_port	Port for DMAP gateway	6002
hdm_host	Host name for the HDM	tardis.sandia.gov
hdm_port	Port for the HDM	7001
ftid	Fileset id	0.361 or 0,,361
fsid	File system id	4
fsname	File system “name”	/dev/xxx
global	Global mount point	/:/hpss/yyy
local	Local mount point	/Fsets/hpss.yyy
family	Family ID	0
cos	Class of service	1 [default 0]

DESCRIPTION

This command creates the HPSS counterpart of a DFS fileset. It also ensures that the HDM, DMAP Gateway, and Nameserver exchange the information about the fileset necessary to coordinate their activities. Although a fileset can be created on DFS without using this utility, HPSS users will not be able to access the fileset until this utility has been run.

The gateway and DMAP hostnames should be specified as fully qualified names -- for example, ‘tardis.sandia.gov’ rather than just ‘tardis’. Be sure to make the corresponding entry in gateways.dat before running create_fset.

The fileset ID can be specified either in the form <hi>.<low> or <hi>,,<low>, where <hi> and <low> are integers. The values must agree with the DFS fileset ID that was returned when the fileset was created using the ‘fts create’ command.

Note that the file system “name” (fsname) must be in the form /dev/<xxx>, where xxx is the physical device or “aggregate” on which the file system resides.

```
     The file system id (fsid) must agree with the actual file
     system id as specified in dfstab.

     Be sure to run create_fsys before running create_fset,
     and be sure to use the same file system IDs in both
     programs.

     For archived aggregates, the global and local mount
     points are typically specified as “NO_MOUNT_POINT”.

     Be sure to choose family and cos values that have already
     been configured on HPSS.

  Environment Variables

     HPSS_PRINCIPAL_SSM, if set, determines the DCE principal
     name to use for securing communications with the DMAP
     Gateway.  The default is hpss_ssm.

     HPSS_KEYTAB_FILE_SERVER, if set, determines the name of
     the keytab file to be used to the DCE principal’s key.
     The default is /krb5/hpss.keytabs.

     Use the hpss_env script to set site-dependent environment
     variables.

EXAMPLE
     create_fset /.:/hpss/dmg htibm2.ca.sandia.gov 6002
         tardis.ca.sandia.gov 7001 0,,361 4
         /dev/tardis_mirror1 /:/hpss/mirror1
         /var/hpss/hdm/hdm1/tardis_mirror1/hpss.mirror1 0 1


FILES
     filesys.dat     file systems and filesets managed by the HDM
     gateways.dat    hosts permitted to run create_fset


SEE ALSO
     create_fsys     create a file system entry in filesys.dat
     crtjunction     create an HPSS junction


SCCS
     man/create_fset.7, gen, 4va   11/20/98   17:20:41
```

I.10 create_fsys — Define a file system to HPSS/DMAP

```
NAME
     create_fsys - define a DFS file system to the HDM

SYNTAX
     create_fsys host port fsid path media option migrate purge stage key

PARAMETERS
     host        Host name where the HDM and DFS are running.
     port        Port on that host.
     fsid        File system id for the file system.
                 Must agree with file system id in dfstab;
     path        Mount path for aggregate.
                 e.g., /opt/dcelocal/var/dfs/aggrs/tardis_disk1
     media       Device where file system resides;
                 e.g., /dev/dsk/c0t2d0s4
     option      “archive/rename”, “archive/delete” or “mirrored”
     migrate     Name of migrate policy for this file system.
     purge       Name of purge policy for this file system.
     stage       How files are staged: “whole” or “partial”.
     key         Encryption key, expressed as a hex number.

DESCRIPTION
     This command defines a DFS file system to the HDM server.
     Until this command has been executed, the HDM will not be
     able to manage files on this file system.  The utility can
     be executed on any host that is listed in the gateways
     configuration file gateways.dat.

     The option parameter determines how the DFS name and data
     spaces will be managed.  If the mirrored option is used,
     files will be accessible from both DFS and HPSS.  If one of
     the archive options is chosen, DFS files will be archived on
     HPSS but will not be accessible from HPSS.  The rename and
     delete suboptions control how archived files are handled
     when they are deleted.  If the delete suboption is used, the
     files are deleted immediately.  If the rename suboption is
     used, the files are not actually deleted, but rather are
     renamed.  This allows an administrator to restore files that
     were accidentally deleted.

     The migrate and purge parameters must refer to policies that
     are already defined in purge.dat.

     The stage parameter determines how files are staged from
     HPSS to DFS.  If “whole” is selected, whole files will be
     staged.  Otherwise, only that part of the file needed to
     complete an operation will be staged.

     The key parameter defines an encryption key that is used
     to authorize the user.  It is typically a 16-digit number.
```

```
     The key must agree with the value specified in gateways.dat
     for the host where create_fsys is running.

EXAMPLE
     The following example shows how a file system is defined.
     The HDM that manages the file system is assumed to run on a
     host named tardis.ca.sandia.gov.  The HDM is using the
     standard port 6002 to listen for requests.  The file system
     id is 2.  The file system will be mirrored on HPSS.  Files
     will be migrated using a migration policy named “run”.  The
     mount paths are as shown in the example.  Files will be
     purged using a purge policy named “wait”.  These names do
     not have any particular significance to the HDM, but they
     must be defined in policy.dat.  Whole-file staging will be
     used.  The encryption key is 123abc.

     create_fsys tardis.ca.sandia.gov 6002 2
         /opt/dcelocal/var/dfs/aggrs/tardis_aggr1
         /dev/tardis_aggr1 mirrored run wait whole 123abc

FILES
     filesys.dat      file systems that the HDM manages
     gateways.dat     hosts permitted to run create_fsys
     policy.dat       migration and purge policy definitions

SCCS
     @(#)32   1.3   man/create_fsys.7, gen, 4va   2/23/99   22:45:25
```

I.11 crtjunction — Create a junction to a fileset

```
NAME
     crtjunction - Create a junction to a file set.

SYNTAX
     crtjunction <Fileset Name> <Path to Junction>

DESCRIPTION
     crtjunction can be used to create junctions to the root of filesets.
     crtjunction only allows creation of junctions within HPSS filesets,
     junctions within archived or mirrored filesets will have to be made
     through the DFS command “fts create”.

     crtjunction can used used to create junctions to any type of fileset.


PARAMETERS

     Fileset Name - The name of the fileset to which the junction will
                    point.

     Path to Junction - A full path in the HPSS name space where the
                        junction is to be created.
EXAMPLE
     The following command will create a junction at “/:/fileset.0” that
     points to the root of the fileset “fileset.0”.

     crtjunction fileset.0 /:/fileset.0

FILES


SEE ALSO
     deljunction(7)

SCCS
     man/crtjunction.7, gen, 4va 11/20/98 15:21:23
```

I.12 deljunction — Delete a junction to a fileset

```
NAME
     deljunction - Delete a junction to a file set.

SYNTAX
     deljunction <Path to Junction>

DESCRIPTION
     deljunction can be used to delete junctions to the root of filesets.
     deljunction will only allow deletions of junctions that reside in an
     HPSS-only fileset. If the junction resides in an archived or
     mirrored fileset then the junction will have to be deleted through
     DFS by using the “fts delmount” command.


PARAMETERS

     Path to Junction - A full path in the HPSS name space of the
                        junction to be deleted.

EXAMPLE
     The following command will delete a junction at (/:/fileset.0).

     deljunction /:/fileset.0

FILES


SEE ALSO
     crtjunction(7)

SCCS
     man/deljunction.7, gen, 4va 11/20/98 15:21:47
```

I.13 dump_sspvs — List Physical Volumes Managed by a Storage Server

```
NAME
     dump_sspvs - List physical volumes managed by a Storage Server.

SYNTAX
     dump_sspvs [ -p Utility Principal Name
                  -k Utility KeyTab Pathname
                  -g HPSS Generic Config Name
                  -d SS Descriptive Name
                  -s Storage Class(optional)
                  -f Family ID(optional)
                  -l (optional)]

DESCRIPTION
     This utility will scan the physical volumes managed by the
     specified Storage Server and output the physical volume
     name, storage map state, family id, storage class and active
     data and volume length on the volume.  A summary of available
     and used bytes by storage class is also output.  A specific
     storage class, family id or locked arguement may optionally
     be selected.




OPTIONS

     -d <SS Descriptive Name>
                          The descriptive name of the HPSS Storage Server
                          which manages the Volumes to be scanned.
                          The descriptive name of a storage server
                          can be found from SSM through: Admin, Configure
                          HPSS, Server Configuration screens.

     -g <HPSS Generic Config Name>
                          The pathname of the SFS generic server
                          configuration file.

     -p <Utility Principal Name>
                          The DCE principal name which this utility will run
                          under.  Usually run as SSM.  The Principal name of
                          the SSM can be found from SSM through: Admin,
                          Configure HPSS, Server Configuration, Security
                          Information screens.

     -k <Utility KeyTab Pathname>
                          The Keytab name which this utility will run with.
                          Usually run as SSM.  The Keytab name of the SSM
                          can be found from SSM through: Admin, Configure
```

HPSS, Server Configuration, Security Information screens.

-s <Storage Class> An optional argument. If the storage class is is specified, the output will be restricted to volumes in that class.

-f <Family ID> An optional argument. If the family id is is specified, the output will be restricted to volumes in that family. Family ID is not defined for disk but will be output to maintain a consistent format.

-l An optional argument. If the “locked” option is specified, the output will be restricted to volumes with a storage map state which is NOT unlocked.

DEFAULTS

dump_sspvs gets default values for the following from two sources. First, the execution environment of the program is checked for the environment variables listed below. For any that are defined, the values become the defaults. Second, for each environment variable not found in the first step, a value is assigned from the HPSS system default values.

SS Descriptive Name
If not supplied on the command line, defaults to the value of the HPSS environment variable HPSS_DESC_SSTAPE.

HPSS Generic Config Name
If not supplied on the command line, defaults to the value of the HPSS environment variable HPSS_CONFIG_SERVER.

Utility Principal Name
If not supplied on the command line, defaults to the value of the HPSS environment variable HPSS_PRINCIPAL_SSM.

Utility KeyTab Pathname
If not supplied on the command line, defaults to the value of the HPSS environment variable HPSS_KEYTAB_FILE_SERVER.

EXAMPLE
The following command will list the physical volumes managed by the storage server “ss_tape”.

```
     $ dump_sspvs -g /.:/encina/sfs/hpss/serverconfig
                  -p hpss_ssm -k /krb5/hpss.keytabs
                  -d ss_tape

                                    Storage
Volume      State         FamilyID  Class       Used Data       Total Space

NAN00000    FREE locked    5001       3       8,999,837,560   21,474,836,480
NAN04000    FREE           6639       3       1,470,010,357   21,474,836,480
NAN04100    FREE              0       3                   0   21,474,836,480
NAN04200    FREE              0       3                   0   21,474,836,480
WEH01000    FREE           6639       3                  72    9,126,805,504
WEH01100    FREE              0       3                   0    9,126,805,504
WEH02000    FREE           6639       4         239,309,458   42,949,672,960
WEH02100    FREE           6639       4              stripe
WEH02200    FREE              0       4                   0   42,949,672,960
WEH02300    FREE              0       4              stripe



Total available bytes:        190,052,302,848
Total used bytes:              10,709,157,447


Storage Class 3
Total available bytes:        104,152,956,928
Total used bytes:              10,469,847,989


Storage Class 4
Total available bytes:         85,899,345,920
Total used bytes:                 239,309,458


FILES


SEE ALSO
     dumpbf(7), dumppv_pvr(7), dumppv_pvl(7), lsvol(7)

SCCS
     man/dump_sspvs.7, gen, 4va   11/23/98   13:26:54
```

I.14 dumpbf — List the Bitfile Segments for a File Pathname

```
NAME
     dumpbf - List the bitfile segments for a file pathname.

SYNTAX
     dumpbf [ -f File Pathname -d Utility Server(CDS) Name -p Utility
     Principal Name -k KeyTab Path -h HPSS Generic Server Path
     -b BFS Descriptive name -n Name Server Descriptive Name
     -s SS Generic Descriptive name ]

DESCRIPTION
     The “dumpbf” HPSS utility will print an ordered set of bitfile
     segments by Storage Type and byte offset within bitfile for the
     input file pathname. This routine will connect to the nameserver
     in order to retrieve the bitfile Id for the specified pathname.
     The bitfile segments will be read from metadata and sorted as
     they are read and placed in a list.  After all the segments have
     been read and sorted, the results are sent to standard output.
     Included in the output for a segment which resides on a tape
     virtual volume will be the physical volume name(s).




OPTIONS
     -f <File Pathname>    The pathname of the file to dump.

     -b <BFS Desc. Name>   The descriptive name of the HPSS Bitfile Server
                           which manages the bitfiles to be scanned.
                           The descriptive name of a bitfile server
                           can be found from SSM through: Admin, Configure
                           HPSS, Server Configuration screens.

     -n <CNS Desc. Name>   The descriptive name of the HPSS Name Server
                           which manages the pathnames to be scanned.
                           The descriptive name of a Name server
                           can be found from SSM through: Admin, Configure
                           HPSS, Server Configuration screens.

     -s <SS Desc. Name>    The descriptive name of the HPSS Storage Server
                           which manages the Volumes where segments reside.
                           The descriptive name of a storage server
                           can be found from SSM through: Admin, Configure
                           HPSS, Server Configuration screens.

     -h <Generic Name>     The pathname of the SFS generic server configuration
                           file.

     -d <CDS Name>         The Cell Directory Service name which this
```

```
                        utility will run under.  Usually run as SSM.
                        The Server CDS name of the SSM
                        can be found from SSM through: Admin, Configure
                        HPSS, Server Configuration screens.

     -p <Principal Name>   The server principal name which this utility
                        will run under.  Usually run as SSM.
                        The Principal name of the SSM
                        can be found from SSM through: Admin, Configure
                        HPSS, Server Configuration, Security Information
                        screens.

     -k <Keytab Name>      The Keytab name which this utility
                        will run with.  Usually run as SSM.
                        The Keytab name of the SSM
                        can be found from SSM through: Admin, Configure
                        HPSS, Server Configuration, Security Information
                        screens.

EXAMPLE
     The following command will list the segments of the file
     "/dsf/Home0508". The file resides on a two wide tape stripe
     volume.  The descriptive names of the servers are:
     ss tape(storage server), bfs(bit file server) and cns(Name Server).


     $ dumpbf -f /dsf/Home0508 -h /.:/encina/sfs/hpss/serverconfig
              -d /.:/hpss/ssmsm -p hpss_ssm -k /krb5/hpss.keytabs
              -b bfs -n cns -s "ss tape"


      Bitfile segment dump of: /dsf/Home0508
      Bitfile Id: Object UUID 003785b6-26b1-1191-ab55-02608c2facae
      ServerDependent 00743a0658eb1005a821000000000004

                  **Tape Segments**:
       StorageClass: 3
      Number of segments at this level = 1
      Total length of segments at this level = 90573824

      Byte Offset: 0
      Segment Length: 90573824   PV(s):  JIM00200    JIM00400



FILES


SEE ALSO
     lsvol(7)


SCCS
```

```
man/dumpbf.7, gen, 4va   2/18/97   12:22:45
```

I.15 dumphpssfs — Dump HPSS Fileset

```
NAME
    dumphpssfs - for Dump HPSS Fileset


SYNTAX
    dumphpssfs <Fileset Name> <Output File> [-c] [-r]

    where

    <Fileset Name>  The name of the HPSS Fileset to be dumped.

    <Output File>   The name of the file to write the ASCII dump.

    -c              Optional command indicating the user wishes to dump
                    the consistency flags.  These flags describe
                    synchronization states between DFS and HPSS.  The
                    default is to not dump the consistency flags.
                    Dumping these flags will degrade the performance
                    and is not required by the 'loadtree' tool.
                    Dumping the consistency flags may be useful on an
                    already mirrored fileset for synchronization
                    verification.

    -r              Optional command indicating the user wishes to
                    restart the dump where the previous dump left off.
                    This option may be used if a previous run of
                    'dumphpssfs' terminated prematurely.  Note: The
                    Restart option assumes that the fileset being
                    dumped has NOT been altered since the last dump
                    attempt; unpredictable results may occur if the
                    fileset has been altered.


DESCRIPTION
    dumphpssfs (for Dump HPSS Fileset) is a utility used in the first
    step of a three step process to import an HPSS fileset into DFS.

    The dumphpssfs utility rumbles through an HPSS fileset and
    generates an ASCII representation of the fileset name space.  The
    resulting data can be used by another utility to generate the same
    fileset on a another system.


ENVIRONMENT
    The dumphpssfs utility is initiated from the command line.  The
    utility differs from some other HPSS utilities in that it uses the
    Client API library.
```

```
    The user must have a dce_login as “hpss_dmg” and must have
    initialized the required Client API environment variables.  The
    environment variables can be set by sourcing the
    $HPSS_ROOT/config/hpss_env file if these environment variables have
    been set up properly by the system administrator.  See the HPSS
    Programmer’s Reference Guide for more information on the required
    Client API environment variables.


ASSOCIATED FILES
    The dumphpssfs utility creates two output files: an output file
    specified on the command line and a restart file in the current
    working directory called DumpHPSSFSRestartFile.

    The first file is specified on the command line as <Output File>.
    This file will contain the ASCII representation of the fileset.
    This file can be used as input to the DMAPI “loadtree” utility.

    The format for each name space object in the <Output File>:

    ====================
    LEVEL: <level>
    TYPE: <type string>
    NAME: <object name>
    DMHANDLE: <length>-<handle>
    EVENTS: 0
    NUMEVENTS: 0
    PERS: 0
    PERSMAN: 0
    DTIME: (<decimal current time>) <text current time>
    CHANGE: 0
    DEV: 0
    INO: 1
    MODE: <mode>
    NLINK: <number of links>
    UID: <user id>
    GID: <group id>
    RDEV: 0
    SIZE: <filesize>
    ATIME: (<decimal last time read>) <text last time read>
    MTIME: (<decimal last time written>) <text last time written>
    CTIME: (<decimal last time metadata update>) \
                                <text last time metadata update>
    BLOCKSIZE: <blocksize>
    NUMBLOCKS: <number of blocks>
    FSID: <fileset number in hex>
    ACCOUNT: <account number - if unix-style accounting should be UID>
    COS: <class of service>
    COMMENT: <comment>
    FAMILYID: <family id>
    LOCATION: <location (0 means local)>
```

```
    MACSECLABEL: <macseclabel>
    ACL List: <number of ACLs> OR <default>
        For each ACL when not “default”:
            ENTRYTYPE: <ACL Entry Type (eg, user_obj, group_obj)>
            PERMS: <permisssions>
            EXPDATE: <expiration date>
            ENTRYID: <Entry Id>
            LOCATION: <Location (0 if local)>
    DM Attributes:
        HPSS_ID = <hex dump of the BitFileId followed by the Object Handle>
        if SYNC_FILESET_DATA Consistency Flag is set: SYNCFLAG
        if CACHE_DATA_VALID Consistency Flag is set: CAC_VAL
        if HPSS_DATA_VALID Consistency Flag is set:  HPS_VAL
    if “file”
        if “hardlink”: HARDLINK: YES
                       LINKPATH: <pathname>
        else
                       HARDLINK: NO
    if “junction”
        JUNCTIONTYPE: regular
        JUNCTIONPATH: <Fileset Name> [<pathname>]
    if “symlink”
        SYMLINK: <pathname>
    ====================
```

The second file, DumpHPSSFSRestartFile, is created in the current working directory. This file is used as a checkpoint/restart file. Information on each hardlink contained in the fileset will be written to this file. This information is needed by the [-r], the Restart feature. Upon Restart, this file will be read, processed, and possibly more logs will be written to it. Therefore restarts should be executed in the same directory from which the initial dump was taken. If the fileset being dumped does not contain any hardlinks, this file will be empty. After a successful dump of a fileset, this file can be deleted by the user.

The format for every DumpHPSSFSRestartFile entry is:

<Object Handle> <Link Count> <Path>

EXAMPLE

In the following example, an HPSS fileset named cnsClient_HPSSOnly_Fileset.piniE is dumped to an Unix file named ‘dump.out’.

```
ksh
dce_login hpss_dmg
. $HPSS_ROOT/config/hpss_env
```

```
/usr/lpp/hpss/bin/dumphpssfs cnsClient_HPSSOnly_Fileset.piniE dump.out
Using file ‘DumpHPSSFSRestartFile’ for HardLink Checkpoint/Restart capabilities.
Dumping=./.
Dumping=./TestDir
Dumping=./TestDir/abcDir
Dumping=./TestDir/abcDir/ABCfile
Dumping=./TestDir/abcfile
Number of objects processed:   (5)
        Files:                 (2)
        Directories:           (3)
        Hard Links:            (0)
        Symbolic Links:        (0)
        Junctions:             (0)



SCCS
    man/dumphpssfs.7, gen, 4va   11/20/98   16:43:00
```

I.16 dumppv_pvl — List the Physical Volumes Managed by a PVL

NAME

dumppv_pvl - List the physical volumes managed by a PVL.

SYNTAX

```
dumppv_pvl [ -l PVL Descriptive Name -d Utility Server(CDS) Name
-p Utility Principal Name -k KeyTab Path -h HPSS Generic Server Path ]
```

DESCRIPTION

This utility will produce a list of the physical volumes managed by the input PVL. The list will include the name, media type and the Storage Class(if allocated) of the volume.

OPTIONS

-l <PVL Desc. Name>	The descriptive name of the HPSS PVL Server which manages the volumes to be scanned. The descriptive name of a PVL server can be found from SSM through: Admin, Configure HPSS, Server Configuration screens.
-h <Generic Name>	The pathname of the SFS generic server configuration file.
-d <CDS Name>	The Cell Directory Service name which this utility will run under. Usually run as SSM. The Server CDS name of the SSM can be found from SSM through: Admin, Configure HPSS, Server Configuration screens.
-p <Principal Name>	The server principal name which this utility will run under. Usually run as SSM. The Principal name of the SSM can be found from SSM through: Admin, Configure HPSS, Server Configuration, Security Information screens.
-k <Keytab Name>	The Keytab name which this utility will run with. Usually run as SSM. The Keytab name of the SSM can be found from SSM through: Admin, Configure HPSS, Server Configuration, Security Information screens.

EXAMPLE

```
     The following command will list the physical volumes
     managed by the PVL “pvl_ampex”.


     $ dumppv_pvl -l pvl_ampex -h /.:/encina/sfs/hpss/serverconfig
               -d /.:/hpss/ssmsm -p hpss_ssm -k /krb5/hpss.keytabs

PVL Physical Volumes:

        Name                    Type            Class
        00006000              MT_TAPE_DD2       unallocated
        00006100              MT_TAPE_DD2       unallocated
        00006200              MT_TAPE_DD2       unallocated
        00006300              MT_TAPE_DD2       unallocated
        00006400              MT_TAPE_DD2       unallocated
        00006600              MT_TAPE_DD2       unallocated
        00006700              MT_TAPE_DD2       unallocated
        00006800              MT_TAPE_DD2       unallocated
        00006900              MT_TAPE_DD2       unallocated
        BUD13000            MT_TAPE_3490E       2
        BUD13100            MT_TAPE_3490E       3
        BUD13200            MT_TAPE_3490E       3
        BUD13300            MT_TAPE_3490E       2
        BUD13400            MT_TAPE_3490E       2
        BUD13500            MT_TAPE_3490E       unallocated
        BUD13600            MT_TAPE_3490E       unallocated
        BUD13700            MT_TAPE_3490E       unallocated
        BUD13800            MT_TAPE_3490E       unallocated
        DD000000          MT_DISK_DEFAULT       1
        GENERI00                     ?          unallocated
        JIM00100            MT_TAPE_3490E       2
        JIM00200            MT_TAPE_3490E       unallocated
        JIM00300            MT_TAPE_3490E       unallocated
        JIM00400            MT_TAPE_3490E       unallocated
        MVS18100             MT_TAPE_3590       unallocated
        MVS18300             MT_TAPE_3590       unallocated
        MVS18400             MT_TAPE_3590       5
        MVS18500             MT_TAPE_3590       5
        MVS18600             MT_TAPE_3590       unallocated



FILES


SEE ALSO
     dumppv_pvr(7), lsvol(7)

SCCS
     man/dumppv_pvl.7, gen, 4va   11/23/98   13:30:09
```

I.17 dumppv_pvr — List the Physical Volumes Managed by a PVR

NAME

dumppv_pvr - List the physical volumes managed by a PVR.

SYNTAX

```
dumppv_pvr [ -l PVR Descriptive Name -d Utility Server(CDS) Name
-p Utility Principal Name -k KeyTab Path -h HPSS Generic Server Path ]
```

DESCRIPTION

This utility will produce a list of the physical volumes managed by the PVR. The output includes a count of the number of mounts performed on a cartridge since it was placed in service.

OPTIONS

-l <PVR Desc. Name>	The descriptive name of the HPSS PVR Server which manages the volumes to be scanned. The descriptive name of a PVR server can be found from SSM through: Admin, Configure HPSS, Server Configuration screens.
-h <Generic Name>	The pathname of the SFS generic server configuration file.
-d <CDS Name>	The Cell Directory Service name which this utility will run under. Usually run as SSM. The Server CDS name of the SSM can be found from SSM through: Admin, Configure HPSS, Server Configuration screens.
-p <Principal Name>	The server principal name which this utility will run under. Usually run as SSM. The Principal name of the SSM can be found from SSM through: Admin, Configure HPSS, Server Configuration, Security Information screens.
-k <Keytab Name>	The Keytab name which this utility will run with. Usually run as SSM. The Keytab name of the SSM can be found from SSM through: Admin, Configure HPSS, Server Configuration, Security Information screens.

EXAMPLE

```
     The following command will list the physical volumes
     managed by the PVR “pvr_ampex”.


     $ dumppv_pvr -l pvr_ampex -h /.:/encina/sfs/hpss/serverconfig
               -d /.:/hpss/ssmsm -p hpss_ssm -k /krb5/hpss.keytabs


     PVR Physical Volumes:

             Name    Total Mounts
             000060      118
             000061      113
             000062      101
             000063      110
             000064       96
             000065        2
             000066        4
             000067        7
             000068       57
             000069        2
             000070       17
             000071       43
             000072        0
             000073        0
             000074        0
             000075        0
             000076        0
             000077        0
             000078        0
             000079        0
             GENERI        0



FILES


SEE ALSO
     dumppv_pvl(7), lsvol(7)


SCCS
     man/dumppv_pvr.7, gen, 4va   11/23/98   13:34:14
```

I.18 gen_reclaim_list — HPSS Volume Reclaim Utility

```
NAME
     gen_reclaim_list - HPSS Volume Reclaim Utility

SYNTAX
     gen_reclaim_list [ -d SS Desc. Name -g HPSS Generic Server Path
     -c Utility Server(CDS) Name -p Utility Principal Name -k KeyTab Path
     -s Storage Class -n Number -f Output File]

DESCRIPTION
     "gen_reclaim_list" is a HPSS System Maintenance Utility.
     It is the first utility of a pair which compose the reclaim
     procedure.  It is the utility which produces a list of ascii
     VVID hpssoids which is sent to the output file specified
     through an argument. The volumes and class of service of the
     storage server specified will be searched.  The ascii soids
     of volumes in EMPTY state will be stored in the specified
     output file.  The number of volumes to find is an argument,
     volumes will be searched until the number is found or all
     volumes managed by the storage server are scanned.




OPTIONS
     -d <SS Desc. Name>    The descriptive name of the HPSS Storage Server
                           which manages the Volumes to be reclaimed.
                           The descriptive name of a storage server
                           can be found from SSM through: Admin, Configure
                           HPSS, Server Configuration screens.

     -g <Generic Name>     The pathname of the SFS generic server configuration
                           file.

     -c <CDS Name>         The Cell Directory Service name which this
                           utility will run under.  Usually run as SSM.
                           The Server CDS name of the SSM
                           can be found from SSM through: Admin, Configure
                           HPSS, Server Configuration screens.

     -p <Principal Name>   The server principal name which this utility
                           will run under.  Usually run as SSM.
                           The Principal name of the SSM
                           can be found from SSM through: Admin, Configure
                           HPSS, Server Configuration, Security Information
                           screens.

     -k <Keytab Name>      The Keytab name which this utility
                           will run with.  Usually run as SSM.
                           The Keytab name of the SSM
```

```
                         can be found from SSM through: Admin, Configure
                         HPSS, Server Configuration, Security Information
                         screens.

     -s <Storage Class>  The storage class of the volumes to scan.
                         The storage classes supported can be found
                         from SSM through: Monitor, Storage Class List
                         screens.

     -n <Number>         The number of empty volumes to reclaim.

     -f <FILE>           Is a file containing a list of the volume ascii
                         HPSS soids to be reclaimed.  This list consists
                         of the volumes found in an EMPTY state.

EXAMPLE
     The following command will scan the volumes in storage class
     3 managed by the storage server “ss_tape” for 20 empty
     volumes.  The ascii soids of the volumes found will be
     placed in the file “recycle_list”.  The file “recycle_list”
     can then be used as input to the utility reclaim.


     $ gen_reclaim_list -d ss_tape -g /.:/encina/sfs/hpss/serverconfig
               -c /.:/hpss/ssmsm -p hpss_ssm -k /krb5/hpss.keytabs
               -s 3 -n 20 -f recycle_list


FILES


SEE ALSO
     reclaim.ksh(7), reclaim(7), repack(7)

SCCS
# man/gen_reclaim_list.7, gen, 4va   11/23/98   11:26:49
```

I.19 getdmattr — Get DM attributes for a file stored in DFS

```
NAME
     getdmattr - get DM attributes for a file that is stored in DFS.

SYNTAX
     getdmattr <Path Name>

DESCRIPTION
     getdmattr is a utility that can be run on the DFS server machine
     that contains the Episode file system that contains the file. This
     utility will be used generally to check on files that appear to
     be inconsistent between DFS and HPSS.

     This tool must be run as root.

     The output from this utility will describe the state of the file and
     will include all DM attributes associated with this file, the region
     information, and the file data residency. Most of the output
     will only be important for debugging purposes.

     See “HPSS Administration Guide: DFS Configuration and Management”
     for more details.

PARAMETERS

     Path Name       - The path to the file through the local UNIX mount
                       point.


EXAMPLE
     The following command will get file and DM attributes information
     for a file on a DMLFS aggregate on the local machine.

     getdmattr /var/hpss/hdm/hdm1/aggr/test.aggr/test.fileset/file.0

FILES


SEE ALSO
     setdmattr(7) archivecmp(7) archivedump(7) archivelist(7)

SCCS
     man/getdmattr.7, gen, 4va 11/20/98 15:22:14
```

I.20 hdm_admin — HPSS/DMAP Administrative Utility

```
NAME
     hdm_admin - manage the HDM server

SYNTAX
     hdm_admin [-b bin] [-k key] [-s id] [-v var] [-y] [command]

PARAMETERS
     bin         Path to HPSS executables.
     key         Shared memory key.
     id          Server ID.
     var         Path to HDM's data files.
     command     Optional one-liner command to execute.

DESCRIPTION
     This utility is used to manage the HDM server.  The utility can
     start the HDM, control certain aspects of the its operation,
     inspect its state, restart it, and stop it.  The utility operates
     by inspecting the HDM's shared memory, and therefore can only be
     run on the machine where the HDM runs.  The utility will work even
     if the HDM is not running, in which case the output shows what the
     HDM was doing when it last ran.

     The bin parameter defines the path to the HPSS executables.  The
     default value is /usr/lpp/hpss/bin.

     The key parameter defines a key used to find the HDM's shared
     memory segment.  The default value is 3789.

     The id parameter defines the server ID.  This value determines
     which section of config.dat will be used to define the HDM's
     configuration.  The default value is 1.

     The var parameter defines the directory where the HDM's data files
     are kept.  This will typically be specified as an explicit path
     such as /var/hpss/hdm or /var/hpss/hdm/hdm1.  If there are several
     HDMs running on one machine, the path can also be specified as a
     string containing the characters '%d'.  The program will substitute
     the server id for %d to determine the actual path.  For example, if
     the server id is 1 and var is /var/hpss/hdm/hdm%d, then the actual
     path will be /var/hpss/hdm/hdm1.  The parameter must contain '%d'
     if hdm_admin's monitor command is to be used.  The default value is
     /var/hpss/hdm.

     If the -y option is specified, the HDM will not ask the user
     whether to go ahead with certain sensitive options, but will
     procede as if the user had answered "yes".  This option allows
     hdm_admin to be used in scripts such as rc.dfs.  The default is to
     always ask.

     If a command parameter is provided, hdm_admin executes the command
```

immediately, then returns control to the shell. The command, which may contain one or more words, must appear last on the line after the switches. A list of the legal commands is shown below. If the command is omitted, hdm_admin enters interactive mode, prompting for commands and executing them until the user quits the program.

In general, when hdm_admin finishes, it does not return a useful error code to the shell. There are two exceptions to this rule. The 'start -wait' command returns 0 if the HDM was started successfully, and a non-zero value otherwise; and the 'ps' command returns 0 if the HDM is running, and a non-zero value otherwise. These values can be used by scripts such as rc.dfs to determine if it is safe to export DFS aggregates.

Environment Variables

If any of the command line parameters is omitted, the default values listed above can be overridden by setting one or more of the following environment variables:

HPSS_HDM_SHMEM_KEY

If set, this determines the shared memory key. The value can be overridden by using the -k option on the command line.

HPSS_HDM_SERVER_ID

If set, this determines the server ID. This value determines which section of config.dat will be read to find the HDM's configuration. The value can be overridden by using the -s option on the command line.

HPSS_PATH_BIN

If set, this determines the path to the hpss_hdm executable. The value is used by the start command to locate the directory for the file. The file's name is always hpss_hdm. The default path is determined by hpss_env, and is typically /usr/lpp/hpss/bin. The value can be overridden by using the -b bin option on the command line.

HPSS_PATH_HDM

If set, this determines the path to the directory where the HDM keeps log files, configuration files, etc. The path is typically /usr/var/hdm or /usr/var/hdm/hdm%d. The value can be overridden by using the -v option on the command line.

COMMANDS

The following is a summary of hdm_admin's subcommands. A detailed description of each subcommand will be found below.

again	Repeat the last command
comment	Enter a comment in the HDM's message log
dlog	Inspect the destroy log
exit	Quit out of hdm_admin
fsstat	Print the file system table

```
     ftstat       Print the fileset table
     help         Get information on how to use hdm_admin
     kill_all     Kill all HDM processes ungracefully
     locks        Inspect the HDM's locks and semaphores
     migrate      Start a file migration cycle
     mlog         Inspect the main event log
     monitor      Switch focus to a different HDM
     printenv     Print environment variables
     ps           Find what the HDM processes are doing
     purge        Start a file purge cycle
     queue        Inspect the event queue
     quit         Quit out of hdm_admin
     report       Change what messages are being logged
     restart      Restart the HDM
     start        Start the HDM
     stats        Inspect statistics
     stop         Stop the HDM
     tcp          Enable/disable changes to aggregates from HPSS
     usage        Show the command line usage for hdm_admin
     zlog         Inspect the zap log

     To get help on one of hdm_admin's subcommands, enter the command
     "help <subcommand>", where <subcommand> is the name of the
     subcommand for which additional information is desired.

     Note: depending on the state of the HDM, some subcommands may not
     be available.  For example, the HDM cannot be restarted if it isn't
     running.

SUBCOMMANDS
     The detailed usage of each hdm_admin subcommand is shown below:

     again
        Purpose     Repeat the last command

     comment <comment>
        Purpose     Enter a comment in the HDM's message log
        <comment>   The comment to be entered - one or more words.

     dlog
        Purpose     Inspect the destroy log

     exit
        Purpose     Quit out of hdm_admin

     fsstat [<media>]
        Purpose     Inspect the file system table
        <media>     Media designator (e.g., /dev/lv01).  If omitted,
                    list all file systems.

     ftstat [<ftname]>
        Purpose     Inspect the fileset table
        <ftname>    Fileset name.  If omitted, list all filesets.
```

```
help [<command>]
   Purpose     Get information on how to use hdm_admin
   <command>   Command for which help is wanted.  If omitted,
               show a list of hdm_admin's commands.

kill_all [<signal>]
   Purpose     Kill all HDM processes ungracefully
   <signal>    Signal to use.  Legal values include -TERM, -15,
               -KILL, and -9.  The default is -TERM.

locks
   Purpose     Inspect the HDM's locks and semaphores

migrate <aggregate>
   Purpose     Start a file migration cycle
   <aggregate> Aggregate where migration is to be started

mlog
   Purpose     Inspect the main event log

monitor [<id>]
   Purpose     Switch focus to a different HDM
   <id>        The id of the server to switch to.  If omitted,
               report which server is currently being monitored.
               To use this command effectively, the HDMs must be
               set up carefully.  See the discussion below for more
               information.

printenv
   Purpose     Print environment variables

ps
   Purpose     Find what the HDM processes are doing

purge <aggregate>
   Purpose     Start a file purge cycle for a file system
   <aggregate> Aggregate where purge is to be started

queue
   Purpose     Inspect the event queue

quit
   Purpose     Quit out of hdm_admin

report [<list>]
   Purpose     Change what messages are being reported
   <list>      List of things to report.  If omitted, display the
               current list.  The list can contain one or more of
               the following keywords: alarm, event, request,
               security, accounting, status, trace, debug, or all.
               If all is specified, everything will be reported.
```

```
     restart
        Purpose     Restart the HDM

     start [-force] [-noisy] [-quiet] [-wait] [<exec> [<config> [<id>]]]
        Purpose     Start the HDM
        -force      Force the HDM to start, even if it is already
                    running.
        -noisy      Enable messages to stdout if they are enabled in
                    config.dat.  This is the default when the start
                    command is specified on hdm_admin's command line.
        -quiet      Turn off messages to stdout, even if they are
                    enabled in config.dat.  This is the default when
                    running interactively.
        -wait       Wait for HDM to start before returning control to
                    hdm_admin.
        <exec>      The name of the executable.  The default depends on
                    the setting of HPSS_PATH_BIN, but is typically
                    /usr/lpp/hpss/bin/hpss_hdm.
        <config>    The name of the configuration file. The default
                    depends on the setting of the environment variable
                    HPSS_PATH_HDM, but it is typically
                    /var/hpss/hdm/hdm1/config.dat.
        <id>        The ServerID section to be used from the
                    configuration file.  The default is 1.

     stats
        Purpose     Inspect statistics

     stop
        Purpose     Stop the HDM

     tcp { enable | disable } [<aggregate>]
        Purpose     Set a flag in the TCP process to enable or disable
                    changes to an aggregate from the HPSS side.  If
                    <aggregate> is omitted, enable or disable all
                    aggregates managed by the HDM.  If the disable
                    option is specified, unmount all locally mounted
                    filesets on the affected aggregates.  Note: the
                    disable command is typically used just before
                    detaching a DFS aggregate.

     usage
        Purpose     Show the command line usage for hdm_admin

     zlog
        Purpose     Inspect the zap log

USING THE 'MONITOR' COMMAND EFFECTIVELY:
     The monitor command is used to switch hdm_admin's focus from one
     HDM to another.  This is only useful if there are several HDMs
     running on one machine.  To use the command effectively, the data
     files for the different HDMs must be kept in directories named
     according to the server id.  For example, the data files for server
```

```
     0 might be kept in /var/hpss/hdm/hdm0 and the files for server 1
     might be in /var/hpss/hdm/hdm1.  To tell hdm_admin the convention
     that is being used, include the characters '%d' in the var
     parameter or the HPSS_PATH_HDM environment variable.  For example
     use /var/hpss/hdm/hdm%d for the example above.

     The characters '%d' are only required if the monitor command is to
     be used.  If they are not present, hdm_admin will not be able to
     keep the different servers' data files separate.

     There is one other requirement for using the monitor command: the
     server's keys must be related to the server id.  For example, if
     server 0 uses key 3700, then server 1 must use key 3701, server 2
     must use key 3702, etc.

EXAMPLES
     To start the HDM using a shell command:
       # hdm_admin start

     To start the HDM and wait to see if it comes up:
       # hdm_admin start -wait

     To see if the HDM is running before exporting files:
       # hdm_admin ps
       # if [ $? = 0 ]; then
       #    dfsexport aggr1
       # fi

     To use hdm_admin interactively to determine what the HDM is doing:
       # hdm_admin
       > ps
       > stats
       > dlog
       > mlog
       > zlog
       > queue
       > exit

     To monitor two HDM servers without restarting hdm_admin:
       # export HPSS_PATH_HDM=/var/hpss/hdm/hdm%d
       # hdm_admin
       > monitor 1
       > ps                -- show processes for server 1
       > monitor 2
       > stop              -- stop processes for server 2

     To enter comments in the message log:
       # hdm_admin comment File system aggr1 is full.
       # hdm_admin comment Migrating files to clear it up.

     To start a migration cycle on an aggregate named aggr1:
       # hdm_admin migrate aggr1
```

```
     To start migration and purge cycles:
       # hdm_admin
       > migrate aggr1
       > purge aggr1
       > quit

     To stop the HDM:
       # hdm_admin stop

     To get help on how to use hdm_admin or one of its commands:
       # hdm_admin
       > usage
       > help
       > help start

FILES
     config.dat     Configuration file for the HDM
     hpss_hdm       Main overseer program for the HDM
     hpss_env       Script to set up the HPSS environment

SCCS
     man/hdm_admin.7, gen, 4va   3/24/99   16:50:26
```

I.21 hdmdump — Dump an ASCII representation of a DMLFS fileset

```
NAME
     hdmdump - Dump an ascii representation of a DMLFS fileset.

SYNTAX
     hdmdump <Fileset ID> <Fileset Name> <Fileset Path> [SetMigrate]

DESCRIPTION
     hdmdump can be run to dump an ascii representation of a DMLFS
     DFS fileset into stdout. This utility can be used to import both
     archived and mirrored filesets into HPSS.

     This tool must be run as root on the machine that contain the disk
     on which the fileset resides.

     When importing a DFS fileset into a mirrored HPSS fileset, the
     SetMigrate option must be set, and the output directed to a file
     that can be used be the utility “loadhpssfs” to load the HPSS
     fileset.

     When importing a DFS fileset into an archived HPSS fileset, the
     SetMigrate option should be set to mark files as migratable but
     the output can be sent to /dev/null since it is not needed to load
     the archive fileset in HPSS (The migration of the files will handle
     loading the archveid fileset).

     See “HPSS Administration Guide: DFS Configuration and Management”
     for more details.

PARAMETERS

     Fileset ID   - Numeric fileset identification.
                          (e.g 0.4, 0.234, etc.)

     Fileset Name - Character name for the fileset.

     Fileset Path - Path to where the fileset is mounted on the
                          local UNIX file system.

     SetMigrate   - This is an optional parameter that if set will mark
                    all files that do not contain the HPSS_ID DM attribute
                    with the attribute MIGRATE set to a value of 1. This
                    must be set during DFS fileset import into HPSS so
                    that file data will migrate into HPSS. If this is not
                    set then the HDM will not know which files need to
                    be migrated.

EXAMPLE
     The following command will send output to the file “hdmdump.out”
```

```
     for fileset 0.4 whose name is test.fileset and is currently locally
     mounted at "/var/hpss/hdm/hdm1/aggr/test.aggr/test.fileset" and will
     set the MIGRATE DM attribute for all files not already backed into
     HPSS.

     hdmdump 0.4 test.fileset \
        /var/hpss/hdm/hdm1/aggr/test.aggr/test.fileset SetMigrate \
                                                        > archdump.out

FILES


SEE ALSO
     loadhpssfs(7), loadtree(7), loadhpssid(7), loadhpssdmid(7),
     dumphpssfs(7)

SCCS
     man/hdmdump.7, gen, 4va 12/4/98 16:33:03
```

I.22 hpss.clean — Kill One or More HPSS Processes

NAME

hpss.clean - Kill one or more HPSS processes

SYNTAX

hpss.clean [-f] { <hpssProcessName> | all }*

DESCRIPTION

This program terminates all processes with a basename of <hpssProcessName>. All processes matching <hpssProcessName> via the "ps -e" command will be killed by signalling the process via the "kill" command. Up to 10 processes can be specified as arguments.

Specifying "all" as the process name will cause all HPSS processes, except those related to SSM and the Startup Daemon, to be signaled. SSM processes, as well as the Startup Daemon, can only be terminated via this utility by explicitly listing them on the command line.

The servers are killed in the following order, using the specified signal (-15 is SIGTERM, -9 is SIGKILL):

Type of Server	Process Base Name	Kill Signal
HPSS Accounting	hpss_acct	15
Metadata Monitor	hpss_mmon	15
MPS	hpss_mps	15
Mount Daemon	hpss_mnt1	15
NFS2 Daemon	hpss_nfs2	15
Mover	hpss_mvr	15
Mover (DCE)	hpss_mvr_dce	15
Mover (TCP)	hpss_mvr_tcp	15
Mover (TCP/IPI)	hpss_mvr_tcp_ipi	15
Mover (TCP/IPI/BMUX)	hpss_mvr_tcp_ipi_bmux	15
Mover (TCP/PFS)	hpss_mvr_tcp_pfs	15
Mover (TCP/PFS/IPI)	hpss_mvr_tcp_pfs_ipi	15
Mover (TCP/IPI)	hpss_mvr_tcp_ipi	15
3494 PVR	hpss_pvr_3494	15
Operator PVR	hpss_pvr_operator	15
3495 PVR	hpss_pvr_3495	15
STK PVR	hpss_pvr_stk	15
Ampex PVR	hpss_pvr_ampex	15
PVL	hpss_pvl	15
Tape Storage Server	hpss_ss_tape	15
Disk Storage Server	hpss_ss_disk	15
BFS	hpss_bfs	15
CNS	hpss_cns	15
Log Client	hpss_logc	15
Log Daemon	hpss_logd	15
Startup Daemon	hpssd	9

```
          SSM Data Server         hpss_ssmds             15
          SSM System Manager      hpss_ssmsm             15

     If the -f option is specified, the specified HPSS processes will be
     forced to terminate via the -9 (SIGKILL) signal.

     The user invoking this script must have the appropriate priviledge
     to kill the selected processes.

     This utility can also be used with the rsh or dsh command to remotely
     kill HPSS processes.

EXAMPLE
     To kill the HPSS startup daemon, issue:

         % hpss.clean hpssd

     To kill all running HPSS processes (except SSM and the Startup Daemon),
     issue:

         % hpss.clean all

     The following command will kill all Log Clients and Movers:

         % hpss.clean hpss_mvr hpss_logc

     To kill all HPSS processes as well as all SSM processes, issue:

         % hpss.clean all hpss_ssmds hpss_ssmsm


FILES


SEE ALSO

     rc.hpss

SCCS
     man/hpss.clean.7, gen, 4va   6/18/96   17:45:43
```

I.23 hpss_delog — HPSS Delog Utility

NAME

hpss_delog - HPSS Delog Utility

SYNTAX

```
hpss_delog Logfile Delogfile[-sStart time] [-eEnd Time]
     [-dDescriptive Name]
     -t<alarm event request security accounting debug trace>
     [-uUser] [-pPrincipal] [-kKeytab File] -l[Location Server]
```

DESCRIPTION

"hpss_delog" is a utility used to analyze the activity and behavior of HPSS. The utility is generally used to troubleshoot problems. The delog process retrieves specific records from the HPSS central log file, converts the records to a readable text format, and outputs the text to a specified file.

The central log files names available for delogging are logfile01 and logfile02. These files reside in the directory name specified in the Log Daemon configuration file. Log files archived to HPSS may also be delogged. Archived log files reside in the /log HPSS directory. To determine the log times for an archvied log file, list the log files. The timestamp of the time when the file was archived to HPSS is appended to the end of the log file name. The format of the archive file names is logfile01_YYMMDDHHMM or logfile02_YYMMDDHHMM.

If no options other than the input log file pathname and output delog pathname are specified, all records in the log are delogged. Additional options allow filtering by start time (-s), end time (-e), server descriptive name(s) (-d), log record type(s) (-t), and user name(s) (-e). Options are described below:

OPTIONS

Logfile	The pathname of the input log file name. The log file may be the primary or secondary logfile (logfile01 / logfile02), or an archived log.
Delogfile	The pathname of the output delog file.
-s<Start Time>	The start time of the first record to retrieve. If specified, the time format is yyyy:mm:dd:hh:mm:ss (year is optional). At most, one -s option may be entered. The default is the first record in the log.
-e<End Time>	The end time of the last record to retrieve. If specified, the time format is yyyy:mm:dd:hh:mm:ss (year is optional). At most,

	one -e option may be entered. The default is the last record in the log.
-d<Descriptive Name>	A server descriptive name associated with the log records to retrieve. Multiple -d options may be specified. The default is all servers.
-t<Record Type>	A log record type associated with the log records to retrieve. Options are alarm, event, request, security, accounting, debug, and trace. Multiple -t options may be specified. The default is all record types.
-u<User>	A user name associated with the log records to retrieve. Multiple -u options may be specified. The default is all users. It should be noted that user names are not generally available from the core HPSS servers.
-p<Principal>	The principal associated with the delog process. If not specified, the value will be obtained from the HPSS_PRINCIPAL_LOG environment variable if defined or to hpss_log. Note: If not specified, the principal will be obtained (in order) from the HPSS_PRINCIPAL environment variable, the HPSS_PRINCIPAL_LOG environment variable, or default to hpss_log.
-k<Keytab File>	The keytab file associated with delog principal. If not specified, the value will be obtained from the HPSS_KEYTAB_FILE_SERVER environment variable if defined or to /krb5/hpss.keytabs. Note: If not specified, the keytab pathname will be obtained (in order) from the HPSS_KTAB_PATH environment variable, the HPSS_KEYTAB_FILE_SERVER environment variable, or default to /krb5/hpss.keytabs.
-l<Location Server>	The Location Server CDS name. If not specified, the value will be obtained from the HPSS_LS_NAME environment variable.

EXAMPLE

Delog all records from log file /var/hpss/log/logfile01 to output file /hpss/data/delog.out.

```
hpss_delog /var/hpss/log/logfile01 /hpss/data/delog.out
```

Delog records between 1:00pm and 3:00pm on March 1, 1999.

```
hpss_delog /var/hpss/log/logfile02 /hpss/data/delog.out
  -s1999:03:01:13:00:00 -e1999:03:01:15:00:00
```

```
    Delog alarm and debug records for server PVL between 1:00pm and 3:00pm on
    March 1, 1999.
        hpss_delog /var/hpss/log/logfile02 /hpss/data/delog.out
          -s1999:03:01:13:00:00 -e1999:03:01:15:00:00 -talarm -tdebug -dPVL

    Delog only alarm records.
        hpss_delog /var/hpss/log/logfile01 /hpss/data/delog.out -talarm

    Delog alarm and event records for the Bitfile Server and Disk Storage
    Server between 8:00am and 9:30am on February 28. The log file referenced
    has been archived to hpss.
        hpss_delog /log/logfile01_9902280745 /hpss/data/delog.out -
s02:28:08:00:00
          -e02:28:09:30:00 talarm -tevent -d'Bitfile Server' -d'Storage Server'

FILES

SEE ALSO

SCCS
     man/hpss_delog.7, gen, 4.1.1, 4.1.1.1   6/4/99   18:34:45
```

I.24 hpssuser — HPSS User Management Utility

```
NAME
     hpssuser - HPSS User Management Utility

SYNTAX
     hpssuser [options...]

DESCRIPTION
     “hpssuser” is an HPSS User Management Utility.  It has the capability
     of adding, deleting, and listing users.

OPTIONS
     -add <user>           Add a DCE and FTP user
     -del <user>           Delete a DCE and FTP user
     -list <user>          List a DCE and FTP user (‘-list ALL’ lists all
                           DCE and FTP users)
     -h | -help            Show this help message

     -dce | -ftp           These options can be used with “-add”, “-del”, and
     -ssm | -aix           “-list” to specify what type of users to work with.
                           By default both DCE and FTP users are worked on.

     -all                  This option is the same as typing -dce, -ftp, -ssm,
                           and -aix.


     -aaid <id>            Use this option if Site Accounting is used, otherwise
                           Unix style accounting is assumed.

     [ Use the following options to avoid being prompted for a certain value ]

     -name         -fullname   -uid     -shell         -home
     -homeprefix   -group      -org     -cell_pass     -user_pass


EXAMPLE
     The following command will create a DCE and SSM user

     $ hpssuser -add nguyenj -dce -ssm

     The following command uses the “-fullname” and “-org” options to
     avoid being prompted for these values.

     $ hpssuser -add nguyenj -fullname “John T. Nguyen” -org hpss

FILES
     HPSS_ROOT/config/templates/s3_config.dat.template
```

```
                                    SAMMI Session configuration file

     HPSS_ROOT/config/templates/SAMMI.template
                                    SAMMI User X resource file

     HPSS_ROOT/bin/api_config.dat   SAMMI Session ID List

     HPSS_ROOT/sammi/hpss_ssm/user_authorization.dat
                                    SAMMI Authorization file

     /var/hpss/ftp/etc/ftppasswd    HPSS FTP Users File

SEE ALSO
     rgy_edit(1), scrub(7), hpss_pftppw(7), mkuser(1)


SCCS
     man/hpssuser.7, gen, 4va   4/18/96   11:21:36
```

I.25 insif — Interactive Name Server Interface

```
NAME
    INSIF - provides an interactive interface to the Name Server.

SYNOPSIS
    insif

DESCRIPTION
    INSIF (for Interactive Name Server InterFace) provides an interactive
    interface to the Name Server.  INSIF translates human commands into
    Name Server commands.  Any results received from the Name Servers are
    formatted and sent to standard out.  INSIF is used to administer,
    debug, and test the Name Server.

    The Storage Area is a space internally set aside by INSIF to store a
    BitfileId, an object handle, and a FilesetId.  This Storage Area can be
    filled using the getbfid, getoh, getfsid, getohffs, or the rsa commands.

    The contents of the Storage Area can be viewed with the psa command and
    can be written to a file with the wsa command.


Interactive Options:
    aclformat           Will toggle the ACL output format between the old
                        'insif' style and 'acl_edit' style.  The default
                        is alc_edit style.

    cd <PathName>       causes a change directory operation to take place.
                        INSIF will fetch the object handle named by the last
                        link in the PathName.  The INSIF prompt will display
                        the PathName to the directory the user is currently
                        'in'.  The newly fetched object handle will become
                        the WorkingDir.  The last link of PathName must be
                        a directory or the command will fail.

    clsa or csa         (for CLear Storage Area) causes the Storage Area to
                        be zeroed out.

    cpwdtosa            (for CoPy Working  Directory object handle TO Storage
                        Area object handle) the working directory object
                        handle is copied to the Storage Area. Any object
                        handle in the Storage Area is overwritten.

    del <P1> [<P2> ... <Pn>]  or  [*]
                        delete the entry(s) named by the last link of the
                        pathnames P1, P2 ... Pn.  Optionally, to delete ALL
                        entries in the current directory use 'del *'.

    delacl [<PathName>]
                        delete an ACL associated with PathName. If no
```

PathName is supplied an ACL associated with the current working directory will be deleted. INSIF will prompt for the ACL information.

delbg <PathName> (for Delete By Golly!) INSIF attempts to assist the user by making sure that orphaned Bitfiles are not created by a delete. This assistance can sometimes lead to problems such as not being able to delete bogus symbolic link entiries. delbg cuts right to the chase and deletes the entry!

delefs (for DELete Empty FileSet) loops through existing filesets and deletes those that are empty.

delfs <FilesetName> | -id [<FilesetId>]
(for DELete Fileset) the Fileset identified by the FilesetName or the FilesetId is deleted.

delfsuoh (for DELete Fileset Using Object Handle) the Fileset identified by the object handle currently in the Storage Area will be deleted.

deluoh (for DELete Using Object Handle) the object identified by the object handle in the Storage Area will be deleted. An object handle must be put into the Storage Area before this command will work. Use the 'getoh' command to fill the Storage Area.

fbn [<PathName>] (for Fix Bad Names) causes INSIF to rumble through the indicated directory 'fixing' up any bad (unprintable) names it finds. It does this by replacing any spaces, single or double quotes, or unprintable characters with random printable characters. If PathName is omitted, the names in the WorkingDir will be 'fixed'.

ga [<PathName>] (for Get Attributes) the attributes for the WorkingDir (or PathName if provided) will be fetched and displayed.

ganc [<PathName>] (for Get Attributes No Chase) the same as ga above, except that the last symbolic link (if any) will not be chased.

gauoh or getattrsuoh
(for Get Attributes Using Object Handle) this option is the same as the 'ga' option except that the object handle used is the one found in the storage area. An object handle must be put into the storage area before this command will work. Use the 'getoh' command to fill the Storage Area.

getacl [<PathName>] fetch and display the ACL(s) associated with

```
                    PathName.  If no PathName is supplied the ACL(s)
                    associated with the current working directory will
                    be displayed.

getbfid <PathName>  get the BitfileId of the file named by PathName.
                    The BitfileId will be fetched and placed into
                    StoredBFID. The contents of StoredBFID can be viewed
                    with the psa command.

gfsa <FilesetName> | [-id <FilesetId>]
                    get and display the Fileset attributes associated
                    with the given FilesetName or FilesetId. The user
                    supplies either a FilesetName or, optionally, a
                    FilesetId. To specify a FilesetId the -id parameter
                    must be used.

getfsaufsid or gfsaufsid
                    (for GET Fileset Attrs Using the stored FilesetID)
                    This command is the same as the getfsattrs command
                    except that the user does not supply a FilesetId.
                    The FilesetId in the Storage Area is used.

getfsid <PathName>  (for GET FilesetID) get the FilesetId stored in the
                    object identified by PathName. The fetched FilesetId
                    is placed in the Storage Area.

getnubfid or gnubfid
                    (for GET Name Using BitFileId ) gets one of possibly
                    many path names to the BitFileId that is currently
                    held in the Storage Area. Use the getbfid command to
                    load the Storage Area.

getnuoh or gnuoh    (for GET Name Using Object Handle) gets the path
                    name to the object described by the object handle in
                    the Storage Area. An object handle must be put into
                    the Storage Area before this command will work. Use
                    the 'getoh' command to fill the Storage Area.

getoh [<PathName>]  get the object handle of the object named by PathName.
                    If no PathName is supplied, get the object handle to
                    the current working directory. The object handle
                    will be fetched and placed into the Storage Area.
                    The contents of the Storage Area can be viewed with
                    the psa command.

getohffs <FilesetName> | [-id <FilesetId>]
                    (for GET Object Handle From Fileset) get the object
                    handle that is stored in the Fileset identified by
                    either FilesetName or FilesetId.  The user supplies
                    either a FilesetName or, optionally, a
                    FilesetId. To specify a FilesetId the -id parameter
                    must be used.
```

glob or globbing — when listing directory entries globbing is used and regular expression evaluation is not used. This is the default behaviour.

gnsc — (for Get Name Server Configuration) the Name Server's global configuration data will be fetched and displayed.

gnss — (for Get Name Server State) the Name Server's global administrative state data will be fetched and displayed.

gotofs <FilesetName> | [-id <FilesetId>] — (for GOTO FileSet) transfer to the root node of the Fileset identified by either FilesetName or FilesetId.The user supplies either a FilesetName or, optionally, a FilesetId. To specify a FilesetId the -id parameter must be used. The WorkingDir will become the root node of the indicated Fileset. This command modifies the Global FilesetId maintained by INSIF (see "setgfsid").

halt — will cause the Name Server to halt as quickly as possible.

help or h — send the help package to standard out.

inoregexp — (for I kNOw REGular EXPression syntax). The user is claiming to be a regular expression expert and there is no need to fix up any regular expression they might enter. The default behaviour is inoregexpnot.

inoregexpnot — (for I kNOw REGular EXPression NOT syntax). This poor person needs help. We will fix up his regular expression input as best we can. This is the default behavior.

ll [<PathName>] — (for Long List) will produce a long listing of the entries in the specified PathName. If no PathName is supplied, the long listing will be of the WorkingDir.

lfs [<HowMany> [<Offset>]] — (for List File Sets) causes Fileset information kept by the Name Server to be fetched and sent to standard out. If the lfs command is used without any parameters, all of the Fileset data is fetched and sent to standard out. The optional HowMany parameter is used to control how many Filesets are fetched and sent, and the Offset is a 64-bit FilesetId used to tell the Name Server where we wish to begin looking at the Fileset data.

```
lgfs [<HowMany> [<Offset>]]
                    (for List Global FileSets) the contents of the
                    global fileset file are fetched and sent to
                    standard out. If the lgfs command is used without
                    any parameters, all of the global fileset data is
                    fetched and sent to standard out. The optional
                    HowMany parameter is used to control how many
                    global fileset records are fetched and sent, and
                    the Offset is a 64-bit FilesetId used to tell the
                    Name Server where we wish to begin looking at the
                    global fileset data.

lj [<HowMany> [<Offset>]]
                    (for List Junctions) the pathnames to all of the
                    Junctions in the database are fetched and sent to
                    standard out. If the lj command is used without any
                    parameters, all of the Junction data is fetched and
                    sent to standard out. The optional HowMany
                    parameter is used to control how many Junction
                    records are fetched and sent, and the optional
                    Offset parameter is an RSN used to tell the Name
                    Server where to begin looking for Junction entries.

lock                the Name Server administrative state will be set to
                    'locked.'

ls [<PathName>] or lst [<PathName>] or dir [<PathName>]
                    will list the contents of PathName if PathName
                    was specified; otherwise, lists the contents of the
                    WorkingDir.


lsr [<PathName>] or lstr [<PathName>] or rls [<PathName>]
                    will recursively list the contents of the directory
                    specified by PathName.  If PathName is omitted the
                    recursive listing starts with the Working
                    directory.

make <n> dirs | files  [<BaseName>]
                    causes n directories or files to be created in the
                    current working directory. n can be any nuymber
                    greater than zero. If the optional BaseName
                    parameter is provifed, it will be used as the 'base
                    name' of the newly created entries.  If BaseName is
                    not provided, the new names will have the form
                    Dir.mmmmmmm or File.mmmmmmm where 0000001 <=
                    mmmmmmm <= 9999999.

mkbfid              will create a bogus BitfileId and put it into the
                    storage area.

mkdir <PathName>    will cause a directory named PathName to be
                    created in the directory described by the WorkingDir.
```

```
mkfile <PathName>   will cause a file named PathName to be created.
                    If INSIF is in UCAPI mode the file will be created
                    using a call to hpss_Open. If INSIF is in UCAPINOT
                    mode a file will be created using a call to
                    ns_Insert using any BitfileId that is in the
                    storage area. If no BitfileId is in the storage
                    area, none will be used.

mkfileset           will cause a Fileset to be made.  INSIF will prompt
                    for all the information needed.

mklink <FilePath> <HardLinkName>
                    will cause a hard link named HardLinkName to be
                    created in the WorkingDir.  FilePath is the path to
                    the file to be linked to.

mkjunction <PathName>
                    causes a Junction point named PathName to be
                    created. The object handle found in the storage
                    area is used as the FilesetHandle
                    (SubTreeHandle). This of course implies that the
                    storage area must first be filled with the 'getoh'
                    command.

mksymlink <SymLinkName> <SymLinkData>
                    will cause a symbolic link entry named SymLinkName
                    to be created in the directory described by the
                    WorkingDir.  SymLinkData is the data that is to be
                    associated with the symbolic link.

mksymlinkn <SymLinkName> <SpecialSymLinkData>
                    will cause a symbolic link entry named SymLinkName
                    to be created in the directory described by the
                    WorkingDir.  However mksymlinkn differs from
                    mksymlink in that it supports a 'special syntax'
                    for the SpecialSymLinkData.  Using this option it
                    is possible to create symbolic link data whose path
                    component lengths can be specified.  This is useful
                    for testing.  For example one could give the
                    SpecialSymLinkData path /<n>/<m>/<o> where n, m,
                    and o are decimal digits.  This will cause the
                    first path component to be n characters long, the
                    second will be m characters long, and the last will
                    be o.

noglob or noglobbing
                    globbing is not used when listing directory entries.
                    Regular expression evaluation is used. Globbing is
                    the default behaviour.

psa or showsa       (for Print Storage Area) will cause the contents of
                    the Storage Area to be displayed on standard out.
```

```
put <PathName> or insert <PathName>
                    put (insert) the BitfileId found in StoredBFId into
                    the directory described by PathName and the object
                    handle in WorkingDir.  The inserted entry will be
                    named PathName.

pwd                 (for Print Working Directory) will cause the object
                    handle in WorkingDir to be displayed on standard out.

quit, end, or exit  causes INSIF to terminate.

rbfid <FileName>    (for Read BitFileId) will cause the contents of file
                    FileName to be read into the BitfileId portion of
                    the Storage Area.

rdel <PathName>     recursively delete all of the entries found in
                    PathName and 'below'.  Note that PathName is also
                    deleted.  PathName must be a directory.

rdlink <PathName>   read the symbolic link data from entry PathName
                    and send this data to standard out.

reloadcache or rc   the Name Server will free all of the space currently
                    being used by the fileset cache and will reload the
                    cache by reading all of the fileset entries from
                    disk.

rename <PN1> <PN2>
                    will cause the Name Server to rename the item named
                    by PN1 (PathName 1) to the name found in PN2
                    (PathName 2).

reinitialize        will cause the Name Server to change its
                    administrative state to ST_REINITIALIZE.

repaired            will cause the Name Server to change its
                    administrative state to ST_REPAIRED.

roh <FileName>      (for Read Object Handle) will cause the contents of
                    file FileName to be read into the ObjHandle portion
                    of the Storage Area.

rsa <FileName>      (for Read into Storage Area) will cause the
                    contents the file FileName to be read into the
                    Storage Area.

rwd <FileName>      (for Read into Working Dir) will cause the contents
                    the file named FileName to be read into WorkingDir.


set [<PathName>] <FieldName> = <Value>
                    used to 'set' a FieldName equal to a Value in the
```

```
                    indicated Path Name.  PathName is optional and if
                    it is omitted, the FieldName in the current working
                    directory will be set. Note that certain attributes
                    can only be set by root. See the Interface
                    Specifications manual for a list which attributes
                    can be set by whom. The object attributes that can
                    be set are:
                        Account
                        Comment
                        GID
                        GroupPerms
                        LinkCount
                        Location
                        MACSecLabel
                        OtherPerms
                        SetGIDOnExe
                        SetStickyBit
                        TLR   (Time Last Read)
                        TLU   (Time Last Used)
                        TLW   (Time Last Written)
                        UID
                        UserPerms

                    The global parameters that can be set are:
                        DDP     (DirDefaultPerms)
                        FDP     (FileDefaultPerms)
                        MBSOB   (MaxByteSizeOfBuffer)
                        MPC     (MaxPathComponents)
                        RootIsGod
                        RootsUID
                        WariningThreshold
                        CriticalThreshold
                        WarningThresholdExceeded
                        CriticalThresholdExceeded
                        MaxRecords


                    there is a special syntax available for some of the
                    options-- the TLR, TLU, and TLW fields can be set
                    to either numeric values, or be set as follows--

                        set TLW = mm/dd/yy  hh:mm:ss

                    the permissions fields-- GroupPerms, OtherPerms, and
                    UserPerms can be set with either numeric values or
                    be set as follows--

                        set OtherPerms = rwxcid

    setacl [<PathName>]
                    will cause INSIF to prompt for ACL information and
                    then attach this collected ACL information to the
                    object named by PathName.  If no PathName is
```

```
                    supplied, the ACL information will be attached to
                    the current working directory.

setasrb             (for SET Administrative State RegisterBitmap) this
                    command is a special case of the 'set' command.  It
                    allows users to set specific bits in the
                    RegisterBitmap field of the Name Server's
                    hpss_server_attrib_t structure.  setasrb was
                    invented so that users don't have to input the
                    RegisterBitmap bits in hexadecimal.  The setasrb
                    command offers a menu of bits to choose from.  It's
                    very nice.


setbfid [<PathName>]
                    this is a special case of the 'set' command.  This
                    command causes the BitfileId found in the storage
                    area to be put into the entry named by PathName.
                    If no PathName is given the BitfileId will be
                    stored into the current WorkingDir.

setfs <FilesetName> | [-id <FilesetId>] <FieldName> = <Value>
                    (for SET Fileset) is used to 'set' fields in a
                    FilesetAttrs structure. Only the Root user can set
                    Fileset fields.  The following Fileset fields are
                    the only fields that can be set--
                        ClassOfService
                        FamilyId
                        FilesetHandle
                        FilesetId
                        FilesetName
                        FilesetType
                        GatewayUUID
                        StateFlags
                        SubSystemId
                        UserData
                        DirectoryCount
                        FileCount
                        HardLinkCount
                        JunctionCount
                        SymLinkCount

setfsarb <FilesetName> | [-id <FilesetId>]
                    (for SET FileSetAttrs RegisterBitmap) will prompt
                    for input that will cause bits to be turned 'on' in
                    the RegisterBitmap field of the FilesetAttrs record
                    indicated by FilesetName or FilesetId.

setfsufsid <FieldName> = <Value>
                    (for SET Fileset Using the stored FilesetId) This
                    command is the same as the setfs command except
                    that the user does not supply a FilesetId. The
                    FilesetId used to identify the Fileset record to be
```

```
                    'set' is taken from the StorageArea.

setgfsid <FilesetId>
                    (for SET Global FilesetID) a global FilesetId is
                    maintained by INSIF. This global FilesetId is used
                    as the input parameter to the Insert, MkDir,
                    MkLink, MkJunction, and MkSymLink APIs. This
                    command allows the user to change this global
                    value.

setreaddirbuffsize <n>
                    sets the ns_ReadDir buffer size to n where n is a
                    decimal integer.  <nk> is an acceptable syntax for
                    n (e.g. 64k).

setsabfidfhi <ASCII HexBitfileId>
                    (for SET StorageArea BitFileID From Hex
                    Input). directly sets the BitfileId field of the
                    StorageArea to the hexidecimal value represented by
                    the HexBitfileId.  For example,
                    0x1234abc789def.......

setsabfidfp         (for SET StorageArea BitFileID From Prompt).  NSDE
                    will prompt for the individual BitfileId fields.

setscrb             (for SET SpecificConfig RegisterBitmap) this
                    command is a special case of the 'set' command.  It
                    allows users to set specific bits in the
                    RegisterBitmap field of the SpecificConfig
                    structure.  setscrb was invented so that users
                    don't have to input the RegisterBitmap bits in
                    hexadecimal.  The setscrb command offers a menu of
                    bits to choose from.  It's pretty nice.


setuoh <FieldName> = <Value>
                    (for SET Using Object Handle) this command is the
                    same as the set command except that the object
                    whose attributes are being set is identified by the
                    object handle in the storage area. Use a command
                    like getoh to put an object handle into the storage
                    area.

shutdown            will cause the Name Server to perform a graceful
                    shutdown.

sizes               the sizes of several interesting data structures
                    related to the Name Server will be printed.

soh <PathName>      (for Show Object Handle) will fetch and display the
                    object handle associated with PathName.

statistics or stats
```

```
                    get and display the Name Server's statistics.

statisticsri or statri
                    get and display the Name Server's statistics AND
                    re-initialize the Name Server's statistics record.

storefsid <FilesetId>
                    store the given 64-bit FilesetId into the
                    StorageArea. This is a mechanism for directly
                    inserting a FilesetId, which may have come from a
                    cut-and-paste, into the StorageArea.

swsaandwd or swwdandsa
                    (for "SWap Storage Area object handle and Working
                    Directory object handle" or "SWap Working Directory
                    object handle and Storage Area object handle") the
                    object handles stored in these two locations are
                    swapped.

ucapi               (for Use Client APIs) causes INSIF to use the Client
                    API routines to delete files.  This the the default
                    behavior.

ucapinot            (for Use Client APIs NOT) causes INSIF to NOT use the
                    Client API routines to delete files.  INSIF will
                    send directly to the Name Server (using the Name
                    Server API routines) to delete objects.  The
                    default is to use the Client API routines.

umask <Cmask>       will call the Client API passing the file creation
                    mask.

unlock              will cause the Name Server to change its
                    administrative state to ST_UNLOCKED.

updacl [<PathName>]
                    update an ACL associated with PathName.  If no
                    PathName is supplied an ACL associated with the
                    current working directory will be updated.  INSIF
                    will prompt for the ACL information.

usercreds or uc     will cause the UID and GID from the current User
                    Credentials to be displayed.

usercreds <FN> = <V>
                    will cause the FN (Field Name) in the UserCreds
                    structure to be set to V (Value). The UserCreds
                    fields that can be set are
                       Name
                       UID
                       GID
                       SecLabel
                       DefAccount
```

```
                        CurAccount
                        DCECellId or Location or Loc
                        NumGroups
                        AltGroups or Groups

    wbfid <FileName>    (for Write BitFileID) the BitfileId found in the
                        Storage Area is written to file FileName.

    woh <FileName>      (for Write Object Handle) the object handle in the
                        Storage Area is written to file FileName.

    wsa <FileName>      (for Write from the Storage Area) will cause the
                        items in the Storage Area to be written to file
                        FileName.

    wwd <FileName>      (for Write from Working Dir) will cause the object
                        handle found in WorkingDir to be written to file
                        FileName.

    zapacls [<PathName>]
                        zap any and all ACL entries that may be associated
                        with the object named by PathName. If PathName is
                        omitted, the ACL entries are deleted from the
                        current directory.



The WorkingDirectory

    The WorkingDirectory is the directory that has last been cd'd to.  For
    example if a user types

        cd /RootDirectory/u21/minton/.nltss/fslib

    fslib becomes the WorkingDirectory.  INSIF internally stores the object
    handle to fslib and remembers this object handle as the WorkingDirectory.

    Many of INSIF's commands implicitly use the WorkingDirectory.  For
    example the commands

        del abc

    and

        get xyz

    delete and fetch entries from the WorkingDirectory.  The object handle
    to the WorkingDirectory can be viewed with the pwd command and can be
    written to a file with the wwd command.

User Credentials

    INSIF connects to the Name Server as a 'trusted' user.  This allows
```

INSIF to present the Name Server with User Credentials that the Name Server will believe and use. The ‘usercreds’ commands allow the user to change the various user credential fields thereby allowing the user to assume any identity. The current User Credentials can be viewed with the ‘usercreds’ command.

ENVIRONMENT Variables

INSIF uses several ENVIRONMENT variables to find names needed during initialization. If it cannot find these ENVIRONMENT variables it will terminate. The ENVIRONMENT variables used by INSIF can usually be found in the file ‘hpss_env.’ Korn shell users can simply source the hpss_env file as follows:

```
. hpss_env
```

Following are the ENVIRONMENT variables used by INSIF to connect to a Name Server running on the ripsaw machine in Livermore. Others should, of course, select these variables as appropriate for their own HPSS.

```
HPSS_CONFIG_SERVER=/.:/encina/server/sfs/hpss/serverconfig.4v1
HPSS_DESC_NS=Name Server
HPSS_CDS_NS=/.:/subsys/hpss/4v1_cns
HPSS_CONFIG_NS=/.:/encina/server/sfs/hpss/nsconfig.4v1
HPSS_DESC_SSM=SSM System Manager
HPSS_PRINCIPAL_SSM=hpss_ssm
HPSS_CDS_SSM=/.:/subsys/hpss/4v1_ssmsm
```

EXAMPLES

The following example will list the contents of the root directory, cd to a test directory, list the contents of that directory, cd to a directory owned by Donna, list the contents of that directory, display all the attributes of a file named RenamedFile, and then display the Name Server’s global state data.

```
insif

  Current setting: UCAPI (Using Client API).

/> lst

   Ty ObjId  FileId Cookie Name
    D 1      0      1      .
    D 1      0      2      ..
    D 3      0      3      CNSTestDir
    D 242    0      242    dir1
    D 1521   0      1521   dir2
    D 1522   0      1522   dir3
    D 1523   0      1523   dir4
    D 1524   0      1524   dir5
  / contains 8 entries.
```

```
/> cd CNSTestDir/abc002
/CNSTestDir/abc002> lst

   Ty ObjId  FileId Cookie Name
    D 118    0      118    .
    D 3      0      3      ..
    D 119    0      119    JimsRoot
    D 120    0      120    DonnasRoot
    D 121    0      121    ThisRoot
    D 122    0      122    ThatRoot
    D 123    0      123    WhichRoot
    D 124    0      124    WhatRoot
    D 125    0      125    SwitchRoot
    D 126    0      126    RootATootToot
    D 127    0      127    TheVeryLastRoot
  /CNSTestDir/abc002 contains 11 entries.

/CNSTestDir/abc002> cd DonnasRoot
/CNSTestDir/abc002/DonnasRoot> lst

   Ty ObjId  FileId Cookie Name
    D 120    0      120    .
    D 118    0      118    ..
    D 137    0      137    ADirectoryWithAVeryLongButInterestingName
    D 138    0      138    dir1
    D 139    0      139    dir2
    D 140    0      140    dir3
    F 1553   1553   1553   File97-06-03#19:47:41
  /CNSTestDir/abc002/DonnasRoot contains 7 entries.

/CNSTestDir/abc002/DonnasRoot> cd dir1
/CNSTestDir/abc002/DonnasRoot/dir1> lst

   Ty ObjId  FileId Cookie Name
    D 138    0      138    .
    D 120    0      120    ..
    F 160    160    160    RenamedFile
    S 185    0      185    SymLink1
    H 1488   1485   1488   abclink
    F 1646   1646   1646   File97-06-03#19:45:03
  /CNSTestDir/abc002/DonnasRoot/dir1 contains 6 entries.

/CNSTestDir/abc002/DonnasRoot/dir1> ga RenamedFile

  Here's the attributes for 'RenamedFile'--
    UID = 303     LinkCount   = 1       Type       = File
    GID = 456     MACSecLabel = 0       ACLOptions = <NONE>

    UserPerms  = RWXCID  CompPerms = RWXCID  FamilyId  = 0x00000000
    GroupPerms = RwXcid  Account   = 0x00000000
    OtherPerms = Rwxcid  Location  = 0
```

```
        GatewayUUID    = 00000000-0000-0000-0000-000000000000
        FilesetId      = 0x8000000000000000   FilesetType = 0 (LOCAL)
        DMHandle       = (No DMHandle)
        DMHandleLength = 0
        FilesetHandle  =
          (All zeros)

        SetGIDOnExe = false   SetStickyBit = false   SetUIDOnExe = false

        FileSize = 0x0000000000000400   ClassOfService = 199
        Comment  = ''
        TimeLastRead     = 36 (Wed Dec 31 16:00:36 1969)
        TimeLastWritten  = 123 (Wed Dec 31 16:02:03 1969)
        TimeLastUsed     = 456 (Wed Dec 31 16:07:36 1969)

        BitFileId:
           ObjectID      = 007dce0e-4881-1de1-933e-02608c2cefca
           ServerDep1    = 896489652
           ServerDep2    = 7
           ServerDep3    = 20525
           ServerDep4    = 97
           ServerDep5    = 98
           SecurityLevel = 0,0
           Reserved      = 0,0,0
           Type          = 4

    /CNSTestDir/abc002/DonnasRoot/dir1> gnss

      Here's the Administrative State data--
        Version                 = 1
        ServerID                = cc7cbc60-f198-11d1-b751-02608c2cd22f
        DescName                = SA Name Server
        ServerName              = /.:/subsys/hpss/ns_jim
        OperationalState        = ST_ENABLED
        UsageState              = ST_ACTIVE
        AdministrativeState     = ST_UNLOCKED
        ExecutionState          = ST_ACTIVE
        ServiceStatus           = STAT_NORMAL
        SecurityStatus          = STAT_NORMAL
        SoftwareStatus          = STAT_NORMAL
        HardwareStatus          = STAT_NORMAL
        CommunicationStatus     = STAT_NORMAL
        ThreadsAlarmThreshold   = 10
        ConnectionAlarmThreshold = 10
        RegisterBitmap          = 0x0000000000000000

    /CNSTestDir/abc002/DonnasRoot/dir1> end

     So Long...

SCCS
    man/insif.7, gen, 4va   11/23/98   14:08:55
```

I.26 loadhpssdmid — Load HPSS DMAPI IDs

NAME

loadhpssdmid - for Load HPSS DM Id's

SYNTAX

loadhpssdmid [-r] < <Input File>

where

-r — Optional command indicating the user is doing a recovery of a fileset rather than an initial import. If a recovery is being performed, then the -r option should be used.

When doing an initial import, it is necessary to mark the files as valid in HPSS; however, when doing a recovery this is not necessary. Thus when the -r option is omitted, the files will be marked as valid in HPSS. Using the -r option when performing recovery will increase the performance of this tool.

<Input File> — The name of the file containing information to be loaded into the HPSS fileset.

DESCRIPTION

loadhpssdmid (for Load HPSS DM ID's) is a utility used in the third step of a three step process to import an HPSS fileset into DFS.

This utility rumbles through an ASCII text file (produced by the 'loadtree' tool) setting the DM ID in each corresponding object in the HPSS Fileset.

ENVIRONMENT

The loadhpssdmid utility is initiated from the command line. The utility differs from some other HPSS utilities in that it uses the Client API library.

The user must have a dce_login as "hpss_dmg" and must have initialized the required Client API environment variables. The environment variables can be set by sourcing the $HPSS_ROOT/config/hpss_env file if these environment variables have been set up properly by the system administrator. See the HPSS Programmer's Reference Guide for more information on the required

```
    Client API environment variables.



INPUT FILE
    The loadhpssdmid utility reads records from stdin.  Users will
    probably want to redirect the output produced by the 'loadtree'
    utility to standard input.

    The input data is an ASCII representation of the DM ID for each
    object.

    The format for each entry:

    DMHANDLE: <length>-<handle>
    HPSS_ID: <hex dump of the BitFileId followed by the NS Object Handle>
    MTIME: (<decimal last time written>) [<text last time written>]



EXAMPLE

    The following example reads the ASCII text file 'load.DFS.out' and
    sets the DM ID's in the corresponding HPSS objects as part of an
    initial import of an HPSS fileset into DFS.

ksh
. $HPSS_ROOT/config/hpss_env
dce_login hpss_dmg
/usr/lpp/hpss/bin/loadhpssdmid < load.DFS.out
Processing entry '0'
Processing ended normally
    Objects Processed:                  (4)
        Objects loaded properly:        (4)
        Object read error:              (0)
            GetDMHandle errors:         (0)
            GetHPSSID errors:           (0)
        Object load error:              (0)
            Entries not found:          (0)
            Other errors:               (0)



SCCS
    man/loadhpssdmid.7, gen, 4va   11/20/98   16:44:42
```

I.27 loadhpssfs — Load HPSS Fileset

```
NAME
    loadhpssfs - for Load HPSS Fileset



SYNTAX
    loadhpssfs <Fileset Name> <Input File> <Output File> [-r]

    where

    <Fileset Name>  The name of the HPSS Mirrored Fileset to load.

    <Input File>    The name of the file containing the ASCII text
                    to be loaded into the HPSS fileset.

    <Output File>   The name of the file that will contain the Output text.

    -r              Optional command indicating to restart the load
                    where the previous load left off.  This option may
                    be used if a previous run of “loadhpssfs”
                    terminated prematurely.  Note:  The Restart option
                    assumes that the fileset being loaded has NOT been
                    altered since the last load attempt.  Unpredictable
                    results may occur if the fileset has been altered.



DESCRIPTION
    loadhpssfs (for Load HPSS Fileset) is a utility used in the second
    step of a three step process to import a DFS fileset into HPSS.

    The loadhpssfs utility rumbles through an ASCII file, and loads
    each object into an HPSS Mirrored Fileset.  The file may have been
    produced by either the hdmdump or dumphpssfs utility.  The
    loadhpssfs utility produces an ASCII text file that will be used
    later by the loadhpssid utility to load the new HPSS IDs back into
    DFS.  The output file is also read by loadhpssfs itself for
    reprocessing objects.

    The HPSS Fileset to be loaded must already exist and be of Type
    ‘Mirrored’.  See the HPSS Administration Guide for more information
    on filesets.

    Use of the [-r] option causes the checkpoint/restart log file,
    LoadHPSSFSRestartFile, to be read.  The LoadHPSSFSRestartFile file
    is created in the current working directory during the initial run
    of “loadhpssfs”.  The checkpoint/restart log file contains the file
    byte offset followed by the pathname of every 5th record read from
    the <Input File> and created in the new HPSS fileset.  When the
```

[-r] option is used, loadhpssfs reads the last logged checkpoint, and continues from this point. It will probably be the case that items already created will be retried. If this occurs, the application will generate a "Duplicate load" warning message for each object that has already been created and then continue. Because an object may have been created, but not all of it's attributes may have been set up correctly, "loadhpssfs" always sets the attributes of existing objects. "loadhpssfs" will generate a "We're now working on loading NEW objects" message when it has completed processing of the duplicates. Duplicate loads will cause additional (duplicate) records to be added to the <Output File>. This will merely cause the tool that reads this file (loadhpssid) to harmlessly load the same object's HPSS ID twice.

HPSS can be configured for either unix-style or site-style accounting. Currently the DFS dumping tool, hdmdump, puts the string "unknown" into each object's ACCOUNT field. 'loadhpssfs' processes an "unknown" ACCOUNT for an object differently depending on whether the UID of the object is local or foreign. For objects with local UID's, loadhpssfs will use the accounting associated with that particular UID. For objects with foreign UID's, loadhpssfs assumes unix-style accounting therefore uses the UID as the ACCOUNT. It then displays a WARNING message for the object. For objects with ACCOUNT values indicated in the <Input File>, loadhpssfs will set the account attribute to the value specified in the <Input File>. See the HPSS Administration Guide for more information on Accounting.

The Family ID for each new object will be the File Family ID of the fileset it is being loaded into. See the HPSS Administration Guide for more information on File Family.

The Class of Service chosen for each file is based on HIGHLY_DESIRED_PRIORITY file size hints. See the HPSS Administration Guide for more information on Class Of Service.

ENVIRONMENT

The loadhpssfs utility is initiated from the command line. The utility differs from some other HPSS utilities in that it uses the Client API library.

The user must have a dce_login as "hpss_dmg" and must have initialized the required Client API environment variables. The environment variables can be set by sourcing the $HPSS_ROOT/config/hpss_env file if these environment variables have been set up properly by the system administrator. See the HPSS Programmer's Reference Guide for more information on the required Client API environment variables.

ASSOCIATED FILES

```
The loadhpssfs utility uses three files: an input file specified on
the command line as <Input File>, an output file specified on the
command line as <Output File>, and a restart file, called
LoadHPSSFSRestartFile, which is created in the current working
directory.

The first file name is specified on the command line as <Input
File>.  This file contains the ASCII representation of a fileset.
The dumphpssfs and hdmdump utilities produce this file.

The format for each object in the <Input File> is:

====================
LEVEL: <level>
TYPE: <type string>
NAME: <object name>
DMHANDLE: <length>-<handle>
EVENTS: 0
NUMEVENTS: 0
PERS: 0
PERSMAN: 0
DTIME: (<decimal current time>) <text current time>
CHANGE: 0
DEV: 0
INO: 1
MODE: <mode>
NLINK: <number of links>
UID: <user id>
GID: <group id>
RDEV: 0
SIZE: <filesize>
ATIME: (<decimal last time read>) <text last time read>
MTIME: (<decimal last time written>) <text last time written>
CTIME: (<decimal last time metadata update>) \
                              <text last time metadata update>
BLOCKSIZE: <blocksize>
NUMBLOCKS: <number of blocks>
FSID: <fileset number in hex>
ACCOUNT: <account number - if unix-style accounting should be UID>
COS: <class of service>
COMMENT: <comment>
FAMILYID: <family id>
LOCATION: <location (0 means local)>
MACSECLABEL: <macseclabel>
ACL List: <number of ACLs> OR <default>
    For each ACL when not “default”:
        ENTRYTYPE: <ACL Entry Type (eg, user_obj, group_obj)>
        PERMS: <permissions>
        EXPDATE: <expiration date>
        ENTRYID: <Entry Id>
        LOCATION: <Location (0 if local)>
DM Attributes:
    HPSS_ID = <hex dump of the BitFileId followed by the Object Handle>
```

```
        if SYNC_FILESET_DATA Consistency Flag is set: SYNCFLAG
        if CACHE_DATA_VALID Consistency Flag is set: CAC_VAL
        if HPSS_DATA_VALID Consistency Flag is set:  HPS_VAL
    if “file”
        if “hardlink”: HARDLINK: YES
                       LINKPATH: <pathname>
        else
                       HARDLINK: NO
    if “junction”
        JUNCTIONTYPE: regular
        JUNCTIONPATH: <Fileset Name> [<pathname>]
    if “symlink”
        SYMLINK: <pathname>
    ====================
```

A second file is specified on the command line as <Output File>. This file will contain ASCII representations of certain object attributes. This file will be used as input to the ‘loadhpssid’ utility. This file will also be read by loadhpssfs to reprocess all the newly created HPSS directories to set their MTIME (Time Last Written).

The format for each entry in the <Output File> is:

```
DMHANDLE: <length>-<handle>
HPSS_ID: <hex dump of the BitFileId followed by the NS Object Handle>
MTIME: (<decimal last time written>) <text last time written>
```

A third file, LoadHPSSFSRestartFile, is created in the current working directory. This file is used as a checkpoint/restart file. The file byte offset followed by the pathname of every 5th record read from the <Input File> is written to this file. This information is needed for [-r], the Restart feature. Upon Restart, this utility continues from the last logged record. (The application will take into account duplicate loads. See above for more details.) Therefore, all restarts should be executed in the same directory as the initial load. After a successful load, this file can be deleted.

The format for each log entry in LoadHPSSFSRestartFile is:

```
<file byte offset> <pathname>
```

EXAMPLE

The following example reads the ASCII text representation of a fileset from file ‘dump.out’ and loads it into the HPSS mirrored

```
    fileset named MyNewFileset.


ksh
. $HPSS_ROOT/config/hpss_env
dce_login hpss_dmg
/usr/lpp/hpss/bin/loadhpssfs MyNewFileset dump.out load.out
Using file 'LoadHPSSFSRestartFile' for Checkpoint/Restart capabilities.
Processing entry '0'
Loading=./.
Loading=./TestDir
Loading=./TestDir/abcDir
Loading=./TestDir/abcDir/ABCfile
Loading=./TestDir/abcfile
Processing ended normally
Re-Processing entry '0'
Re-Processed 4 entries.
Number of objects processed:   (5)
        Files:                 (2)
        Directories:           (3)
        Hard Links:            (0)
        Symbolic Links:        (0)
        Junctions:             (0)
        EEXIST Objects:        (0)



SCCS
    man/loadhpssfs.7, gen, 4va   11/20/98   16:43:52
```

I.28 loadhpssid — Load the HPSS ID for files in a mirrored fileset

```
NAME
     loadhpssid - Load the HPSS ID for files in a mirrored fileset.

SYNTAX
     loadhpssid < <Input File>

DESCRIPTION
     loadhpssid is used to load into DMLFS files the ID of the HPSS file
     that is used to back its directory and file data. This tool will
     run with input coming from the output generated by the tool
     “loadhpssfs”. This tool will be used during DFS fileset import into
     a mirrored fileset on HPSS. This tool should only be used during
     mirrored fileset imports.

     This tool must be run as root on the machine that contain the disk
     on which the fileset resides.

     See “HPSS Administration Guide: DFS Configuration and Management”
     for more details.

PARAMETERS

     Input File - This is the file generated by the tool “loadhpssfs”
                  during the import of the DFS fileset into HPSS. The file
                  contains DM handles to HPSS ID mappings.

EXAMPLE
     The following command will load the HPSS ID’s into the DFS files
     that mirror the HPSS files.

     loadhpssid < loadhpssfs.output

FILES


SEE ALSO
     loadhpssfs(7), loadtree(7), loadhpssid(7), loadhpssdmid(7),
     dumphpssfs(7)

SCCS
     man/loadhpssid.7, gen, 4va 11/20/98 15:23:07
```

I.29 loadtree — Load DFS with files that mirror a fileset

```
NAME
     loadtree - Load DFS with files that mirror a fileset

SYNTAX
     /usr/lpp/hpss/dmapi/dmapitools/loadtree/loadtree < <Input File> \
                                                     > <Output File>

DESCRIPTION
     loadtree is used to load a representation of an HPSS fileset into
     DFS. This tool can be used to import an HPSS fileset into DFS, and
     it can be used to recover a DFS mirrored fileset from a disk crash.

     This tool must be run as root on the machine that contain the disk
     on which the fileset resides. It also must be run from the root
     of the local mount point for the fileset.

     loadtree uses the output from the utility dumphpssfs as its input.
     loadtree sends output to stdout that maps the DM handle for the new
     DFS objects built into the mirrored HPSS object that they
     represent.

     See “HPSS Administration Guide: DFS Configuration and Management”
     for more details.

PARAMETERS

     Input File - This is the file generated by the tool “dumphpssfs”
                  as output and is an ascii representation of the HPSS
                  fileset being imported into DFS or being recovered.

     Output File - loadtree generates output to stdout which is an ascii
                   representation of the DM handle to HPSS ID mapping of
                   objects mirrored in the fileset.

EXAMPLE
     The following command will load DFS with a mirror of an HPSS
     fileset using the output from dumphpssfs that was stored in
     dumphpssfs.out. It will generate an output file to be used by
     loadhpssdmid to load DM handles into HPSS that correspond to
     the mirrored object in DFS. It is assumed that loadtree is being
     run by root, and the current working directory is the local mount
     point to the root of the fileset to be mirrored.

     /usr/lpp/hpss/tools/dmapi/dmapitools/loadtree/loadtree \
                                       < dumphpssfs.out \
                                       > loadtree.out


FILES
```

```
SEE ALSO
     loadhpssfs(7), loadtree(7), loadhpssid(7), loadhpssdmid(7),
     dumphpssfs(7)

SCCS
     man/loadtree.7, gen, 4va 11/20/98 15:23:40
```

I.30 lsacl — HPSS List ACL Utility

```
NAME
     lsacl - HPSS List ACL Utility

SYNTAX
     lsacl [ -f Object Path Name ]

DESCRIPTION
     "lsacl" is a HPSS User Utility.  This utility is used to list
     access control information associated with a specific pathname.
     The output from lsacl is in the correct format for input
     to chacl(change acl).  The output from lsacl for an object ACL
     consists of two sections: the comments section and the ACL
     entries section.  The comments section contains three lines:
     name of the object, the object owner, and the object
     owning group.  The ACL entries section contains lines with three
     fields seperated by a colon. The first field contains the ACL entry
     tag type.  The second field contains the ACL entry qualifier.  The
     third field contains the access permissions.


  Output Format:


     #file:<filename>
     #owner:<uid>
     #group:<gid>
     user::<perm>
     mask::<perm>
     user:<uid>:<perm>
     user:<uid2>:<perm>
     group::<perm>
     group:<gid>:<perm>
     group:<gid2>:<perm>
     other::<perm>

The types of permissions are:
        r - read
        w - write
        x - execute
        i - insert
        d - delete
        c - control

        The utility lsacl is initiated from the command line.
        The pathname is the key input value.   The utility differs
        from some other HPSS utilities in that it uses the client
        api library.  The user must have a dce_login and two
        environment variables must be set.  The environment variables
        for the CDS names of the NameServer and the Bitfile Server are:
```

```
        HPSS_HPNS_NAME and HPSS_BFS_NAME.


OPTIONS
     -f <Object Path Name> The HPSS path name of the object whose ACL's
                           are to be listed.



EXAMPLE
     The following steps are used to list the ACL's associated with the
     directory /ampex owned by the HPSS user hpss_client1.

     dce_login hpss_client1

     export HPSS_LS_NAME=/.:/hpss/ls

     lsacl -f /ampex         {input}
     #file: /ampex           {output}
     #owner:1006
     #group:system
     user::rwx---
     mask::r-x---
     group::r-x---


FILES


SEE ALSO
     chacl(7)

SCCS
     man/lsacl.7, gen, 4va   11/23/98   14:01:55
```

I.31 lsfilesets — List all of the filesets that are being managed by the Name Server

```
NAME
     lsfilesets - list all of the HPSS fileset's.

SYNTAX
     lsfilesets

DESCRIPTION

     lsfilesets will list all the fileset's that are being managed by
     the Name Server.


OPTIONS

EXAMPLE
     The following command will list out all the filesets. You must be dce
     logged in to issue this command.

     lsfilesets

     Output will have the form:

     Fileset Name:     ball.archive
     Fileset ID:       0.22
     Gateway UUID:     2edf7f6a-def1-11d1-a574-02608c2cd22f
     Name Server UUID: 5a475c92-deef-11d1-a574-02608c2cd22f
     Class Of Service: 0
     Family ID:        0
     Fileset Type:     ARCHIVED

     Fileset Name:     ball.mirror
     Fileset ID:       0.34
     Gateway UUID:     2edf7f6a-def1-11d1-a574-02608c2cd22f
     Name Server UUID: 5a475c92-deef-11d1-a574-02608c2cd22f
     Class Of Service: 0
     Family ID:        0
     Fileset Type:     MIRRORED

     Fileset Name:     DebbieTest2
     Fileset ID:       3272205073.4240182136
     Gateway UUID:     00000000-0000-0000-0000-000000000000
     Name Server UUID: 5a475c92-deef-11d1-a574-02608c2cd22f
     Class Of Service: 0
     Family ID:        0
     Fileset Type:     HPSS_ONLY

     Number of Filesets Found:   3
```

```
FILES


SEE ALSO

SCCS
     man/lsfilesets.7, gen, 4va 11/20/98 15:24:04
```

I.32 lshpss — List Information About HPSS

```
NAME
     lshpss - List information about HPSS

SYNTAX
     lshpss [options ...]

DESCRIPTION
     This utility displays various HPSS resources like Class of Service,
     Hierarchy, and Storage Class.

     Before running this script, you must be authorized to access
     the SFS files.  This can be done by typing “dce_login”
     with “encina_admin” or one of the HPSS servers like “hpss_ssm”
     as an argument.

OPTIONS

   -cos              Show Class of Service list
   -hier             Show Hierarchy list
   -sc               Show Storage Class list
   -migp             Show Migration Policies
   -purgep           Show Purging Policies
   -vol              Show Physical Volumes
   -dev              Show Mover Devices
   -drv              Show PVL Drives
   -svr              Show HPSS Servers
   -mvr              Show HPSS Movers
   -acct             Show Accounting Policies
   -logp             Show Log Policies
   -locp             Show Location Policies
   -ffam             Show File Families
   -bfs              Show Bitfile Servers
   -ns               Show Name Servers
   -pvr              Show PVRs
   -logd             Show Log Daemons
   -logc             Show Log Clients
   -nfsd             Show NFS Daemons
   -listmeta         List metadata
   -all              Display all of the above

   -dumpmeta         Dump all metadata for HPSS to separate files
   -dump             Dump one metadata file
   -sdt              Go into an sdt shell
   -h                Show this help message


EXAMPLE
   The following command will list all the resources :
```

```
   $ dce_login hpss_ssm
   $ lshpss -all

   To list classes of service and purge policies :

   $ dce_login hpss_ssm
   $ lshpss -cos -purgep

SEE ALSO
     sdt.hpss

SCCS
     man/lshpss.7, gen, 4.1.1, 4.1.1.1   6/4/99   14:39:54
```

I.33 lsjunctions — List all of the junctions that are being managed by the Name Server

```
NAME
     lsjunctions - list all of the junctions that are being managed by
                   the Name Server.

SYNTAX
     lsjunctions

DESCRIPTION

     lsjunctions will list all the junctions, and their associated
     filesets, that are being managed by the Name Server. Since
     multiple junctions can be linked to a single fileset, it is
     possible to have similar listings (except for the junction name)
     for different junctions that point at the same fileset.

     The user must be DCE logined in as hpss_ssm before executing the
     utility.

OPTIONS

EXAMPLE
     The following command will list out all the junctions.

     lsjunctions

     Output will have the form:

     Junction Name:    FilesetRoot.2546: ./home/hpss/fs_hpss
     Fileset Name:     File Family FS
     Fileset ID:       2178027353.1221831000
     Gateway UUID:     00000000-0000-0000-0000-000000000000
     Name Server UUID: eff42414-8de6-11d1-9712-08005abaaca9
     Class Of Service: 8
     Family ID:        300
     Fileset Type:     HPSS_ONLY

     Junction Name:    FilesetRoot.2546: ./home/hpss/fs_6a139
     Fileset Name:     fs_6a139
     Fileset ID:       0.139
     Gateway UUID:     434d8292-8de5-11d1-9712-08005abaaca9
     Name Server UUID: eff42414-8de6-11d1-9712-08005abaaca9
     Class Of Service: 5
     Family ID:        0
     Fileset Type:     ARCHIVED

     Junction Name:    FilesetRoot.2546: ./home/hpss/fs_6m271
     Fileset Name:     fs_6m271
     Fileset ID:       0.271
```

```
     Gateway UUID:     434d8292-8de5-11d1-9712-08005abaaca9
     Name Server UUID: eff42414-8de6-11d1-9712-08005abaaca9
     Class Of Service: 0
     Family ID:        1
     Fileset Type:     MIRRORED

     Junction Name:    FilesetRoot.2546: ./home/hpss/empty_junction
     Fileset Name:     <No corresponding fileset>

     Number of Junctions Found:   4

FILES


SEE ALSO
    The crtjunction, deljunction and lsfilesets utilities

SCCS
     @(#)89   1.1   man/lsjunctions.7, gen, 4va   11/20/98   15:35:56”
```

I.34 lsrb — List Files Last Accessed Before

The LSRB utility uses a command line interface, and the definition is as follows:

NAME

lsrb List of files, grouped by volume and sorted by date, that have a last access date prior to a specified date.

SYNTAX

```
lsrb   yyyymmdd [-c <CDS NAME>] [-p <Principal Name>] [-k <Keytab Name]
                [-h <Generic Name>] [-s <Sleep Value>]
```

DESCRIPTION

LSRB utility is a HPSS System Maintenance Utility. It is used to generate a list of files that have a last access date prior to the specified date. The list of files will be grouped by volume, and sorted in ascending order (oldest file first). This can be useful when identifying files that are candidates for deletion because they have not been accessed for a long time. A date must be provided to the LSRB utility in the yyyymmdd format (e.g. 19980131 -- January 31, 1998)).

OPTIONS

-c <CDS Name>	The Cell Directory Service name which this utility will run under. Usually run as SSM. The Server CDS name of the SSM can be found from SSM through: Admin, Configure HPSS, Server Configuration screens.
-p <Principal Name>	The server principal name which this utility will run under. Usually run as SSM. The Principal name of the SSM can be found from SSM through: Admin, Configure HPSS, Server Configuration, Security Information screens.
-k <Keytab Name>	The Keytab name which this utility will run with. Usually run as SSM. TheKeytab name of the SSM can be found from SSM through: Admin, Configure HPSS, Server Configuration, Security Information screens.
-h <Generic Name>	The pathname of the SFS generic server configuration file.
-s <Sleep Value>	The Sleep Value is used to throttle back the utility. It represents the number of seconds to sleep between physical volumes to reduce the utility’s load on the CPU.

EXAMPLE

```
     The following command will list files, that were accessed prior to
     January 01, 1996.  The list of files will be sorted and grouped by
     storage server and volume.

     lsrb 19960101 -c /.:/hpss/ssmsm -p hpss_ssm -k /krb5/hpss.keytabs \
                   -h /.:/encina/sfs/hpss/serverconfig -s 5

FILES

     None.

SEE ALSO

     lsvol(7)

SCCS

     @(#)44   1.1   man/lsrb.7, gen, 4.1.1, 4.1.1.1   4/7/99   14:27:14
```

I.35 lsvol — List Pathname of Files with Segment on Volume

```
NAME
     lsvol - List pathname of files with segment on volume.

SYNTAX
     lsvol [ -v Physical Volume Name -c Utility Server(CDS) Name -p Utility
     Principal Name -k KeyTab Path -h HPSS Generic Server Path -s SS
     Generic Descriptive name -b BFS Generic Descriptive name
     -n Name Server Descriptive Name -f ]

DESCRIPTION
     “lsvol” is a HPSS System Maintenance Utility. It is used
     to list the pathnames of files which have a segment on the
     input Storage Server Physical Volume.  This can be usefull
     when a cartridge is damaged and a list of files affected
     is needed.  It also has an additional feature.  If a segment
     is found which does not have a file associated(orphan segment)
     with it, lsvol will prompt the user to remove or allow the
     segment to remain.  An option to force the removal of orphan
     segments without prompting is also available.



OPTIONS
     -v <SS Vol. Name>     The Storage Server Physical Volume
                           Name of the Volume you to scan.

     -s <SS Desc. Name>    The descriptive name of the HPSS Storage Server
                           which manages the Volumes to be scanned.
                           The descriptive name of a storage server
                           can be found from SSM through: Admin, Configure
                           HPSS, Server Configuration screens.

     -b <BFS Desc. Name>   The descriptive name of the HPSS Bitfile Server
                           which manages the bitfiles to be scanned.
                           The descriptive name of a bitfile server
                           can be found from SSM through: Admin, Configure
                           HPSS, Server Configuration screens.

     -n <CNS Desc. Name>   The descriptive name of the HPSS Name Server
                           which manages the pathnames to be scanned.
                           The descriptive name of a Name server
                           can be found from SSM through: Admin, Configure
                           HPSS, Server Configuration screens.

     -h <Generic Name>     The pathname of the SFS generic server configuration
                           file.

     -c <CDS Name>         The Cell Directory Service name which this
```

```
                          utility will run under.  Usually run as SSM.
                          The Server CDS name of the SSM
                          can be found from SSM through: Admin, Configure
                          HPSS, Server Configuration screens.

     -p <Principal Name>  The server principal name which this utility
                          will run under.  Usually run as SSM.
                          The Principal name of the SSM
                          can be found from SSM through: Admin, Configure
                          HPSS, Server Configuration, Security Information
                          screens.

     -k <Keytab Name>     The Keytab name which this utility
                          will run with.  Usually run as SSM.
                          The Keytab name of the SSM
                          can be found from SSM through: Admin, Configure
                          HPSS, Server Configuration, Security Information
                          screens.

     -f                   The force option will allow the deletion of
                          orphan SS storage segments without testing
                          for an active length.  The deletes will be
                          done automatically without prompt.

EXAMPLE
     The following command will list the pathnames of files with
     segments on the physical volume BUD13100. The descriptive
     names of the servers are: ss_tape(storage server), bfs(bit file
     server) and cns(Name Server).


     $ lsvol   -v BUD13100 -h /.:/encina/sfs/hpss/serverconfig
               -c /.:/hpss/ssmsm -p hpss_ssm -k /krb5/hpss.keytabs
               -s ss_tape -b bfs -n cns

     Connected to CNS, rc = 0
     Connected to SS, rc = 0
     Connected to BFS, rc = 0
     FilesetRoot.2546: ./home/smith/test56781
     FilesetRoot.2546: ./home/smith/test5678

     Number of VV Segments found 2
     Number of Identical Bitfile ID Owner Segments found 0
     Number of File Pathnames found 2
     Number of File Pathnames get errors 0

FILES


SEE ALSO
     dumpbf(7), dumppv_pvr(7), dumppv_pvl(7)

SCCS
```

```
man/lsvol.7, gen, 4va   11/23/98   12:49:45
```

I.36 managesfs — Manage HPSS Metadata Files Utility

NAME

managesfs - Manage HPSS Metadata Files Utility

SYNTAX

managesfs

DESCRIPTION

The “managesfs” utility is used primarily during initial HPSS configuration to create metdata files using the Encina Structure File Server (SFS). For each metadata file it creates, it defines the appropriate metadata fields, index keys, and access control permissions. The utility can also be used to query, empty, and destroy metadata files. Before invoking the tool, the user must be DCE logged in as a principle with “administer”, “create”, and “inquire” permissions for the corresponding SFS server.

WARNING: USING THIS UTILITY, A USER WITH APPROPRIATE PERMISSIONS CAN EASILY EMPTY AND/OR DELETE HPSS METADATA INFORMAITON, WHICH MAY RENDER AN EXISTING PRODUCTION HPSS ENVIRONMENT USELESS. ONCE METADATA IS EMPTIED OR DELETED, IT CANNOT BE RECOVERED EXCEPT BY USING BACKUP FILES PREVIOUSLY CREATED. USE THE EMPTY AND DELETE OPTIONS WITH EXTREME CARE.

Environment Variables

SFS_VOL	If set, this variable is used as the default SFS data volume name.
ENCINA_SFS_SERVER	If set, this variable is used as the default SFS server name. The variable will be set to the latest SFS server name value specified while executing the utility.

Interactive Options

When started, a prompt will appear to select/verify the SFS server name. This will be followed by a menu of options, which are explained below.

1 - Create file

Use this option to create one or more HPSS metadata files. The list of possible metadata file types will then be displayed. To create a single file, select the corresponding number. To create all files, select the last option.

NOTE: Files created with this option will be defined on the SFS data volume shown by option 7 on the main menu. Be sure to verify and/or change the data volume name prior to creating any metadata files.

If a single file was selected to be created, a prompt will appear for

the SFS filename to create, with an appropriate default value. A metadata file will then be created with this filename on the current SFS data volume setting, along with all corresponding indices. The file permissions are also set to allow the group “hpss_server” (the DCE group under which all HPSS servers run) to delete, exclusively-open, insert, inquire, read, and update records.

2 - Empty file

Use this option to empty an existing HPSS metadata file. Emptying a file removes all records in the file.

WARNING: USE THIS OPTION WITH EXTREME CARE. THE METADATA DELETED USING THIS OPTION CAN ONLY BE RECOVERED USING PREVIOUS SFS BACKUPS.

The list of possible metadata file types will then be displayed. After selecting the appropriate type of file, a prompt for the actual filename is presented, along with an appropriate default value.

3 - Destroy file

Use this option to delete an existing HPSS metadata file. Destroying a file removes all records in the file as well as the definition of the file itself.

WARNING: USE THIS OPTION WITH EXTREME CARE. THE METADATA DELETED USING THIS OPTION CAN ONLY BE RECOVERED USING PREVIOUS SFS BACKUPS.

The list of possible metadata file types will then be displayed. After selecting the appropriate type of file, a prompt for the actual filename is presented, along with an appropriate default value.

4 - Query file

Use this option to query an existing HPSS metadata file. The information presented from this option includes the actual metadata fields defined for an SFS file, the number of records in the file, storage-space information, etc.

The list of possible metadata file types will then be displayed. After selecting the appropriate type of file, a prompt for the actual filename is presented, along with an appropriate default value.

5 - List all files

Use this option to list all files defined in the current Encina SFS server.

6 - Change SFS server

Use this option to define which Encina SFS server with which to communicate. After changing the SFS server name, all subsequent SFS operations will be directed to that SFS server.

```
7 - Change SFS volume

    Use this option to define which Encina SFS data volume with which to
    create new SFS files.

8 - Change SFS filename extension

    Use this option to specify a standard filename extension for use in
    creating new SFS files.  While used primarily during HPSS
    development, it allows administrators to create a completely
    different set of HPSS metadata files with the same SFS server by
    using a unique filename extension.
```

EXAMPLE

The following example shows how to create a “bitfile” metadata file on SFS data volume “sfsVol1” managed by SFS server “/.:/encina/sfs/hpss”:

```
% managesfs
Enter SFS Server Name [/.:/encina/sfs/hpss]:

Available Operations:

    1 - Create file
    2 - Empty file
    3 - Destroy file
    4 - Query file
    5 - List all files
    6 - Change SFS server (currently “/.:/encina/sfs/hpss”)
    7 - Change SFS volume (currently “sfsVolhpss”)
    8 - Change SFS filename extension (currently <none>)

Select operation (<RETURN> to exit)> 7

Enter SFS Volume [sfsVolhpss]: sfsVol1

Available Operations:

    1 - Create file
    2 - Empty file
    3 - Destroy file
    4 - Query file
    5 - List all files
    6 - Change SFS server (currently “/.:/encina/sfs/hpss”)
    7 - Change SFS volume (currently “sfsVol1”)
    8 - Change SFS filename extension (currently <none>)

Select operation (<RETURN> to exit)> 1


Available Metadata Objects
```

```
 1 -   General Server Configurations  32 -   Migration/Purge Configurations
 2 -   Accounting Policies            33 -   Migration Policies
 3 -   Account Log Records            34 -   Purge Policies
 4 -   Account Summary Records        35 -   Migration/Purge Checkpoints
 5 -   Account Snapshot Records       36 -   Mount Daemon Configuration
 6 -   Migration Records              37 -   Mover Configurations
 7 -   Purge Records                  38 -   Mover Devices
 8 -   BFS Configurations             39 -   NFS V2 Daemon Configuration
 9 -   Bitfiles                       40 -   PVL Configurations
10 -   Bitfile COS Changes            41 -   PVL Activities
11 -   Bitfile Tape Segments          42 -   PVL Drives
12 -   Bitfile Disk Segments          43 -   PVL Jobs
13 -   Bitfile Disk Allocation Maps   44 -   PVL Physical Volumes
14 -   BFS Storage Segment Checkpoint 45 -   PVR Configurations
15 -   BFS Storage Segment Unlinks    46 -   3494 Cartridges
16 -   Classes of Service             47 -   3495 Cartridges
17 -   DMAP Gateway Configurations    48 -   AML Cartridges
18 -   DMAP Gateway File Sets         49 -   Operator Cartridges
19 -   File Families                  50 -   STK Cartridges
20 -   Hierarchies                    51 -   Remote Site Configurations
21 -   NS Configurations              52 -   Storage Server Configurations
22 -   NS ACL Extensions              53 -   Storage Classes
23 -   NS Fileset Attrs               54 -   SS Disk Storage Maps
24 -   NS Global Filesets             55 -   SS Tape Storage Maps
25 -   NS Objects                     56 -   SS Disk Storage Segments
26 -   NS Text Extensions             57 -   SS Tape Storage Segments
27 -   Log Client Configurations      58 -   SS Disk Physical Volumes
28 -   Log Daemon Configurations      59 -   SS Tape Physical Volumes
29 -   Log Policies                   60 -   SS Disk Virtual Volumes
30 -   Location Server Policies       61 -   SS Tape Virtual Volumes
31 -   Metadata Monitor Configuration 62 -   <All of the above>

  Select type of file to create (<RETURN> for Main Menu)> 4

  SFS filename [bitfile]:

  Creating file bitfile...
```

FILES

The following default SFS filenames are used by managesfs:

File Number/Description	Default SFS Filename
1 - General Server Configurations	serverconfig
2 - Accounting Policies	accounting
3 - Account Log Records	acctlog
4 - Account Summary Records	acctsum
5 - Account Snapshot Records	acctsnap
6 - Migration Records	bfmigrec

```
 7 - Purge Records                     bfpurgerec
 8 - BFS Configurations                bfs
 9 - Bitfiles                          bitfile
10 - Bitfile COS Changes               bfcoschange
11 - Bitfile Tape Segments             bftapesegment
12 - Bitfile Disk Segments             bfdisksegment
13 - Bitfile Disk Allocation Maps      bfdiskallocrec
14 - BFS Storage Segment Checkpoint    bfsssegchkpt
15 - BFS Storage Segment Unlinks       bfssunlink
16 - Classes of Service                cos
17 - DMAP Gateway Configurations       dmg
18 - DMAP Gateway File Sets            dmgfileset
19 - File Families                     filefamily
20 - Hierarchies                       hierarchy
21 - CNS Configurations                nsconfig
22 - NS ACL Extensions                 nsacls
23 - NS Fileset Attrs                  nsfilesetattrs
24 - NS Global Filesets                nsglobalfilesets
25 - NS Objects                        nsobjects
26 - NS Text Extensio ns               nstext
27 - Log Client Configurations         logclient
28 - Log Daemon Configurations         logdaemon
29 - Log Policies                      logpolicy
30 - Location Server Policies          lspolicy
31 - Metadata Monitor Configuration    mmonitor
32 - Migration/Purge Configurations    mps
33 - Migration Policies                migpolicy
34 - Purge Policies                    purgepolicy
35 - Migration/Purge Checkpoints       mpchkpt
36 - Mount Daemon Configuration        mountd
37 - Mover Configurations              mover
38 - Mover Devices                     moverdevice
39 - NFS V2 Daemon Configuration       nfs2
40 - PVL Configurations                pvl
41 - PVL Activities                    pvlactivity
42 - PVL Drives                        pvldrive
43 - PVL Jobs                          pvljob
44 - PVL Physical Volumes              pvlpv
45 - PVR Configurations                pvr
46 - 3494 Cartridges                   cartridge_3494
47 - 3495 Cartridges                   cartridge_3495
48 - AML Cartridges                    cartridge_aml
49 - Operator Cartridges               cartridge_operator
50 - STK Cartridges                    cartridge_stk
51 - Remote Site Configurations        site
52 - Storage Server Configurations     ss
53 - Storage Classes                   storageclass
54 - SS Disk Storage Maps              storagemapdisk
55 - SS Tape Storage Maps              storagemaptape
56 - SS Disk Storage Segments          storagesegdisk
57 - SS Tape Storage Segments          storagesegtape
58 - SS Disk Physical Volumes          sspvdisk
59 - SS Tape Physical Volumes          sspvtape
```

```
       60 - SS Disk Virtual Volumes           vvdisk
       61 - SS Tape Virtual Volumes           vvtape


SEE ALSO


SCCS
     man/managesfs.7, gen, 4.1.1, 4.1.1.1   4/5/99   16:14:44
```

I.37 metadata_info —Display Information on HPSS Metadata Files

```
NAME
     metadata_info - Display information on HPSS metadata files

SYNTAX
     metadata_info [-s] [<sfsServerName>]

DESCRIPTION
     This tool displays information on all SFS files managed by the SFS server
     <sfsServerName>.  If <sfsServerName> is not specified, the environment
     variable ENCINA_SFS_SERVER is used.

     By default, the total number of records stored in each file is displayed.
     The “-s” option will display additional information on record sizes as
     well as primary and secondary indices, including the size of each key and
     whether each key is unique.

     In order to use this utility, you must be DCE logged in as a principal
     with query permission for the SFS server.

EXAMPLE
     To display the number of records in each HPSS metadata file for SFS
     server “/.:/encina/sfs/hpss”, issue:

        % metadata_info /.:/encina/sfs/hpss
        SFS Files Managed by Server “/.:/encina/sfs/hpss”:


        accounting              : 1 records at 3400 bytes ea.
                                  Primary key is 4 bytes and UNIQUE
        accounting.convert.old  : 0 records at 4440 bytes ea.
                                  Primary key is 4 bytes and UNIQUE
        acctlog                 : 0 records at 32 bytes ea.
                                  Primary key is 16 bytes and UNIQUE
        acctsnap                : 0 records at 32 bytes ea.
                                  Primary key is 12 bytes and UNIQUE
        acctsum                 : 27447 records at 56 bytes ea.
                                  Primary key is 12 bytes and UNIQUE
        bfcoschange             : 0 records at 32 bytes ea.
                                  Primary key is 32 bytes and UNIQUE
        bfdiskallocrec          : 1152 records at 496 bytes ea.
                                  Primary key is 40 bytes and UNIQUE
        bfdisksegment           : 129 records at 56 bytes ea.
                                  Primary key is 44 bytes and UNIQUE
        bfmigrrec               : 12 records at 64 bytes ea.
                                  Primary key is 36 bytes and UNIQUE
                                  Secondary key 1 is 12 bytes and NOT unique
        bfpurgerec              : 72 records at 56 bytes ea.
                                  Primary key is 36 bytes and UNIQUE
                                  Secondary key 1 is 40 bytes and NOT unique
```

```
bfs                     : 2 records at 2896 bytes ea.
                          Primary key is 16 bytes and UNIQUE
bfsssegchkpt            : 0 records at 88 bytes ea.
                          Primary key is 68 bytes and UNIQUE
                          Secondary key 1 is 32 bytes and NOT unique
bfssunlink              : 0 records at 80 bytes ea.
                          Primary key is 72 bytes and UNIQUE
bfssunlink.convert.old  : 0 records at 64 bytes ea.
                          Primary key is 32 bytes and UNIQUE
bftapesegment           : 412 records at 112 bytes ea.
                          Primary key is 44 bytes and UNIQUE
bitfile                 : 10540 records at 352 bytes ea.
                          Primary key is 32 bytes and UNIQUE
bitfile.no_acct         : 617 records at 352 bytes ea.
                          Primary key is 32 bytes and UNIQUE
cartridge_3494          : 114 records at 220 bytes ea.
                          Primary key is 8 bytes and UNIQUE
                          Secondary key 1 is 12 bytes and UNIQUE
cartridge_3494_test     : 1 records at 220 bytes ea.
                          Primary key is 8 bytes and UNIQUE
                          Secondary key 1 is 12 bytes and UNIQUE
cartridge_3495          : 0 records at 220 bytes ea.
                          Primary key is 8 bytes and UNIQUE
                          Secondary key 1 is 12 bytes and UNIQUE
cartridge_aml           : 0 records at 220 bytes ea.
                          Primary key is 8 bytes and UNIQUE
                          Secondary key 1 is 12 bytes and UNIQUE
cartridge_operator      : 1 records at 220 bytes ea.
                          Primary key is 8 bytes and UNIQUE
                          Secondary key 1 is 12 bytes and UNIQUE
cartridge_stk           : 0 records at 220 bytes ea.
                          Primary key is 8 bytes and UNIQUE
                          Secondary key 1 is 12 bytes and UNIQUE
cos                     : 28 records at 88 bytes ea.
                          Primary key is 4 bytes and UNIQUE
                          Secondary key 1 is 32 bytes and UNIQUE
dmg                     : 1 records at 176 bytes ea.
                          Primary key is 16 bytes and UNIQUE
dmgfileset              : 21 records at 2312 bytes ea.
                          Primary key is 8 bytes and UNIQUE
                          Secondary key 1 is 16 bytes and NOT unique
filefamily              : 6 records at 100 bytes ea.
                          Primary key is 4 bytes and UNIQUE
                          Secondary key 1 is 32 bytes and UNIQUE
hierarchy               : 22 records at 148 bytes ea.
                          Primary key is 4 bytes and UNIQUE
logclient               : 2 records at 412 bytes ea.
                          Primary key is 16 bytes and UNIQUE
logdaemon               : 1 records at 1064 bytes ea.
                          Primary key is 16 bytes and UNIQUE
logpolicy               : 28 records at 40 bytes ea.
                          Primary key is 32 bytes and UNIQUE
lspolicy                : 1 records at 440 bytes ea.
```

```
                          Primary key is 4 bytes and UNIQUE
migpolicy               : 12 records at 160 bytes ea.
                          Primary key is 4 bytes and UNIQUE
mmonitor                : 1 records at 164 bytes ea.
                          Primary key is 16 bytes and UNIQUE
mountd                  : 1 records at 1056 bytes ea.
                          Primary key is 16 bytes and UNIQUE
mover                   : 5 records at 2228 bytes ea.
                          Primary key is 16 bytes and UNIQUE
moverdevice             : 295 records at 160 bytes ea.
                          Primary key is 4 bytes and UNIQUE
mpchkpt                 : 23 records at 72 bytes ea.
                          Primary key is 12 bytes and UNIQUE
                          Secondary key 1 is 12 bytes and UNIQUE
mps                     : 1 records at 2896 bytes ea.
                          Primary key is 16 bytes and UNIQUE
nfs2                    : 1 records at 4328 bytes ea.
                          Primary key is 16 bytes and UNIQUE
nsacls                  : 31 records at 204 bytes ea.
                          Primary key is 8 bytes and UNIQUE
nsconfig                : 1 records at 908 bytes ea.
                          Primary key is 16 bytes and UNIQUE
nsfilesetattrs          : 89 records at 212 bytes ea.
                          Primary key is 8 bytes and UNIQUE
nsfilesets              : 0 records at 360 bytes ea.
                          Primary key is 8 bytes and UNIQUE
nsglobalfilesets        : 89 records at 172 bytes ea.
                          Primary key is 8 bytes and UNIQUE
                          Secondary key 1 is 128 bytes and UNIQUE
                          Secondary key 2 is 24 bytes and NOT unique
                          Secondary key 3 is 24 bytes and NOT unique
nsobjects               : 104007 records at 328 bytes ea.
                          Primary key is 4 bytes and UNIQUE
                          Secondary key 1 is 12 bytes and NOT unique
                          Secondary key 2 is 8 bytes and NOT unique
                          Secondary key 3 is 36 bytes and NOT unique
nstext                  : 305 records at 1040 bytes ea.
                          Primary key is 5 bytes and UNIQUE
purgepolicy             : 12 records at 64 bytes ea.
                          Primary key is 4 bytes and UNIQUE
pvl                     : 3 records at 1580 bytes ea.
                          Primary key is 16 bytes and UNIQUE
pvlactivity             : 288 records at 1080 bytes ea.
                          Primary key is 16 bytes and UNIQUE
pvldrive                : 296 records at 168 bytes ea.
                          Primary key is 4 bytes and UNIQUE
                          Secondary key 1 is 12 bytes and NOT unique
pvljob                  : 288 records at 56 bytes ea.
                          Primary key is 4 bytes and UNIQUE
pvlpv                   : 403 records at 92 bytes ea.
                          Primary key is 12 bytes and UNIQUE
                          Secondary key 1 is 8 bytes and NOT unique
pvr                     : 4 records at 460 bytes ea.
```

```
                                Primary key is 16 bytes and UNIQUE
        pvr_shelf               : 6 records at 472 bytes ea.
                                Primary key is 16 bytes and UNIQUE
        serverconfig            : 37 records at 3840 bytes ea.
                                Primary key is 16 bytes and UNIQUE
                                Secondary key 1 is 32 bytes and UNIQUE
                                Secondary key 2 is 4 bytes and NOT unique
        site                    : 0 records at 184 bytes ea.
                                Primary key is 16 bytes and UNIQUE
        site_save               : 2 records at 168 bytes ea.
                                Primary key is 16 bytes and UNIQUE
        ss                      : 4 records at 728 bytes ea.
                                Primary key is 16 bytes and UNIQUE
        sspvdisk                : 289 records at 232 bytes ea.
                                Primary key is 12 bytes and UNIQUE
                                Secondary key 1 is 36 bytes and NOT unique
        sspvtape                : 91 records at 232 bytes ea.
                                Primary key is 12 bytes and UNIQUE
                                Secondary key 1 is 36 bytes and NOT unique
        storageclass            : 35 records at 160 bytes ea.
                                Primary key is 4 bytes and UNIQUE
        storagemapdisk          : 31 records at 2152 bytes ea.
                                Primary key is 32 bytes and UNIQUE
        storagemapdisk.save     : 28 records at 2200 bytes ea.
                                Primary key is 32 bytes and UNIQUE
                                Secondary key 1 is 32 bytes and NOT unique
        storagemaptape          : 96 records at 152 bytes ea.
                                Primary key is 32 bytes and UNIQUE
                                Secondary key 1 is 20 bytes and NOT unique
                                Secondary key 2 is 24 bytes and NOT unique
        storagesegdisk          : 8537 records at 336 bytes ea.
                                Primary key is 32 bytes and UNIQUE
                                Secondary key 1 is 32 bytes and NOT unique
        storagesegtape          : 435 records at 336 bytes ea.
                                Primary key is 32 bytes and UNIQUE
                                Secondary key 1 is 32 bytes and NOT unique
        vvdisk                  : 31 records at 312 bytes ea.
                                Primary key is 32 bytes and UNIQUE
        vvtape                  : 77 records at 312 bytes ea.
                                Primary key is 32 bytes and UNIQUE

FILES


SEE ALSO

     sfsadmin query file

SCCS
     man/metadata_info.7, gen, 4.1.1, 4.1.1.1   4/7/99   14:16:22
```

I.38 mps_reporter — HPSS MPS Report Utility

```
NAME
     mps_reporter - HPSS Migration/Purge Server Report Utility

SYNTAX
     mps_reporter [ -b ] file_list

DESCRIPTION
     mps_reporter is a utility which generates user readable output
     from raw report files created by the Migration Purge Server (MPS).

     Each file in file_list is processed and the output is sent to stdout.

     A report contains the following types of information based on events
     recorded in the MPS:

         When the MPS started up.

         Information about the disk migration, disk purge or tape
         migration of a Bitfile.

         Summary information of a disk migration, disk purge or tape
         migration run.

OPTIONS
     -b  Generate a brief summary report only.

EXAMPLE
     The following command will generate a report from a raw MPS report
     file and display the results on stdout.

     $ mps_reporter mps_report.960521

     The following command  will generate a summary report from a raw
     MPS report file.

     $ mps_reporter -b mps_report.960521

     The following command will generate a report for the month of May
     and save the results in the “Report.9605” file.

     $ mps_reporter mps_report.9605?? > Report.9605

FILES


SEE ALSO


SCCS
     man/mps_reporter.7, gen, 4va  7/3/96  17:55:04
```

I.39 multinoded — HPSS Multinode Daemon

```
NAME
multinoded      - HPSS Multinode Daemon.

PURPOSE
The HPSS Multinode Daemon provides all the functions for the HPSS
Parallel FTP Client to perform a multinodal (multiprocessor) parallel
transfer.

SYNTAX
Note:   The HPSS Multinode Daemon is normally started by the inetd daemon.
It can also be controlled from the command line, using SRC commands.

multinoded [ flags ]

DESCRIPTION
The HPSS Multinode Daemon is the mechanism that allows the HPSS Parallel
FTP Client to perform a parallel transfer of a file to (from) a HPSS
host using multiple (remote) processors.  The HPSS Multinode Daemon
needs to be configured on a number of nodes.  These nodes can be an
individual workstation, or they can be a addressable processor, like
those found on an SP2.  Once the HPSS  Multinode Daemons have been
configured to run on a number of nodes, a list containing those nodes
needs to be created and made available to every workstation that will be
using the HPSS Parallel FTP Client.  The list of available nodes will be
referred to as the “Configuration File.”

FLAGS
-r  Enable root to transfer a file using the HPSS Multinode Daemon.  By
    default root file transfers are not allowed.  Allowing root to
    transfer a file is a security hole.

-i  Enable identification daemon (identd) authentication.  The identd
    must be running on every client that will be using the HPSS
    Multinode Daemon running with this flag.

CONFIGURE A NODE TO RUN THE HPSS MULTINODE DAEMON
The HPSS Multinode Daemon is started by the inetd daemon.  Two files
must be modified before the inetd daemon will automatically start a
process:
     (1) /etc/services
     (2) /etc/inetd.conf

An entry must be added to the /etc/services file for the HPSS Multinode
Daemon.  The following is an example entry similar to the one that
should be added to the /etc/services file:

multinoded      2513/tcp       # multinode daemon

NOTE: Once you have selected a port, it MUST remain constant across EVERY
      node.
```

```
Furthermore, an entry must be added to the /etc/inetd.conf file for the
HPSS Multinode Daemon.  The following are two examples of entries,
similar to the one that should be added to the /etc/inetd.conf file:

multinoded  stream  tcp nowait  root  /[path]/multinoded multinoded
multinoded  stream  tcp nowait  root  /[path]/multinoded multinoded -i

BUILDING A CONFIGURATION FILE
A configuration file will be used to specify different IP addresses
or remote nodes on which the data transfer processes are to be spawned by
HPSS Parallel FTP Client.  The default configuration file is
/usr/local/ftp/pftp.config.  The user can override this with the
PFTP_CONFIG_FILE environment variable, or by specifying the -m command
line option (e.g. pftp_client -m./pftp.config).

The configuration file will contain the following information:

Control Address
   The IP address or machine name where a multinode daemon has been
   configured to run.
Data Address
   The IP address or name to be used by the multinode daemon to move data
   from the mover to the multinode daemon.  If a data address is not
   specified, the control interface will be used as the data interface.

SAMPLE ENTRIES IN A CONFIGURATION FILE
# Control_Interface Data_Interface Comment
192.94.47.21           # pluto: control & data share 192.94.47.21
192.94.47.43           # hpss : control & data share 192.94.47.43
pluto                  # pluto: control & data share pluto (192.94.47.21)
hpss                   # hpss : control & data share hpss (192.94.47.43)
192.94.47.21 192.225.22.21 # pluto: control on 192.94.47.21
                       #          data on 192.225.22.21
192.94.47.43 192.225.22.43 # hpss : control on 192.94.47.43
                       #          data on 192.225.22.43
pluto pluto-F          # pluto: control on pluto
                       #          data on pluto-F

LOAD LEVELING
Multinode parallel ftp load leveling across the various nodes is
accomplished using an offset file and a properly ordered configuration
file. The offset file is automatically created and maintained by the HPSS
Parallel FTP Client.  The offset file will have the same path and name as
the configuration file, with the .off extension.  The offset file is used
to level the load across every interface in the configuration file for
each instance of the HPSS Parallel FTP Client.  By placing the
configuration file in a shared location, load leveling can be achieved
for each instance of the HPSS Parallel FTP Client across the network.
Each instance of the HPSS Parallel FTP Client builds its own internal
node list using the configuration file.  The HPSS Parallel FTP Client
will use the same internal node list, to transfer files, its entire
life.  Nodes can only be added to this list by increasing the pwidth.
```

```
Decreasing the pwidth will not remove nodes from the internal node
list.  Because of this, it is very important to properly order the
configuration file by alternating the interfaces of various processors.

FILES
/etc/services
/etc/inetd.conf
/usr/local/ftp/pftp.config

RELATED INFORMATION
ftp, pftp_client(7), pftpd(7), auth_manager(7).

The syslogd daemon.

The .netrc file format.


SCCS
       @(#)90   1.3   man/multinoded.7, gen, 4.1.1, 4.1.1.1   4/5/99   17:21:16
```

I.40 nfsmap — Manipulate the HPSS NFS Daemon's Credentials Map

```
NAME
     nfsmap - manipulate the HPSS NFS daemon’s credentials map

SYNTAX
     nfsmap [options...] [command] [nfs_hostname]

DESCRIPTION
     The nfsmap program manages entries in the HPSS NFS daemon’s credentials
     map.  This map is used to determine which uid is used when a user
     accesses HPSS files using NFS.  End users can make entries in the map
     on their own behalf by running dce_login to acquire a DCE ticket, and
     then running nfsmap to make the entry itself.  Privileged users may
     also make entries on behalf of other users.

OPTIONS
     -a                 list/delete all entries of a specified category
     -c <chargecode>    charge code
     -f <flag>          flags controlling the type of entry
     -h <hostname>      host for the credential map entry
     -l <lifetime>      lifetime of credential
     -p <principal>     DCE Principal name
     -u <uid>           User id for credentials map entry

COMMANDS
     add                add a credential map entry
     delete             delete one or more entries from credentials map
     help               get help on how to use nfsmap
     list               list one or more entries in credentials map
     reset              clear the map

DEFAULTS
     chargecode         0 => use default value in DCE registry
     command            The add command
     flag               Entry not locked, not permanent.
                        Use flag = l, p, or lp to change this.
     hostname           The host where the person is running the program.
     lifetime           25h [i.e., 25 hours]
     nfs_hostname       The host where the person is running the program.
     principal          DCE principal name of person running program.
     uid                Unix uid of the person running the program.

DISCUSSION
     When a user tries to access an NFS file, the NFS daemon looks in
     credentials map to determine the user’s access rights.  The map
     contains entries that map the user’s uid into the actual uid that will
     be used to determine the access rights.  The entries are keyed to the
     computer from which the user is trying to access the file.  If the user
     is logged in to several computer, s/he will need to run nfsmap on each
```

one.

To use the program, the user must first run dce_login to obtain a ticket, and then run nfsmap to make the map entry itself. Users may make entries in the map on their own behalf, provided the NFS daemon has been configured to permit this. The daemon may also be configured to define a set of “privileged” users who may make entries in the map on behalf of other users. This feature can be used to define permanent entries, for example.

The NFS daemon maps between uids, so that the uid used by the user on the remote host need not agree with the one used for determining file access rights on HPSS. The daemon’s credentials map contains the information needed to map between the remote uid and the HPSS uid. By default, the remote uid is the one that the user is using when s/he runs nfsmap, and the HPSS uid is the one in the DCE account that the user has logged in to using dce_login. These defaults may be changed by using the -u and -p options, respectively. Only privileged users may use these options.

The mapping that will be done depends on the host from which the user will be trying to access files. By default, the map entry uses the host on which nfsmap was invoked. This can changed using the -h switch. Only privileged users may use this option.

By default, the ‘list’ and ‘delete’ commands apply to only one entry in the credentials map. By using the -a option, this behavior can be changed so that all entries can be listed or deleted.

The default charge code is obtained from the user’s entry in the DCE registry. The default can be overridden by using the -c option. The chargecode must be a numeric value, but the interpretation of code is defined by the site’s administrator.

The flag field can only be set by a privileged user. When a map entry is locked, the corresponding user cannot modify the entry. When an entry is permanent, it will never expire, regardless of the value of the lifetime.

The lifetime field determines how long an entry will last before it is removed from the credentials map. The default is 25 hours. Normally, the daemon automatically removes entries from the credentials map as soon as they expire. However, there is site-defined grace period that preserves credentials that have been recently used. This can be used to protect long-running jobs that make occasional use of NFS.

The lifetime is expressed in seconds by default. However, you may use forms like 10d, 10h, or 10m to express the time in days, hours, or minutes respectively.

Note that not all options make sense for a particular command. For example, the -c option cannot be used with the delete command.

```
EXAMPLES
     Unless otherwise stated, the following commands assume that the NFS
     deamon is running on host tardis, and that the user is running on host
     k9, where the file system will be mounted.  The examples also assume
     that the person running nfsmap has already run dce_login.

     To add a credential:
       nfsmap add tardis

     To replace the credential in the previous example with a new one:
       nfsmap add tardis

     To add a credential with a lifetime of 5 minutes:
       nfsmap add -l 5m tardis

     To add a permanent credential:
       nfsmap add -f p tardis

     To add a credential that is both permanent and locked:
       nfsmap add -f lp tardis

     To add a credential on behalf of user joe who is running on host k9
     with uid 200:
       nfsmap -u 200 -p joe -h k9 add tardis

     To list one's own credential:
       nfsmap list tardis

     To list all credentials:
       nfsmap -a list tardis

     To delete a credential:
       nfsmap delete tardis


RESTRICTIONS
     To run the program, you must run dce_login first.

     Users must have entries in the DCE registry before entries can be made
     for them.

SEE ALSO
     nfs(5)
```

I.41 nsde — Name Server Database Editor

```
NAME
    NSDE - for Name Server Database Editor

SYNTAX
    nsde  [-f <FileName>]

    where

    -f <FileName>   optional command that specifies which input file NSDE
                    is to read to find the Name Server file names.  The
                    default file is '.nsde'.  See the NSDE Input Files
                    section below.

DESCRIPTION
    NSDE (for Name Server Database Editor) is a utility that can display
    and symbolically edit the Name Server database files.

NSDE interactive commands

    n               displays the n'th block.  This block becomes the current
                    block.

    <cr>            a CarriageReturn causes the current block to be displayed.

    aclmode         all future blocks that are displayed will be ACL blocks
                    from the ACL file.

    bye end exit quit or q
                    causes NSDE to do just what you'd expect.

    cc [<StartingIndex>]
                    (for Consistency Check) this option causes NSDE to
                    perform a general consistency check of the database.
                    If the optional StartingIndex parameter is specified,
                    the consistency check will begin at that index (RSN).

    ccf [<Date> [<Time]]
                    (for Consistency Check Files) NSDE will check each file
                    found in the Name Server database against the Bitfile
                    Server's database. If a corresponding Bitfile entry is
                    NOT found, the pathname to the file is sent to standard
                    out. An optional Date and Time may be supplied and only
                    files created after this time will be examined. If no
                    Date or Time are supplied, all files are examined.

    ccfv [<Date> [<Time>]]
                    (for Consistency Check Files Verbose output). This option
                    is the same as the ccf option except much more
                    information is sent to standard out. In addition to the
```

path names, the object block and an ACL notification are sent.

comment or com — when an object block is displayed this command causes any attached Comment to be displayed.

debug or debugon
turns on the debugging flag which causes debugging messages to be sent to standard out. See ‘debugoff’.

debugoff — turns off the debugging flag and debugging messages will no longer be sent to standard out. See ‘debug’.

dsr or dontshowres or dontshowreserved
object blocks which are printed to standard out will not contain the ‘reserved’ fields. This is the default. See the showreserved command.

dumpname or dn — if the current block is an Object Block the Name field will be shown in both hexidecimal and ASCII. However if the current block is a Text block the Text field will be shown in both hexidecimal and ASCII.

extname or en — the Extended Name text block associated with the current object block will be fetched and displayed.

ffle or fde — (for Find FreeList Entries) or (Find Deleted Entries) this option causes NSDE to display the indexes of all the entries that are currently on the FreeList (entries that have been deleted). NSDE uses the BitfileIdKey to find these entries.

fflec or fdec — (for Find FreeList Entry Count) or (Find Deleted Entry Count) this option causes NSDE to count the number of FreeList entries and to report this count.

find <FieldName> = <Value>
examines FieldName in each block looking for a match to Value and prints the index of each block containing this Value. Note that we can search for ‘<’, ‘>’, or ‘=’ Value. If the field being examined is a field which contains an ASCII string (these fields are Name, FilesetName, Text, and UserData) the syntax changes slightly-- an ‘=’ causes NSDE to look for an exact match to the string, but a ‘>’ or ‘<’ causes NSDE to do a ‘pattern’ match. A pattern match occurs if Value is found in any portion of string.

findallsymlinks or fasl
find and displays all symbolic link entries and their associated symbolic link data.

findholes or fh — finds all of the ‘holes’ in the object block file. A

hole is a record that has never been written. The RSN of each hole is reported.

fixgfsr <NameServerUUID>
(for FIX Global FileSet Records) this command fixes a problem in the Name Server's Global Fileset file. The NameServerUUID is the UUID of the Name Server that owns the Global Fileset file.

fixhi <NumberToFix> [<StartingIndex>]
(for FIX HashIndex) this command looks at each Name entry looking for 'fat' (the upper bit is set) characters. When it finds a Name that contains any fat characters it checks the HashIndex for correctness. If the HashIndex is incorrect it is corrected. NumberToFix is for confidence building-- nsde will only 'fix' this many records and then stop fixing records. The user can then check that the fixes were made correctly. The optional StartingIndex is the RSN where nsde is to begin looking for records to fix or report.

flfsa <FilesetId>
(for Force Load FileSetAttrs) the FilesetAttrs block indentified by FilesetId is forced to become the Current Block.

flgfs <FilesetId>
(for Force Load Global FileSet) the Global Fileset block indentified by FilesetId is forced to become the Current Block.

fn <FieldName> = <Value>
(for Find Next) this command is similar to the 'find' command, but differs in that it starts its search from the 'current block + 1', and stops after finding the Next matching entry. Note that, like the find command, we can search for '<', '>', or '=' Value.

fs or fileset the FilesetAttrs record associated with the current object will be displayed.

fsht (for Fileset Hash Table) to create the fileset hash table to assist in Fileset file lookup (fs or fileset command).

fshtp (for Fileset Hash Table print) to print the existing entries in the Fileset Hash Table.

gfs (for Global FileSet) the Global Fileset record associated with this entry will be displayed.

help causes the help package to be sent to standard out.

```
lastblock or lb or lrsn
                determine and print out the RSN of the last object block
                written to the database.  Note that “last” is used in
                the spatial sense, not the temporal sense.

lfsa            (for List FileSetAttrs) all of the FilesetAttr records
                will be displayed.

lgfs            (for List GlobalFileSets) all of the GlobalFileset
                records will be displayed.

next or n       causes NSDE to fetch and display the next object block.

nextacl or nacl if the current block is an OBlock, any ACLs pointed to
                from this OBlock will be displayed.  If the current
                block is an ACLBlock, any ACLS pointed to from this
                ACLBlock will be displayed.

numrecordsused or nru
                causes NSDE to display the number of object block records
                that have been used.

oblockmode      all future blocks that are displayed will be Object
                blocks from the object block file.  This is the default.

path or p       send the path name to standard out.  The path name from
                the root block to the current block is sent to standard
                out.

set <FieldName> = <Value>
                sets the FieldName in the current block to Value.

setall [<StartingIndex>] <FieldName> = <Value>
                sets the FieldName field to the value found in Value in
                ALL of the object blocks. If the optional StartingIndex
                parameter is supplied, setall starts in that object
                block.  Currently this command operates only on the
                object block Version field.

sfs <FilesetId>
                shows the FilesetAttrs for the given FilesetId.

showreserved or sr or showres
                the reserved fields of any object blocks that are sent
                to standard out will be printed.

sizes           the sizes of several interesting data structures related
                to the Name Server will be printed.

stats [<StartingIndex>]
                NSDE will tally up all of the object block
                statistics. This tallied information will be collected
                into the global fileset structures. If the optional
```

parameter StartingIndex is supplied, NSDE will begin gathering statistic from this object block index. If no StartingIndex is supplied, NSDE will start at object block zero. Also see the statsffs, statsall, and statsgt commands.

statsall or sta
all of the collected statistics are displayed. If no statistics have been collected yet, we collect them and then display them. This command displays the statistics for each fileset and then shows the grand totals.

statsffs <FilesetId>
(for STATisticS For Fileset) NSDE will display the statistics data we have collected for the indicated FilesetId. If we haven't collected any statistics data yet, we will collect it and then hunt for the indicated FilesetId.

statsgt (for STATisticS Grand Total) display the Grand Totals of all the statistics data we have collected. If we haven't collected any statistics data yet, then collect it and then display the Grand Totals.

symlink or sl the text block containing the Symbolic Link data associated with the current object block will be fetched and displayed.

zap [n1 n2 ... nm]
if 'n' values are supplied this option causes object blocks n1, n2, ... nm to be put onto the FreeList. This is done by writing the FreeList pattern into the indicated block(s). If no 'n' values are supplied, the current block is zapped. If the current block is an object block, the block is overwritten with FreeList pattern. If, however, the current block is an ACLBlock or a TextBlock, the block is deleted and the corresponding object block (if any) is updated.

zapnb n1 n2 ... nm
(for ZAP New Block(s)) this command is similar to the zap command, but writes the FreeList pattern to previously unwritten blocks. The 'n' values are not optional. This command only operates on ObjectBlocks.

zeroreserved or zr
this command causes all of the the reserved fields in the current object block to be zeroed out (set to zero).

NSDE Input Files

The Name Server uses five SFS files to store its metadata. The path names to these files, as well as one of the Bitfile Servers metadata

```
    files, are stored in either environment variables or in a file that we
    refer to as an 'NSDE Input File'. If the user explicitly supplies an
    NSDE Input File we gather all of the path names from that file. If the
    user does not explicitly supply an NSDE Input File name we first search
    for the pathnames in certain environment variables, and failing this,
    we assume a default NSDE input file name which is a 'dot' file named
    ".nsde".  We look first for the .nsde file in the current working
    directory and then next in the user's HOME directory.  If we can't find
    a .nsde file (or if it doesn't contain one or more of the path names)
    NSDE will supply default path name(s) and hope for the best.  Following
    is a typical NSDE Input File used by NSDE on the Livermore ripsaw
    machine.  This file is named .nsde and is kept in the user's home
    directory.

    #
    #    This file contains path names to the Name Server's SFS
    #    metadata files and one of the Bitfile Server's metadata
    #    files. NSDE reads this file to obtain these path names.
    #
    #
    ACLFile           /.:/encina/server/sfs/hpss/nsacls.4v1
    FilesetAttrsFile  /.:/encina/server/sfs/hpss/nsfilesetattrs.4v1
    GlobalFSFile      /.:/encina/server/sfs/hpss/nsglobalfs.4v1
    ObjFile           /.:/encina/server/sfs/hpss/nsobjects.4v1
    TextFile          /.:/encina/server/sfs/hpss/nstext.4v1
    BFSFile           /.:/encina/server/sfs/hpss/bitfile.4v1

SCCS
    man/nsde.7, gen, 4va   11/4/98   18:36:33
```

I.42 pftp_client — Transfers files between a local host and a remote host

```
NAME
pftp_client           - HPSS Parallel FTP Client user/password
                        authentication over TCP/IP
pftp_client_ident     - HPSS Parallel FTP Client IDENT authentication
                        over TCP/IP
krb5_gss_pftp_client  - HPSS Parallel FTP Client using Generic Security
                        Service and Kerberos credentials over TCP/IP
pftp_client_ipi3      - HPSS Parallel FTP Client user/password
                        authentication over HIPPI
pftp_client_ident_ipi3
                      - HPSS Parallel FTP Client IDENT authentication
                        over HIPPI
krb5_gss_pftp_client_ipi3
                      - HPSS Parallel FTP Client using Generic
                        Security Service and Kerberos credentials over
                        HIPPI
PURPOSE
Transfers files between a local host and a remote host.

SYNTAX
pftp_client [ standard ftp flags ][ extended pftp flags ]
            [ HostName [ Port ] ]

DESCRIPTION
The pftp_client command uses the File Transfer Protocol (FTP) to transfer
files between a local host and a remote non-HPSS host, or a local host
and a remote HPSS host.  If communicating with a remote HPSS host, the
pftp_client command can perform a parallel file transfer utilizing a
single local processor (parallel transfer) or multiple processors
(multinode transfer).

STANDARD FTP FLAGS
Consult the ftp man pages.

EXTENDED PFTP FLAGS
-k NAME     Sets the Generic Security Service realm to NAME.

-t          Turns on packet tracing.

-m          Enable the use of multiple processors to complete a parallel
            file transfer to/from an HPSS host.  To utilize multiple
            processors: the multinode daemon (multinoded) must be employed;
            and a parallel ftp configuration file, containing a list of
            nodes configured to run the multinode daemon, must be available.

-w VAL      Sets the parallel stripe width to VAL.

-p NAME     Sets the partition name to NAME.
```

```
ISSUING SUBCOMMANDS
See the UNIX ftp man page.

STANDARD FTP COMPATIBLE SUBCOMMANDS
!            disconnect      mput             rhelp
$            form            nlist            rmdir
?            get             nmap             rstatus
account      glob            ntrans           runique
append       hash            open             send
ascii        help            prompt           sendport
bell         image           proxy            size
binary       lcd             put              status
bye          ls              pwd              struct
case         macdef          quit             sunique
cd           mdelete         quote            system
cdup         mdir            recv             tenex
close        mget            reinitialize     trace
cr           mkdir           remotehelp       type
delete       mls             rename           user
debug        mode            reset            verbose
dir          modtime         restart

UNSUPPORTED FTP SUBCOMMANDS
block        exp_cmd         non-print        stream
ebcdic       local           record           telnet

EXTENDED PFTP SUBCOMMANDS
mpget        pappend         pput             reget
mpput        passive         protect          setpblocksize
multinode    pget            psocket          setpwidth
newer        pipi3           quote site/site  umask
fget             fput        mfget         mfput

DESCRIPTION OF EXTENDED PFTP SUBCOMMANDS
mpget                    Get multiple files in parallel.
mpput                    Send multiple files in parallel.
multinode   Enable the use of multiple processors to complete a parallel
                         file transfer to (from) an HPSS host.  To utilize
multiple
                         processors: the multinode daemon (multinoded) must be
                         employed; and a parallel ftp configuration file,
                         containing a list of nodes configured to run the
multinode
                         daemon, must be available.
newer                    Get file if remote file is newer than local file.
pappend                  Send one file in parallel.
passive                  Toggle passive transfer mode.  (passive mode NOT
supported
                         by HPSS PFTP Daemons)
pget                     Retrieve one file in parallel.
pipi3                    Set parallel transfers to go over ipi3.
pput                     Send one file in parallel.
protect                  Set protection level for file transfer.
```

```
psocket                 Set parallel transfers to go across sockets.
reget                   Get file restarting at end of local file.
setpblocksizeSet the block size of the parallel stripe.
setpwidth   Set the width of the parallel stripe.
quote site
site                    Send site specific command to remote server. Try “rhelp
                        site” or “site help” for more information. quote site or
                        site supported commands:

                        chgrp          Change group ownership of remote file.
                        chmod          Change file permissions of remote file.
                        chown          Change ownership of remote file.
                        chuid          Change ownership of remote file.
                        setcos         Set the preferred Class of Service.
                        symlink        Establish a symbolic link.

umask                   Get (set) umask on remote side.

fget, mfget Get or get multiple files from HPSS and store into a local
                        parallel file system (e.g., GPFS)
fput, mfput Put or put multiple files from a local parallel file system
                        (e.g., GPFS) into HPSS.

EXAMPLES
pftp_client destination port
pftp_client_ident destination port
krb5_gss_pftp_client destination port

BUILDING A CONFIGURATION FILE

A configuration file will be used to specify different IP addresses
or remote nodes on which the data transfer processes are to be spawned by
HPSS Parallel FTP Client (multinode mode) or which network interfaces to
use for network striping (multihomed mode).  The default configuration file
is /usr/local/ftp/pftp.config.  The user can override this with the
PFTP_CONFIG_FILE environment variable, or by specifying the -m command
line option (e.g. pftp_client -m./pftp.config).  It is the Administrators
responsibility to correctly configure and maintain the pftp.config
file as well as implementing the multinoded components on all remote
systems.

The configuration file will contain the following information:

Control Address
   The IP address or machine name where a multinode daemon has been
   configured to run or the network interface (multihomed clients)
   to use for each child process. For the non-multinode (multihomed mode)
   parallel FTP Client, the first Control_Interface will be used for the
   first child process, the second for the second child process, etc.  If
   more striping is specified than interfaces in the pftp.config file,
   one or more interfaces will be used as appropriate.  Independent
   pftp_clients have NO knowledge of other pftp_clients and NO effort is
   made to maintain state within the pftp.config file.  If the client is
```

```
   run in the multihomed mode and the Control_Interface in the pftp.config
   file specifies an invalid address for the local machine, the client
   will FAIL - If a site wishes to implement the multinode mechanism, it
   is STRONGLY recommended that a script be developed (specifying the -m
   option) which is to be used by all customers on the host.

Data Address
   The IP address or name to be used by the multinode daemon to move data
   from the mover to the multinode daemon.  If a data address is not
   specified, the control interface will be used as the data interface.
   The Data Address is ignored for the multihomed situation.

SAMPLE ENTRIES IN A CONFIGURATION FILE

# Control_InterfaceData_Interface   Comment
my_interface1       # My first Interface(multihomed clients)
my_interface2       # My second Interface(multihomed clients)
my_interface3       # My third Interface(multihomed clients)
my_interface4       # My fourth Interface(multihomed clients)
192.94.47.21           # pluto: control & data share 192.94.47.21
192.94.47.43           # hpss : control & data share 192.94.47.43
pluto                  # pluto: control & data share pluto (192.94.47.21)
hpss                   # hpss : control & data share hpss (192.94.47.43)
192.94.47.21 192.225.22.21 # pluto: control on 192.94.47.21
                       #           data on 192.225.22.21
192.94.47.43 192.225.22.43 # hpss : control on 192.94.47.43
                       #           data on 192.225.22.43
pluto pluto-F          # pluto: control on pluto
                       #           data on pluto-F

LOAD LEVELING
Multinode parallel ftp load leveling across the various nodes is
accomplished using an offset file and a properly ordered configuration
file. The offset file is automatically created and maintained by the HPSS
Parallel FTP Client.  The offset file will have the same path and name as
the configuration file, with the .off extension.  The offset file is used
to level the load across every interface in the configuration file for
each instance of the HPSS Parallel FTP Client.  By placing the
configuration file in a shared location, load leveling can be achieved
for each instance of the HPSS Parallel FTP Client across the network.
Each instance of the HPSS Parallel FTP Client builds its own internal
node list using the configuration file.  The HPSS Parallel FTP Client
will use the same internal node list, to transfer files, its entire
life.  Nodes can only be added to this list by increasing the pwidth.
Decreasing the pwidth will not remove nodes from the internal node
list.  Because of this, it is very important to properly order the
configuration file by alternating the interfaces of various processors.

FILES
~/.netrc
/usr/local/ftp/pftp.config
PFTP_CONFIG_FILE environment variable
-m config_file_name command line option
```

RELATED INFORMATION
ftp, multinoded(7), pftpd(7), auth_manager(7).

The .netrc file format.

Copying Files Using the ftp Command in AIX Version 4.1 System User's Guide: Communications and Networks.

Network Overview in AIX Version 4.1 System Management Guide: Communications and Networks.

HPSS User's Guide

HPSS System Administration Guide

SCCS

SccsId[] = " @(#)91 1.4 man/pftp_client.7, gen, 4.1.1 2/7/00 14:58:44";

I.43 pftpd — HPSS Parallel FTP Daemon

NAME

hpss_pftpd - HPSS Parallel FTP Daemon using user/password authentication.
hpss_pftpd_amgr - HPSS Parallel FTP Daemon using an Authentication Manager

PURPOSE

The HPSS Parallel FTP Daemon provides all the functions of the standard FTP Daemon plus additional features providing parallel network functions when communicating with the HPSS Parallel FTP Client.

SYNTAX

Note: The HPSS Parallel FTP daemon is normally started by the inetd daemon. It can also be controlled from the command line, using SRC commands. TCP wrapper applications may also be specified. (NOT provided by HPSS.)

hpss_pftpd [flags]

DESCRIPTION

The HPSS Parallel FTP daemon is the DARPA Internet File Transfer Protocol (FTP) server process. The HPSS Parallel FTP daemon uses the Transmission

Control Protocol (TCP) to listen at the port specified with the ftp command service specification in the /etc/services file.

Both HPSS Parallel FTP Daemons will substitute the directory: /users/no_home_dir for the home directory if no home directory is specified in the appropriate ftppasswd file or the DCE Registry. This directory should be created by the HPSS Administrator with the permissions r-xr-xr-x. Messages will be logged in syslog (if activated) to prompt the HPSS Administrator to correct the problem; e.g., insert a home directory in the appropriate field.

The Credential-based (Generic Security Service, GSS) version of the hpss_pftpd_amgr supports Cross Cell Authentication. With Cross Cell Authentication the client user's home directory is read from the remote DCE Registry (either the home attribute or the HPSS.homedir Extended Registry attribute). The local home directory is preficed with textual contents of the remote DCE Cellname; e.g., the local DCE cell is /.../my.dce.cell, the end user is george in a remote DCE cell, /.../his.dce.cell. The home directory for george in /.../his.dce.cell is /home/georges.homedir. The hpss_pftpd_amgr will establish georges HPSS home directory as: /his.dce.cell/home/georges.homedir. For Cross Cell support, it is necessary to create the appropriate home directories for the Cross Cell customers AND the Access Control Lists for the directories must be set using the insif utility. Refer insif(7).

Changes to the HPSS Parallel FTP daemon can be made using the System Management Interface Tool (SMIT) or System Resource Controller (SRC), by editing the /etc/inetd.conf or /etc/services file.

If you change the /etc/inetd.conf or /etc/services file, run the refresh -s inetd or kill -1 InetdPID command to inform the inetd daemon of the changes to its configuration files.

Supported File Transfer Protocol Requests

The ftpd daemon currently supports the following FTP requests:

ABOR	Terminates previous command.
ACCT	Specifies account (ignored).
ALLO	Allocates storage (vacuously).
APPE	Appends to a file.
CDUP	Changes to the parent directory of the current working directory.
CWD	Changes working directory.
DELE	Deletes a file.
HELP	Gives help information.
IDLE	Sets idler time (SITE IDLE 60).
LIST	Gives list files in a directory.
MKD	Makes a directory.
MDTM	Shows last modification time of file.
MODE	Specifies data transfer mode.
NLST	Gives a name list of files in directory (this FTP request is the same as the ls command).
NOOP	Does nothing.

```
PASS     Specifies a password.
PASV     Prepares for server-to-server transfers.
PORT     Specifies a data connection port.
PWD      Prints the current working directory.
QUIT     Terminates session.
RETR     Retrieves a file.
RMD      Removes a directory.
RNFR     Specifies rename-from file name.
RNTO     Specifies rename-to file name.

SITE     The following nonstandard or UNIX-specific commands are supported
         by the SITE request:
         CHMOD    Changes mode of a file (SITE CHMOD 755 FileName).
         HELP     Gives help information (SITE HELP).
         UMASK    Changes umask (SITE UMASK 002).
SIZE     Returns size of current file.
STAT     Returns the status of the server.
STOR     Stores a file.
STOU     Stores a file using a unique file name.
STRU     Specifies the structure of data transfer as a file structure.
SYST     Shows operating system type of server system.
TYPE     Specifies data transfer type with the Type parameter.
USER     Specifies user name.

HPSS EXTENDED SUPPORTED FTP REQUESTS

ALLO64   Allocate storage.
AUTH     Authentication type.
ADAT     Authentication date.
PBSZ     Parallel buffer size.
PCLO     Parallel close.
POPN     Parallel open.
PPOR     Parallel port.
PROT     Protection level.
PRTR     Parallel retrieve.
PSTO     Parallel store.
SITE     The following nonstandard or UNIX-specific commands are supported
         CHGID    Change group id.
         CHGRP    Change group.
         CHOWN    Change owner.
         CHUID    Change user id.
         SETCOS   Set class of service.
         STAGE    Stage file.
         SYMLINK  Symbolic link.
         WAIT     Wait.

The ftpd daemon should be controlled using the System Management Interface
Tool (SMIT) or by changing the /etc/inetd.conf file. Entering ftpd
at the command line is not recommended.

FLAGS

-a      For the GSS version of the Daemon this mandates the use of
```

credentials for authentication. The “user” command is disabled. Do NOT specify for the non-GSS hpss Parallel FTP Daemon!

-b Used to specify the FTP Buffersize. Specify the “hpss_option BUFSZ #” in the ftpaccess file. The # is the size of the buffer in MegaBytes (# * 1048576).

-d Turns on debugging. This sets the HPSS environment variable HPSS_DEBUG to 1. This option may be repeated up to three (3) times on the inetd command line to increment the level of debugging desired

-h Causes the “hpss_option HOST” record to be read from the ftpaccess file. This is used to specify the network interface to be used between the HPSS FTP Daemon and the HPSS Data Movers for non-parallel data transfers. This sets the environment variable HPSS_HOSTNAME used by the Client API.

-p # Allows for specification of a port for listening.

-s str This is used to specify the syslog “facility”. The default is LOG_DAEMON (3<<3). The format of the string is “3<<3” (including the parenthesies). Refer to the appropriate syslog header files for specification of other syslog facilities; e.g., LOG_LOCAL7.

-t Used to specify the Daemon timeout value in seconds. This reads the “hpss_option DTO #” record from the ftpaccess file.

-u Used to specify the default umask for the FTP Daemon. This reads the “hpss_option UMASK octal” record from the ftpaccess file. Note: the value MUST be a valid umask (in octal) between 000 and 777.

-v Used to specify HPSS Client API logging. The specifics for HPSS Client API logging are dependent upon “log” commands in the ftpaccess file. HPSS Client API Logging for the HPSS Parallel FTP Daemons is recorded in the hpss_ftpd.log file in the adm sub-directory of the FTPBaseDir - default: /var/hpss/ftp/. E.g., /var/hpss/ftp/adm/hpss_ftpd.log.

-z Turns on a 15 second sleep in the Daemon to allow attaching in the debugger.

-A Used to allow “passive” connections. Note: passive connections are NOT currently supported with the HPSS Parallel FTP Daemons. Note: this may cause other indeterminant problems and should NOT be specified. STRONGLY discouraged!

-C Disable the quote site setcos command.

-D str This is used to read the path to the “FTP Base Dir”. Default: /var/hpss/ftp

-F str This is used to specify the name of the ftpaccess file.
Default: ftpaccess.

-H The standard FTP Daemon places persons without “home directories” in the “/” directory. This may be unwise for security reasons. Specifying the -H option will deny access to clients without “home directories” and will create a syslog record indicating the problem.

-G Used to specify that the Authentication Manager is to be invoked for authentication. This cause the “hpss_option AMGR path/name” record to be read from the ftpaccess file. Refer: auth_manager(7).

-I Used to allow authentication based on a “Trusted” hosts file. The “trusted_hosts” file is in the etc sub-directory of the FTPBaseDir. This is HIGHLY discouraged for security reasons. The trusted_hosts file MUST NOT be writable by anyone other than the owner which should be root!

-K Used to specify an alternate HPSS keytab file. This option causes the “hpss_option KTAB string” record to be read from the ftpaccess file.

-L Used to read the “hpss_option LS rpcgroup” record from the ftpaccess file.

-N Used to read the “hpss_option SITE cellname” record from the ftpaccess file. NOT currently utilized! Do NOT Specify.

-P Used to read the “hpss_option PRINC principal” record from the ftpaccess file.

-R Enables Port Range Support to set the port range to be used by the Movers when connecting to the Parallel FTP Client for Parallel features.

-S Causes the DCE Registry to perform authentication and authorization for the HPSS FTP Daemon. This avoids the requirement for maintenance of the ftppasswd file.

-T Used to read the “hpss_option MTO #” record in the ftpaccess file. This option specifies a maximum timeout value to the HPSS FTP Daemons.

-U Specify to use udp ONLY. Problems have been observed in DCE and/or HPSS if tcp support is NOT disabled. HIGHLY Recommended.

-X Use the HPSS.homedir and HPSS.gecos Extended Registry Attributes (ERAs) from the DCE registry. This implies the -S option. If the client does NOT have the specified ERA the values will be the values obtained from the “home” and “description” attributes in the DCE Registry. Use the command “dcecp -c principal show {user} -all” command to determine if the HPSS.homedir and/or HPSS.gecos extended registry attributes are set for a customer. Use the

```
        “dcecp -c account show {user} -all” command to determine the
        “description” and “home” values for a customer.   Note: if the
        client is from a Trusted Cross Cell, the customer name will be in
        the form: /.../foreign.dce.cell/user.

EXAMPLES

hpssftp stream tcp nowait root /{path}/hpss_pftpd hpss_pftpd -hLUddd
hpssftpamgr stream tcp nowait root /{path}/hpss_pftpd_amgr hpss_pftpd_amgr \
            -ahGLSX

FILES

/etc/inetd.conf
/etc/services
/var/hpss/ftp/adm/hpss_ftpd.log
/var/hpss/ftp/adm/xferlog
/var/hpss/ftp/etc/ftpaccess
/var/hpss/ftp/etc/ftpbanner
/var/hpss/ftp/etc/ftppasswd
/var/hpss/ftp/etc/ftpusers
/var/hpss/ftp/etc/trusted_hosts
/users/no_home_dir
/{cellname}
/usr/lpp/hpss/bin/insif

RELATED INFORMATION

pftp_client(7), multinoded(7), auth_manager(7), syslogd.

Assorted TCP Wrapper applications

HPSS User’s Guide

HPSS System Administration Guide


SCCS

            @(#)92   1.3   man/pftpd.7, gen, 4.1.1   2/7/00   15:42:31
```

I.44 plu — Purge List Utility

```
NAME
     plu - List files that have a purge record in the HPSS metadata.

SYNTAX
     plu [-s <SC Id> | -all] [-show | -noshow] [-create | -access]
         [-l | -lonely | -lexp ] [-ascending | -descending] [-c <CDS>]
         [-p <Principal>] [-k <Keytab>] [-g <General>] [-? | -h] [-v]

DESCRIPTION
     “plu” is a HPSS System Maintenance Utility.  It is used to generate a
     list of full pathnames of files that have purge records in the HPSS
     metadata. A particular storage class may be specified, but plu defaults
     to listing files from all storage classes.  The listing can be ordered in
     ascending or descending order by either a file’s creation time or time of
     last access.  plu can also be used to list the purge-lock status of
     files, including expired locks.
     NOTE:
         -Options -g -c -p -k are not necessary if the
          /usr/lpp/hpss/config/hpss_env file has been sourced since the
          environment variables will then contain the appropriate information.
         -When sorting by last access time, plu uses the last time of user
          access, not necessarily the last time of HPSS access (through a
          migration to a 2nd level of disk, for example).

OPTIONS
     -s <SC Id>       A valid storage class id number to list files from.

     -v               Displays verbose output (progress is indicated).

     -? or -h         Displays usage/help screen.

     -all             Files will be listed from all storage classes
                      (this is the default).

     -show            List the storage class id number for each file
                      (this is the default when all storage classes are
                      being listed).

     -noshow          The storage class id number will not be listed
                      (this is the default when only one storage class is
                      being listed).

     -create          Files will be sorted by their creation times

     -access          Files will be sorted by their last access times
                      (this is the default).

     -l               List all files with purge records, marking purge
                      locked files with a “L” indicator, and
                      expired purge locked files with a “X” indicator.
```

```
     -lonly          List purge locked files, with a “L” indicator, and
                     expired purge locked files, with a “X” indicator, only.

     -lexp           List expired purge locked files only.  The “X” indicator
                     will not be printed.

     -ascending      Sort in ascending order. (this is the default)

     -descending     Sort in descending order.

     -g <General>    The pathname of the SFS generic server configuration
                     file. If this parameter is not specified, the
                     HPSS_CONFIG_SERVER environment variable will be used.

     -c <CDS>        The Cell Directory Service name which this utility will
                     run under. The Server CDS name of the SSM can be found
                     from SSM through: Admin, Configure HPSS,
                     Server Configuration screens.  If this parameter is not
                     specified, the HPSS_CDS_SSMSM environment variable will
                     be used.

     -p <Principal>  The server principal name which this utility will run
                     under.  Usually run as SSM.  The Principal name of the
                     SSM can be found from SSM through: Admin, Configure
                     HPSS, Server Configuration, Security Information
                     screens. If this parameter is not specified, the
                     HPSS_PRINCIPAL_SSM environment variable will be used.

     -k <Keytab>     The Keytab name which this utility will run with. The
                     Keytab name of the SSM can be found from SSM through:
                     Admin, Configure HPSS, Server Configuration,
                     Security Information screens.  If this parameter is not
                     specified, the HPSS_KEYTAB_FILE_SERVER environment
                     variable will be used.

EXAMPLES
     The following command will list files that have purge records for
     storage class 1 sorted from oldest to newest by their creation dates
     using /.:/encina/sfs/hpss2/serverconfig as the general server config
     SFS filename:

     $ plu -s 1 -create -g /.:/encina/sfs/hpss2/serverconfig
     /home/goodwinj/plutest/file1
     /home/goodwinj/plutest/file2
     /home/goodwinj/plutest/file8


     However, assuming that the appropriate environment variables are set,
     the following command will list all purge records (from all storage
     classes) while indicating purge-locked files and expired files:

     $ plu -l
```

```
      X       1  /home/goodwinj/plutest/expiredfile
      L       8  /it_test/test03/lockedfile
              1  /home/jgerry/regularfile

SEE ALSO

SCCS
     man/plu.7, gen, 4.1.1, 4.1.1.1   4/5/99   17:16:13
```

I.45 reclaim — HPSS Volume Reclaim Utility

```
NAME
     reclaim - HPSS Volume Reclaim Utility

SYNTAX
     reclaim [ -s SS Desc. Name -l PVL Descriptive Name
     -g HPSS Generic Server Path -c Utility Server(CDS) Name
     -p Utility Principal Name -t Transit File Name
     -k KeyTab Path ] < FILE

DESCRIPTION
     “reclaim” is a HPSS System Maintenance Utility. It is the
     second utility of a pair which compose the reclaim procedure.
     It is the utility which does the actual operations of
     deleting and creating SS volumes.  It reads standard
     input for candidates and then will delete and create
     the volume(Map, VV and PV(s)).  The deletes and creates
     are done in seperate transactions.  A recovery file
     is generated by reclaim to track a volume while it
     has been deleted but not yet created.  If the
     recovery/transit file exists when this utility is
     started, it will first read entries from this file
     and attempt to create them. The first utility in the
     pair is gen_reclaim_list.



OPTIONS
     -s <SS Desc. Name>    The descriptive name of the HPSS Storage Server
                           which manages the Volumes to be reclaimed.
                           The descriptive name of a storage server
                           can be found from SSM through: Admin, Configure
                           HPSS, Server Configuration screens.

     -l <PVL Desc. Name>   The descriptive name of the HPSS PVL Server
                           which manages the volumes to be reclaimed.
                           The descriptive name of a PVL server
                           can be found from SSM through: Admin, Configure
                           HPSS, Server Configuration screens.

     -g <Generic Name>     The pathname of the SFS generic server configuration
                           file.

     -c <CDS Name>         The Cell Directory Service name which this
                           utility will run under.  Usually run as SSM.
                           The Server CDS name of the SSM
                           can be found from SSM through: Admin, Configure
                           HPSS, Server Configuration screens.
```

-p <Principal Name>	The server principal name which this utility will run under. Usually run as SSM. The Principal name of the SSM can be found from SSM through: Admin, Configure HPSS, Server Configuration, Security Information screens.
-t <Transit Name>	The name of a file which will hold volume recovery information. Volume information is written to the file while it is deleted from HPSS. If reclaim fails before the volume is created the information is saved in this file and an attempt to create the volume will be made during the next run of the reclaim utility.
-k <Keytab Name>	The Keytab name which this utility will run with. Usually run as SSM. The Keytab name of the SSM can be found from SSM through: Admin, Configure HPSS, Server Configuration, Security Information screens.
FILE	Is a file containing a list of the volume ascii HPSS soids to be reclaimed. This list is normally generated through the gen_reclaim_list utility but can be generated with an editor.

EXAMPLE

The following command will reclaim the volumes specified in the file “recycle_list”. The volumes specified in the “recycle_list” must be managed by the storage server input. The file “transit_file” will contain any volumes deleted and not recreated.

```
$ reclaim -s ss_tape -l PVL -g /.:/encina/sfs/hpss/serverconfig
          -c /.:/hpss/ssmsm -p hpss_ssm -k /krb5/hpss.keytabs
          -t transit_file < recycle_list
```

The contents of the file recycle_list contains the soid for four volumes.

Example:

```
$ cat recycle_list
00122c08-4b6b-1137-b000-02608c2cd22f
007bf214738f1015b455000000000001
0020b8e0-4b8d-1137-b000-02608c2cd22f
007bf214738f1015b455000000000001
004b1658-3c2a-1119-a29e-02608c2cd22f
007bf214738f1015b455000000000001
00144fb0-4f79-112a-b83e-02608c2cd22f
007bf214738f1015b455000000000001
```

```
FILES


SEE ALSO
     reclaim.ksh(7), gen_reclaim_list(7), repack(7)

SCCS
     man/reclaim.7, gen, 4va   5/6/98   15:27:43
```

I.46 reclaim.ksh — HPSS Volume Reclaim Utility

```
# man/reclaim.ksh.7, gen, 4va   9/18/96   13:32:09
NAME
     reclaim.ksh - HPSS Volume Reclaim Utility

SYNTAX
     reclaim.ksh [ SS_Descriptive_Name Number_to_Reclaim Principal
                   CDS_path Keytab HPSS_Generic_path Working_Dir
                   [Storge_Class] ]

DESCRIPTION
     “reclaim.ksh” is a HPSS System Maintenance Utility. It is a script
     which manages the reclaim procedure.  The Reclaim process consists
     of two utilities: gen_reclaim_list and reclaim.  This script is
     used to automate the reclaim procedure and is invoked by SSM.
     It can also be run standalone to reclaim volumes.



OPTIONS
     SS_Descriptive_Name   The descriptive name of the HPSS Storage Server
                           which manages the Volumes to be reclaimed.
                           The descriptive name of a storage server
                           can be found from SSM through: Admin, Configure
                           HPSS, Server Configuration screens.

     Number_to_Reclaim     The number of empty volumes to reclaim.

     Principal             The server principal name which this utility
                           will run under.  Usually run as SSM.
                           The Principal name of the SSM
                           can be found from SSM through: Admin, Configure
                           HPSS, Server Configuration, Security Information
                           screens.

     CDS_path              The Cell Directory Service name which this
                           utility will run under.  Usually run as SSM.
                           The Server CDS name of the SSM
                           can be found from SSM through: Admin, Configure
                           HPSS, Server Configuration screens.

     HPSS_Generic_path     The pathname of the SFS generic server configuration
                           file.

     Keytab                The Keytab name which this utility
                           will run with.  Usually run as SSM.
                           The Keytab name of the SSM
                           can be found from SSM through: Admin, Configure
                           HPSS, Server Configuration, Security Information
                           screens.
```

```
     Working_Dirs          The Working directory to place report/recovery
                           files.

     Storge_Class          The storage class of the volumes to scan.
                           The storage classes supported can be found
                           from SSM through: Monitor, Storage Class List
                           screens.
```

EXAMPLE

```
     The following command will scan the volumes in storage class
     3 managed by the storage server “ss_tape” for 20 empty
     volumes.  The files generated by the reclaim procedure will
     be placed in the directory /tmp.

     $ reclaim.ksh ss_tape 20 hpss_ssm /.:/hpss/ssm /krb5/hpss.keytabs
             /.:/encina/sfs/hpss/d2.svs.serverconfig /tmp 3
```

FILES

```
     Created for a working directory of /tmp.


     CANDIDATE_LIST - The filename of the list of VV’s which
                      are candidates for reclaim.  This file
                      will contain the ascii VVID hpssoids.

              Formed: $WORK_DIR”/”$SS_DESC_NAME”_”$STORAGE_CLASS”_gen_list
                      /tmp/ss_tape_3_gen_list


     GEN_REP_LOG    - The report log file for the utility which
                      generates the list of VV reclaim
                      candidates.

              Formed: $WORK_DIR”/”$SS_DESC_NAME”_”$STORAGE_CLASS”_gen_log
                      /tmp/ss_tape_3_gen_log


     RECLAIM_RECOVERY_LOG - The file which will contain the VV(map,VV,PV(s))
                            metadata of deleted entries which were not
                            created.  The reclaim utiltity will attempt
                            to create on the next run.

              Formed: $WORK_DIR”/”$SS_DESC_NAME”_”$STORAGE_CLASS”_transit
                      /tmp/ss_tape_3_reclaim_log


     RECLAIM_REP_LOG - The report log file for the reclaim(2nd) utility.
```

```
              Formed: $WORK_DIR""$SS_DESC_NAME"_"$STORAGE_CLASS"_reclaim_log
                      /tmp/ss_tape_3_reclaim_log



SEE ALSO
     gen_reclaim_list(7), reclaim(7), repack(7)

SCCS
     @(#)reclaim.ksh.7  3.1   4/19/96   12:20:52
```

I.47 recover — Recover HPSS Data from Secondary Copy

NAME

recover - Used to recover HPSS data from secondary copies
in the event of media errors.

SYNTAX

```
recover [-r -c “Storage Class ID” | -x]
              [-p “Utility Principal Name”
               -d “Utility Server(CDS) Name”
               -k “Utility KeyTab Pathname”
               -h “HPSS generic server config file”
               -s “SS Generic Descriptive name”
               -b “BFS Generic Descriptive name”
               -n “Name Server Descriptive Name”]
             volume_id [volume_id,[....]]
```

DESCRIPTIONS

This command can be used to recover HPSS data from a secondary copy, or when no secondary copy exists, it can be used to clean up HPSS metadata and report what data has been lost. The recover command is executed with the names of one or more HPSS physical volumes. These volumes must be in the HPSS Physical Volume Library Format. The format consists of the six character cartridge name followed by two zeros. Volumes specified for one recovery session must all exist in the same storage class. All recovery utility status and error message are logged to a file in the /tmp directory. The file is named ‘recover.PID.DATE’. Where ‘PID’ is the recover process ID and ‘DATE’ is the date and time of the recovery.

This utility can be used to recover or cleanup storage resources for either tape or disk.

In general, this utility operates by determining the identifier for each bitfile that has damaged segments on the specified volumes. Each of these bitfiles is then opened in an exclusive mode and write/migration times are checked to verify that there is valid data at a duplicate copy level in the storage hierarchy. If valid copy data exists, the damaged segments are purged from the current level, and the valid data is then migrated or staged to the current level. If no valid copy data exists, the segments are purged from the current level and information is logged to indicate the loss of data from the storage level for each segment that was purged.

Running in the clean up mode (-x), this utility purges all segments for the specified volumes and reports file names and segments that were purged. For disk storage levels, it also reports whether the file was migrated successfully to the next storage level after the last update.

This command is implemented as a Korn shell script that makes use of several different assistance commands: repack, recover_pvs, and recover_segs.

OPTIONS

-r This indicates that a repack of the specified volumes should be attempted before the recover process. This mode simply utilizes the repack utility to report the segments that can't be recovered and uses this input to drive the recovery process. First, 'repack' is run on the volumes, then any segments that can't be repacked are used as input to the recovery processing. For each of these segments, 'recover' tries to locate a valid segment from a secondary copy. If a valid secondary copy is located the damaged segment is recovered from the copy. If no secondary copy is found, the segment is reported as lost and the corresponding metadata is deleted. Since, part of this recovery includes a repack of of the specificed volume, there must be enough free space in the storage class to repack all the associated virtual volumes.

-x The prescence of this option indicates that all storage and bitfile segments that point to the specified volume(s) will be removed and the storage resources deleted. After the storage resources are deleted the volumes are exported from HPSS. This option is typically used when a volume (most often disk) is damaged and valid data exists at a lower level in the hierarchy. This option should not be used on a storage level that is a member of a copy hierarchy, since it essentially deletes one of the copies that will not be repopulated until data is modified. It is important to note that none of the file names that correspond to the deleted resources will be removed by the recover utility.

Run without either of the options above (-r|-x), the utility attempts to recover all segments on the specified volumes from a secondary copy. This mode is useful when trying to repopulate a damaged copy level, for a tape that can't be mounted, with data from another copy.

-c "Storage Class" This is the ID number of the storage class that contains the suspected bad volumes. This number can be acquired from the SSM Storage Class Information screen. This option must be specified only when running repack as part of the recover processing.

All other options can be specified via the command line or the HPSS Environment Table. If the HPSS_ROOT environment variable is set, the following information will be obtained from the environment

table. These values can be overridden using the command line options.

-p “DCE Principal Name” This is the DCE principal name under which the recover command will run. Typically, this value should be a trusted principal (defaults to the HPSS_PRINCIPAL_SSM value specified in the environment file).

-d “CDS Pathname” This is the CDS path name of the server that the recover command will run as (defaults to the HPSS_CDS_SSMSM value specified in the environment file).

-k “KeyTab Pathname” This is the security keytab file path name that the recovery command will use to authenticate its identity with DCE (defaults to the HPSS_KEYTAB_FILE_SERVER value specified in the environment file).

-h “Generic Configfile” This is the CDS path name of the HPSS generic server configuration file (defaults to the HPSS_CONFIG_SERVER value specified in the environment file).

-s “Storage Server Descriptive Name” This is the descriptive name of the HPSS storage server (defaults to “Tape Storage Server”).

-b “Bitfile Server Descriptive Name” This is the descriptive name of the HPSS bitfile server (defaults to “BFS”).

-n “Name Server Descriptive Name” This is the descriptive name of the HPSS name server (defaults to “Name Server”).

EXAMPLE

1. Suppose you have two physical volumes (AA000100 and AA010000) that are contained within one or more virtual volumes in a storage class (ID number 3). To attempt a data recovery from theses volumes, enter:

```
recover -r -c 3 AA000100 AA010000
```

If segments for the 3 files below can’t be repacked and the storage class has a secondary copy, then the following type of log messages would be expected:

```
========= Trying to recover bitfile =========
003fcadc-53fb-10cf-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/test1 )
========= Trying to recover bitfile =========
163001bc-2274-10ef-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/test2 )
```

```
========= Trying to recover bitfile =========
0786ab2c-156b-1047-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/test3 )
========= Trying to recover bitfile =========
 .
 .
 .

========= Deleting VV =========
0723ab2c-176b-11d2-aab3-0004ac4915ba
00226b52 efec 11d1 9e bd 00 00 00 00 00 04
========= Deleting VV =========
0456ab2c-2274-10ef-8c81-02608c2f971f
00226b52 efec 11d1 9e bd 00 00 00 00 00 04
```

Note: The numeric values above represent the SOIDs for each of the bitfiles. The first line is the UUID of the bitfile and the second line is the server dependent information.

If the storage class is of type ‘tape’, on level 1 of the hierarchy, and has no secondary copy, then the following type of log messages would be expected:

```
========= Can’t recover bitfile =========
003fcadc-53fb-10cf-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/test1 )
lost segments from storage level 1
offset = 0    , length = 32768
offset = 32768, length = 32768
offset = 65536, length = 32768
 .
 .
 .

========= Can’t recover bitfile =========
163001bc-2274-10ef-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/test2 )
lost segments from storage level 1
offset = 0    , length = 32768
offset = 32768, length = 32768
offset = 65536, length = 32768
 .
 .
 .

========= Can’t recover bitfile =========
0786ab2c-156b-1047-8c81-02608c2f971f
```

```
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/test3 )
lost segments from storage level 1
offset = 0    , length = 32768
offset = 32768, length = 32768
offset = 65536, length = 32768
 .
 .
 .

========= Deleting VV =========
0723ab2c-176b-11d2-aab3-0004ac4915ba
00226b52 efec 11d1 9e bd 00 00 00 00 00 04
========= Deleting VV =========
0456ab2c-2274-10ef-8c81-02608c2f971f
00226b52 efec 11d1 9e bd 00 00 00 00 00 04
```

2. Suppose you have two physical volumes (AA000100 and AA010000) that are contained within one or more virtual volumes. To recover all the segments on these volumes from a secondary copy, enter:

```
recover AA000100 AA010000
```

Note: The -r flag is missing indicated that repack in not to be run. Instead, the recovery generates a list of all segments on the volumes and then attempt to recover them from valid secondary copies.

If secondary copies are enabled for the COS, the following type of log messages would be expected:

```
========= Trying to recover bitfile =========
003fcadc-53fb-10cf-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/test1 )
========= Trying to recover bitfile =========
163001bc-2274-10ef-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/test2 )
========= Trying to recover bitfile =========
0786ab2c-156b-1047-8c81-02608c2f971f
00336b52 4631 10cf 00 00 00 02
path ( Fileset24: /home/bill/test3 )
========= Trying to recover bitfile =========
 .
 .
 .

========= Deleting VV =========
0723ab2c-176b-11d2-aab3-0004ac4915ba
00226b52 efec 11d1 9e bd 00 00 00 00 00 04
```

```
========= Deleting VV =========
0456ab2c-2274-10ef-8c81-02608c2f971f
00226b52 efec 11d1 9e bd 00 00 00 00 00 04


3. Suppose you have a physical disk volume (VOL00100) that is
contained within one or more virtual volumes. To attempt to remove
bitfile and storage segments that point to this volume (and export
the volume from HPSS), enter:


Export HPSS_ROOT=/usr/lpp/hpss
recover -x -s “Disk Storage Server” VOL00100

If the files on the disk have been migrated to the next level
since the last update, the following type of log messages would
be expected:

========= Cleaning up segments for bitfile =========
b3c986a0-5f55-11d2-aab3-0004ac4915ba
c8d153f8 efec 11d1 9e bd 00 00 00 00 00 04
path ( Fileset24: /home/bill/test1 )
lost file from storage level 0:
this file has been migrate since the last update!
========= Cleaning up segments for bitfile =========
b3c986a0-5f55-11d2-aab3-0004ac4915ba
c8d153f8 efec 11d1 9e bd 00 00 00 00 00 04
path ( Fileset24: /home/bill/test2 )
lost file from storage level 0:
this file has been migrate since the last update!
========= Cleaning up segments for bitfile =========
b3c986a0-5f55-11d2-aab3-0004ac4915ba
c8d153f8 efec 11d1 9e bd 00 00 00 00 00 04
path ( Fileset24: /home/bill/test3 )
lost file from storage level 0:
this file has been migrate since the last update!
========= Cleaning up segments for bitfile =========
 .
 .
 .

========= Deleting VV =========
0723ab2c-176b-11d2-aab3-0004ac4915ba
00226b52 efec 11d1 9e bd 00 00 00 00 00 04


If the files on the disk haven’t been migrated since the last
update, the following type of log messages would be expected:

========= Cleaning up segments for bitfile =========
b3c986a0-5f55-11d2-aab3-0004ac4915ba
c8d153f8 efec 11d1 9e bd 00 00 00 00 00 04
path ( Fileset24: /home/bill/test1 )
```

```
    lost file from storage level 0:
    this file has NOT been migrate since the last update!
    ========= Cleaning up segments for bitfile =========
    b3c986a0-5f55-11d2-aab3-0004ac4915ba
    c8d153f8 efec 11d1 9e bd 00 00 00 00 00 04
    path ( Fileset24: /home/bill/test2 )
    lost file from storage level 0:
    this file has NOT been migrate since the last update!
    ========= Cleaning up segments for bitfile =========
    b3c986a0-5f55-11d2-aab3-0004ac4915ba
    c8d153f8 efec 11d1 9e bd 00 00 00 00 00 04
    path ( Fileset24: /home/bill/test3 )
    lost file from storage level 0:
    this file has NOT been migrate since the last update!
    ========= Cleaning up segments for bitfile =========
     .
     .
     .

    ========= Deleting VV =========
    0723ab2c-176b-11d2-aab3-0004ac4915ba
    00226b52 efec 11d1 9e bd 00 00 00 00 00 04


    This will also delete storage resource associated with these
    volumes.


FILES

    /usr/lpp/hpss/config/hpss_env     Specifies the HPSS environment file.

    /usr/lpp/hpss/bin/recover         Specifies the command file.

    /usr/lpp/hpss/bin/repack          Specifies the repack command file.

    /usr/lpp/hpss/bin/recover_segs    Specifies the assistance command file.

    /usr/lpp/hpss/bin/recover_pvs     Specifies the assistance command file.



SEE ALSO

    The repack command.

    HPSS System Administration Guide: HPSS Special Intervention Procedures
    provides information about recovering HPSS files from damaged volumes.


SCCS

    man/recover.7, gen, 4va  11/20/98  15:53:37
```

I.48 remove — HPSS Volume Remove Utility

NAME

remove - HPSS Volume Remove Utility

SYNTAX

```
remove [ -s SS Desc. Name -l PVL Desc. Name -g HPSS Generic Server
Path -c Utility Server(CDS) Name -p Utility Principal Name
-t  Storage Class ID -f Transit File Name -n No Export
-k KeyTab Path ] < FILE
```

DESCRIPTION

"remove" is a HPSS System Maintenance Utility. Its intended to be used during the procedure of replacing one storage technology with another. This utility will remove a volume in a retired state from the storage server and optionally export the volume from the PVL. If the volume is exported from the PVL all volume PVL and PVR metadata will be removed and the cartridge will be ejected(in the case of a robot). In order for a volume to be removed, the map must be in retire state, the number of active segments must not be greater than zero and there must not be any active data on the volume.

Volume candidates for removal can be specified via two methods. The first method allows a list of Storage Server physical volumes to be specified through standard input. This is the most selective way to remove volumes as specific volumes can be chosen. The second method is to specify the storage class of the volumes to be removed. This is a more general approach and will allow for a large number of volumes to be retired during a single run.

The storage server resource deletes are done in seperate transactions. If the PVL export option is specified, a recovery file is generated by remove to track a volume while it has been deleted but not yet exported. This is needed because the storage server resource delete results in the clearing of the allocated client field in the PVL metadata. A export of a volume with the allocated client set will result in an error. If the recovery/transit file exists when this utility is started, it will first read entries from this file and attempt to export them.

OPTIONS

```
-s <SS Desc. Name>    The descriptive name of the HPSS Storage Server
                      which manages the Volumes to be removed.
                      The descriptive name of a storage server
                      can be found from SSM through: Admin, Configure
                      HPSS, Server Configuration screens. If set, the
                      environment variable HPSS_DESC_SSTAPE will be used
                      if an argument is not specified on the command line.

-l <PVL Desc. Name>   The descriptive name of the HPSS PVL
                      which manages the Volumes to be removed.
                      The descriptive name of a storage server
                      can be found from SSM through: Admin, Configure
                      HPSS, Server Configuration screens.

-g <Generic Name>     The pathname of the SFS generic server configuration
                      file. If set, the environment variable
                      HPSS_CONFIG_SERVER will be used if an argument is
                      not specified on the command line.

-c <CDS Name>         The Cell Directory Service name which this
                      utility will run under.  Usually run as SSM.
                      The Server CDS name of the SSM
                      can be found from SSM through: Admin, Configure
                      HPSS, Server Configuration screens.
                      If set, the environment variable
                      HPSS_CDS_SSMSM will be used if an argument is
                      not specified on the command line.

-p <Principal Name>   The server principal name which this utility
                      will run under.  Usually run as SSM.
                      The Principal name of the SSM
                      can be found from SSM through: Admin, Configure
                      HPSS, Server Configuration, Security Information
                      screens. If set, the environment variable
                      HPSS_PRINCIPAL_SSM will be used if an argument is
                      not specified on the command line.

-t <Storage Class ID> The storage class id of the volumes to be
                      removed.  If volumes are to be selected
                      via standard input, do not set.
                      The Storage Class id can be found from SSM
                      through: Monitor, Storage Class List,
                      Detailed Info screens.

-f <Transit Name>     The name of a file which will hold volume
                      recovery information.  Volume information
                      is written to the file while it is deleted
                      from the storage server. If the PVL export
                      fails the information is saved
                      in this file and an attempt to export the
                      volume will be made during the next run of
                      the remove utility.
```

```
     -k <Keytab Name>      The Keytab name which this utility
                           will run with.  Usually run as SSM.
                           The Keytab name of the SSM
                           can be found from SSM through: Admin, Configure
                           HPSS, Server Configuration, Security Information
                           screens. If set, the environment variable
                           HPSS_KEYTAB_FILE_SERVER will be used if an argument
is
                           not specified on the command line.

     -n                    If specified, the volumes will be removed from
                           the storage server but will NOT be exported
                           from the PVL.  The volumes will remain in
                           PVL and PVR metadata.

     FILE                  Is a file containing a list of Storage Server
                           physical volume names to remove.

EXAMPLE
     The following command will remove(but not export) the
     Storage Server physical volumes specified in the file
     “remove_list”.
     The volumes listed in the “remove_list” must be managed
     by the storage server specified input argument.


     $ remove -s ss_tape -l PVL -g /.:/encina/sfs/hpss/serverconfig
              -c /.:/hpss/ssmsm -p hpss_ssm -k /krb5/hpss.keytabs
              -f transit_file -n < remove_list

                           OR (using environment variables and ksh)

     $ . /usr/lpp/hpss/config/hpss_env

     $ remove -l PVL -f transit_file -n < remove_list



     The contents of the file remove_list contains the SS PV names for
     four volumes.
       Example:
       $ cat remove_list
       BCK05900
       BUD13200
       BCK06000
       BCK06100




     The command will remove(and export) volumes in retire
     map state which are managed by the Storage server
     ss_tape with the storage class 5. The PV names for
```

```
     any export failures will be placed in the file
     transit_file.


     $ remove -s ss_tape -l PVL -g /.:/encina/sfs/hpss/serverconfig
               -c /.:/hpss/ssmsm -p hpss_ssm -k /krb5/hpss.keytabs
               -f transit_file -t 5

                            OR (using environment variables and ksh)

     $ . /usr/lpp/hpss/config/hpss_env

     $ remove -l PVL -f transit_file -t 5


FILES


SEE ALSO
     retire(7), repack(7)

SCCS
     man/remove.7, gen, 4va   11/23/98   11:37:34
```

I.49 repack — HPSS Volume Repack Utility

```
NAME
     repack - HPSS Volume Repack Utility

SYNTAX
     repack [ -g SS Desc. Name -h HPSS Generic Server Path
     -d Utility Server(CDS) Name -p Utility Principal Name -k KeyTab Path
     -c Storage Class -n Number -s Source List -t Target List
     -v Repack Threshold -r Bad Media -a Family ID -f -m -o]

DESCRIPTION
     “repack” is a HPSS System Maintenance Utility.  This utility
     is normally invoked through SSM for removable media(i.e. tape)
     but can also be initiated from the command line.  Attempting
     to repack a specific volume currently must be initiated through
     command line execution.

     In the case of fixed media(i.e. disk), repack must be initiated
     via command line.  Automatic selection of volumes is not supported
     for disk, specific volumes must be specified.

     Repack’s function is to take storage segments from sparse volumes and
     move them to other volumes in the same storage class and family.
     This will result in a net increase in the number of available volumes
     as the sparse volumes are emptied, reclaimed and returned to service.

     Normally only volumes in the EOM(end of media) state are
     selected for repack.  If it is desired to select a volume
     which hasn’t reached EOM, the volume must be specified
     through the -s option.  To insure that data is not moved
     to the source volume while repacking(or shortly after),
     set the Volume Map Administrative state to “locked” before
     executing repack.  This is especially significant for
     fixed media(i.e. disk).

     The manner in which volumes are selected is controllable through
     the command line arguments.  If one or more specific volumes are
     desired to be repacked, the Storage Server physical volume names are
     placed in a file and the file is specified on the command line.
     This could be of use when a volume should be cleared due to i/o errors,
     physical damage, replacing media, specific order policy of repack, etc.
     Target(destination) volumes can also be specified in a similar fashion.
     Repack will select source volumes for removable media if no source names
     file is provided on the command line.  In this case the number
     of volumes and storage class are required command line arguments.  The
     family can be specified optionally.

     If certain command line arguments are not specified, environment
     variables found in /usr/lpp/hpss/config/hpss_env will be used(if
     set).
```

OPTIONS

-g <SS Desc. Name>	The descriptive name of the HPSS Storage Server which manages the Volumes to be reclaimed. The descriptive name of a storage server can be found from SSM through: Admin, Configure HPSS, Server Configuration screens. If set, the environment variable HPSS_DESC_SSTAPE will be used if an argument is not specified on the command line.
-h <Generic Name>	The pathname of the SFS generic server configuration file. If set, the environment variable HPSS_CONFIG_SERVER will be used if an argument is not specified on the command line.
-d <CDS Name>	The Cell Directory Service name which this utility will run under. Usually run as SSM. The Server CDS name of the SSM can be found from SSM through: Admin, Configure HPSS, Server Configuration screens. If set, the environment variable HPSS_CDS_SSMSM will be used if an argument is not specified on the command line.
-p <Principal Name>	The server principal name which this utility will run under. Usually run as SSM. The Principal name of the SSM can be found from SSM through: Admin, Configure HPSS, Server Configuration, Security Information screens. If set, the environment variable HPSS_PRINCIPAL_SSM will be used if an argument is not specified on the command line.
-k <Keytab Name>	The Keytab name which this utility will run with. Usually run as SSM. The Keytab name of the SSM can be found from SSM through: Admin, Configure HPSS, Server Configuration, Security Information screens. If set, the environment variable HPSS_KEYTAB_FILE_SERVER will be used if an argument is not specified on the command line.
-c <Storage Class>	The storage class of the volumes to scan. The storage classes supported can be found from SSM through: Monitor, Storage Class List screens. This option is ignored if the -s option is supplied, but must be supplied if -s is omitted.
-n <Number>	The number of volumes to repack. If a source file is not specified, the utility will attempt to select this number of VVs which meet the

selection criteria to repack. If a source file is specified, this argument is not required.

-s <Source List> The pathname to a file which contains the Storage Server physical volume names of the Source volumes to repack. This option allows specific volumes to be selected. It may prove useful when attempting to clear damaged or failing volumes(cartridges, disks, etc.). It is a required argument when repacking fixed media (disks).

-t <Target List> The pathname to a file which contains the Storage Server physical volume names of the Target volumes to repack to. This option allows the destination of data moved from source volumes to be selected. The target specification is a hint passed to the Storage Server. Under some conditions the Storage Server will move a segment to a volume other than the target. Repack will report this condition when the segment is moved and will also report the sum of segments moved to non-target specified volumes.

This argument is optional. If it is not provided, the Storage Server will select destination volumes automatically. This procedure is preferred.

-v <Threshold> The repack threshold value. When a volume's assigned space drops below this percentage of the estimated size of a volume in the storage class, the VV is a candidate for repack. The default value is 40%.

-a <Family ID> Family ID value. Specify the family id if you want to restrict repack candidates to a specific family, and repack is selecting the candidates (-s not specified). If not specified, volumes in the specified storage class (-c) in all families will be selected.

-f Force, this flag causes repack to select source VVs that are above the repack threshold, which would otherwise be skipped.

-r Display to standard output, any segment move error encountered for a VV move. This option is in place as an adjunct to the NSL UniTree conversion process. When this option is specified, repack will expect input volumes to be in binary SOID form.

-m Volumes in "retired" state will be repacked rather than volumes at EOM. This option is

used during the process of replacing a removable media type with another. It is used in conjunction with the retire and remove utilities.

-o Volumes that are on checked-out (shelved) tapes will be included in the repack. By default, volumes on checked-out tapes are not included.

EXAMPLE

The following command will repack 5 removable media(tape) volumes specified in the file “vv_repack_file”. The state of the volumes will NOT be checked for EOM. Set the Volume Map state to EOM or Administrative state to “locked” before running repack in order to insure the volumes are read-only. Target volumes will be automatically selected by the storage server.

Contents of the file vv_repack_file:

```
cat vv_repack_file
BUD13200
BUD13300
BUD13400
BUD13500
BUD13600
```

```
$ repack -g “Tape Storage Server” -h /.:/encina/sfs/hpss/serverconfig
          -d /.:/hpss/ssmsm -p hpss_ssm -k /krb5/hpss.keytabs
          -s vv_repack_file
```

OR (using environment variables)

```
$ . /usr/lpp/hpss/config/hpss_env

$ repack -s vv_repack_file
```

The following command will repack 1 fixed(disk) volume specified in the file disk_repack.src, managed by storage server “Disk Storage Server” to the specified target volume(in disk_repack.tar). The source volume map Administrative state should be set “locked” before repack is invoked. Segments are not guaranteed to be moved to the target.

Contents of the file disk_repack.src:

```
cat disk_repack.src
SSA01000
```

Contents of the file disk_repack.tar:

```
cat disk_repack.tar
SSA07100
```

```
     Using environment variables(other than Storage Server)

     $ . /usr/lpp/hpss/config/hpss_env


     $ repack -g “Disk Storage Server” -s ./disk_repack.src
              -t ./disk_repack.tar


     The following command will repack 10 “retired” removable media(tape)
     volumes with less than 20% data in use in storage class 3.  The
     target volumes will be selected automatically by repack.

     Using environment variables

     $ . /usr/lpp/hpss/config/hpss_env

     $ repack -n 10 -c 3 -v 20 -m



     The following command will repack 5 removable media (tape)
     volumes in storage class 3 and family 7 to volumes of the same
     class and family, selected by the storage server.

     Using environment variables

     $ . /usr/lpp/hpss/config/hpss_env

     $ repack -n 5 -c 3 -a 7

FILES


SEE ALSO
     reclaim.ksh(7), gen_reclaim_list(7), reclaim(7), retire(7), remove(7)

SCCS
      @(#)88   3.14   man/repack.7, gen, 4.1.1, 4.1.1.1   6/17/99  10:06:48
```

I.50 retire — HPSS Volume Retire Utility

```
NAME
     retire - HPSS Volume Retire Utility

SYNTAX
     retire  [ -s SS Desc. Name -g HPSS Generic Config  -c Utility
     Server Name -p Utility Principal Name -k KeyTab Path
     -t Storage Class ID -m Map State ] [ < FILE ]


DESCRIPTION
     “retire” is an HPSS System Maintenance Utility. Its
     intended use is to facilitate replacing one storage
     technology with another by changing the MapState field
     of obsolete technology physical volumes to MAP_RETIRED.
     Volumes in retired state will not be written by the system.
     Retired volumes may be removed from the system by repacking
     all of the useful information on them to other volumes,
     leaving the retired volumes empty.  These volumes can then
     be removed from the system with the “remove” utility.

     Volume candidates for retirement can be specified
     via two methods.  The first method allows a list
     of Storage Server physical volume names to be
     specified through standard input. This is the most
     selective way to retire volumes as specific volumes
     can be chosen.

     The second method is to specify both the storage
     class and map state of the volumes to be retired.
     This is a more general approach and will allow
     for a large number of volumes to be retired
     during a single run.



OPTIONS
     -s <SS Desc. Name>    The descriptive name of the HPSS Storage Server
                           which manages the Volumes to be retired.
                           The descriptive name of a storage server
                           can be found from SSM through: Admin, Configure
                           HPSS, Server Configuration screens.
                           The default value for this argument is the value
                           of the system environment variable
                           HPSS_DESC_SSTAPE.

     -g <Generic Name>     The pathname of the SFS generic server
                           configuration file.  The default value for this
                           argument is the value of the system environment
```

```
                        variable HPSS_CONFIG_SERVER.

     -c <CDS Name>      The Cell Directory Service name which this
                        utility will run under, usually SSM.
                        The Server CDS name of the SSM
                        can be found from SSM through: Admin, Configure
                        HPSS, Server Configuration screens.
                        The default value of this argument is the value
                        of the system environment variable HPSS_CDS_SSMSM.

     -p <Principal Name> The server principal name which this utility
                        will run under, usually SSM.  The Principal name
                        of the SSM can be found from SSM through: Admin,
                        Configure HPSS, Server Configuration,
                        Security Information screens.  The default value
                        of this argument is the value of the system
                        environment variable HPSS_PRINCIPAL_SSM.

     -k <Keytab Name>   The Keytab path name which this utility
                        will run with.  The default value of this argument
                        is the value of the system environment variable
                        HPSS_KEYTAB_FILE_SERVER.

     -t <Storage Class ID> The storage class id of the volumes to be
                        retired.  If this argument is supplied,
                        the -m argument must also be supplied.
                        The Storage Class id can be found from SSM
                        through: Monitor, Storage Class List,
                        Detailed Info screens.

     -m <Map State>     The map state of the volumes to retire.
                        Legal values are:

                                MAP_FREE
                                MAP_EOM
                                MAP_EMPTY

     FILE               Is a file containing a list of the Storage
                        Server PV names of volumes to be retired,
                        one to a line.  The PV names must be in HPSS
                        internal format, i.e. they must be 8 character
                        names -- the first six are the conventional
                        PV name -- the last two characters must be “00”.

                        Note:  If the -t and -m arguments are not
                        supplied, and “File” is not supplied on
                        the command line, the program will read
                        standard input.


EXAMPLE
     The following command will retire the volumes specified
     in the file “retire_list”.  The volumes specified in the
```

```
     "retire_list" must be managed by the storage server
     specified argument.


     $ retire  -s ss_tape -g /.:/encina/sfs/hpss/serverconfig
               -c /.:/hpss/ssmsm -p hpss_ssm -k /krb5/hpss.keytabs
               < retire_list

     The contents of the file retire_list contains the Storage
     Server physical volume names for 4 virtual volumes.
     four volumes.
       Example:
       $ cat retire_list
       BCK56900
       BUD13000
       BUD13500
       BCK65000



     The following command will retire volumes managed by the Storage
     server ss_tape with the storage class 5 and in a map
     state of MAP_EOM.


     $ retire  -s ss_tape -g /.:/encina/sfs/hpss/serverconfig
               -c /.:/hpss/ssmsm -p hpss_ssm -k /krb5/hpss.keytabs
               -t 5 -m MAP_EOM

     The command line can be simplified by use of the system environment
     variables.  These can be used to supply values for the -s, -g, -c,
     -p and -k arguments.  An example using the Bourne or Korn shell is:

        . /usr/lpp/hpss/config/hpss_env  (sources the HPSS environment)

        retire -t 12 -m MAP_FREE

FILES


SEE ALSO
     remove(7), repack(7)

SCCS
     man/retire.7, gen, 4va   11/23/98   13:54:41
```

I.51 scrub — Cleansing and Removal of Undesirable Bugs

NAME

scrub - Interactive interfaces to HPSS Client APIs

SYNTAX

scrub

DESCRIPTION

This utility provides an interactive, command-line interface in using and exercising the HPSS client application programming interfaces (APIs). It was originally developed to aid in testing HPSS. As such, HPSS administrators and users may find it helpful in local testing of HPSS features.

The user should be logged into DCE prior to invoking scrub. Note that some commands (e.g., “migrate”) may require special permissions with the Bitfile Server.

COMMANDS

The following is a summary list of commands available in scrub:

access	Checks for accessibility of named file
cd	Change current working directory
chacct	Change account of a file/directory
changecos	Change a bitfile’s class of service
chgrp	Change group id of a file/directory
chmod	Change file/directory permissions
chown	Change owner id of a file/directory
close	Close file
date	Toggle display of timestamp before executing each command
dump	Dump file metadata
echo	Turn echo of commands on or off
fclear	Create a hole in the open file
fmigrate	Migrate open file
fpurge	Purge open file
fstat	Display stats on open file
ftruncate	Truncate open file
getattrs	Display file attributes
hash	Control display of character for each read or write
help	Display list of commands or command options
hints	Set class-of-service hints
ls	Listing contents of a directory
lseek	Display/set offset
migrate	Migrate file
mkdir	Create a directory
mkuserdir	Make a home directory for a user
open	Open a file
purge	Purge file
pwd	Print current working directory
query	Display internal variables

```
quit          Terminate program
read          Read data
rename        Rename file/directory
rmdir         Remove a directory
scan          Scan a file and display format information
setuid        Set the UID for current user
stage         Stage data in a file
stat          Display stats on file
truncate      Truncate file
umask         Set umask
unlink        Remove file/directory
write         Write data

   Notes:  - To get help on a specific command, enter “help <command>”
           - Byte values may be specified using kb, mb, and gb (e.g., 10mb)
```

USAGE

```
The detailed usage of each scrub command is shown below:

access <path> { [r][w][x] | exists }
   <path>          Path to check access of
   r               Check for read access
   w               Check for write access
   x               Check for search/execute access
   exists          Check for existence of pathname

cd [<path>]
   <path>          If blank, changes directory to root/login directory

chacct <path> <new-acct-id>
   <path>          Name of file/directory to change
   <new-acct-id>   Id of new account

changecos <path> <cos-id>
   <path>            Path of file to change class of server of
   <cos-id>          New class of service id

chgrp <path> <new-group-id>
   <path>          Name of file/directory to change
   <new-owner-id>  Id of new group

chmod <path> <mode>
   <path>          Name of file/directory to change
   <mode>          New permission mode (in octal)

chown <path> <new-owner-id>
   <path>          Name of file/directory to change
   <new-owner-id>  Id of new owner

close

date
```

```
dump <path> [ bfs-config <bfs-config-metadata-file> ]
            [ ss-config <ss-config-metadata-file> ]
            [ output <output-file> ]
            [ all | [bfdesc] [slevel] [tapeseg] [diskseg] [diskmap] ]
   <bfs-config-metadata-file>
                  BFS server configuration metadata file path
   <ss-config-metadata-file>
                  SS server configuration metadata file path
   <output-file>  File to write to (default: stdout)
   all            Dump all bitfile metadata (default)
   bfdesc         Dump bitfile descriptor
   slevel         Dump storage level statistics
   tapeseg        Dump bitfile tape segments
   diskseg        Dump bitfile disk segments
   diskmap        Dump bitfile disk maps

echo [ on | off ]
   on             Turns echo of input commands on
   off            Turns echo of input commands off
   No arguments prints the current setting

fclear <bytes>
   <bytes>        Length of hole to create starting at current offset

fmigrate <level> [ purge ] [ force ]
   <level>        Storage level to migrate from
   purge          Purge data after migration
   force          Migrate even when copy of data exists at lower level

fpurge { all | <offset> <length> } <level>
   all            Indicates to purge all data at specified level
   <offset>       Offset position of data to purge
   <length>       Number of bytes to purge
   <level>        Storage level to purge data from

fstat

ftruncate <new-length>
   <new-length>   New file length (in bytes)

getattrs <path>
   <path>         Name of file/directory to get attributes of

hash [ on | off ]
   on             Turns hash marks on
   off            Turns hash marks off
   No arguments prints the current setting

help [<command>]
   <command>      If present, shows command options

hints [ cos-id <id> [<priority>] |
```

```
              cos-name <name> [<priority>] |
              transfer-rate <bytes/sec> [<priority>] |
              avg-latency <secs> [<priority>] |
              min-filesize <bytes> [<priority>] |
              max-filesize <bytes> [<priority>] |
              access-freq { hourly | daily | weekly | monthly |
                            archive } [<priority>] |
              read-ops { read | random | parallel | sequential }*
                         [<priority>] |
              write-ops { write | write-many | append | random |
                          parallel | sequential }* [<priority>] |
              stage on-open [ async ] [<priority>] |
              clear ]
   cos-id          Sets the class of service id and priority
   cos-name        Sets the class of service name and priority
   transfer-rate   Sets the transfer rate and priority
   avg-latency     Sets average latency (in secs) and priority
   min-filesize    Sets minimum filesize (in bytes) and priority
   max-filesize    Sets maximum filesize (in bytes) and priority
   access-freq     Sets access frequency and priority
   read-ops        Sets read operations (one or more) and priority
   write-ops       Sets write operations (one or more) and priority
   stage           Sets staging code and priority
   clear           Clears/resets hints information

   <priority>      Can be one of the following:
                     none
                     low
                     desired
                     highly-desired
                     required (default)

   No arguments print current settings

ls [-l] [<path>]
   -l              Prints long listing
   <path>          If blank, lists current directory

lseek [+<offset-bytes> | -<offset-bytes> | =offset-bytes> |
           eof+<offset-bytes> | eof-<offset-bytes>]
   +<offset-bytes>     Move offset forward from current position
   =<offset-bytes>     Move offset backward from current position
   +<offset-bytes>     Move offset to an absolute position
   eof+<offset-bytes>  Move offset forward from end of file
   eof-<offset-bytes>  Move offset backward from end of file

migrate <path> <level> [ purge ] [ force ]
   <path>          File to migrate
   <level>         Storage level to migrate from
   purge           Purge data after migration
   force           Migrate even when copy of data exists at lower level

mkdir <path> <mode>
```

```
   <path>          Directory to create
   <mode>          Permissions for new directory (e.g., 775)

mkuserdir <path> <uid> <gid>
   <path>          User's HOME directory
   <uid>           UID of user
   <gid>           GID of user

open <path> <oflags>
   <path>          File to open
   <oflags>        Open flags (e.g., rwc), which can be:
                     r - Read
                     w - Write
                     a - Append
                     c - Create
                     e - Exclusive
                     t - Truncate
   Note that if hints have been set, they are used

purge <path> { all | <offset> <length> } <level>
   <path>          File to purge
   all             Indicates to purge all data at specified level
   <offset>        Offset position of data to purge
   <length>        Number of bytes to purge
   <level>         Storage level to purge data from

pwd

query

quit

read { <total-bytes> | all } [ <buffersize> ]
                                  [ format=<format> | nocheck ]
    read <buffersize> <iterations>x [ format=<format> ]

   <total-bytes>     Specifies total number of bytes to read
                     ("all" will read to EOF)
   <buffersize>      Number of bytes to read each time
   <format>          Specifies expected format of data to be read:
                       0  - expecting binary zeros
                       1  - expecting binary ones
                       n# - expecting values equal to "n" times offset
   nocheck           No data format checking should be done (for best
                      performance)
   <iterations>      Number of times to read <buffersize>

rename <current-path> <new-path>
   <current-path>  Current path to rename
   <new-path>      Target pathname

rmdir <path>
   <path>          Directory to remove
```

```
scan <path> [<buffersize>]
   <path>            Path of file to scan
   <buffersize>      Number of bytes per read (default is 1048576)

setuid <uid>
   <uid>           New UID (e.g. zero (0) for root)

stage { all | <offset> <length> } <level> [async]
   all             Stage all data from specified level
   <offset>        Offset position of data to stage
   <length>        Number of bytes to stage
   <level>         Storage level to stage data to
   async           Stage data asynchronously

stat <path>
   <path>          Name of file/directory to get stats on of

truncate <path> <new-length>
   <path>          Name of file to truncate
   <new-length>    New file length (in bytes)

umask <mode>
   <umask>         New umask value (in octal with leading zero)

unlink <path>
   <path>          Pathname to delete

write <total-bytes> [ <buffersize> ]
                        [ format=<format> ]
    write <buffersize> <iterations>x [ format=<format> ]

   <total-bytes>     Specifies total number of bytes to write
   <buffersize>      Number of bytes to write each time
   <format>          Specifies format of data to be written:
                       0  - write binary zeros
                       1  - write binary ones
                       n# - write values equal to “n” times offset
                       random - write random data to minimize compression
   <iterations>      Number of times to write <buffersize>
```

EXAMPLE

In the following scrub commands, an HPSS file is created in class-of-service 3. 100 megabytes are written into the file in 8mb chunks using a data pattern that is equal to the byte offset. The last 20mb are then overwritten using a byte pattern of all binary ones followed by writing 50mb of binary zeros. The file is then read back for content verification.

```
% scrub
scrub> hints cos-id 3
```

```
     scrub> open testfile crw
     scrub> write 100mb 8mb
     scrub> lseek -20mb
     scrub> write 20mb format=1
     scrub> write 50mb format=0
     scrub> close
     scrub> open testfile r
     scrub> read 80mb
     scrub> read 20mb 8mb format=1
     scrub> read 50mb 4mb format=0
     scrub> close
     scrub> quit

FILES

     /usr/lpp/hpss/bin/scrub       Scrub program with only TCP support
     /usr/lpp/hpss/bin/scrub_ipi   Scrub program with both TCP and IPI support

SEE ALSO


SCCS
     man/scrub.7, gen, 4va   11/23/98   12:37:57
```

I.52 setdmattr — Set DM attributes for a file that is stored in DFS

```
NAME
     setdmattr - Set DM attributes for a file that is stored in DFS.

SYNTAX
     setdmattr <Path Name> <DM attribute> <Attribute Value>

DESCRIPTION
     setdmattr is a utility that can be run on the DFS server machine
     that contains the Episode file system that contains the file. This
     utility will generally be used to clean up state that has become
     inconsistent between a DFS and HPSS archived fileset.

     This tool must be run as root.

     The output from this utility will describe the state of the file and
     will include all DM attributes associated with this file, the region
     information, and the file data residency. This output will show
     whether the DM attributes were set correctly. Most of the output
     will only be important for debugging purposes.

     See “HPSS Administration Guide: DFS Configuration and Management”
     for more details.

PARAMETERS

     Path Name       - The path to the file through the local UNIX mount
                       point.

     DM attribute    - Name of the attribute to set.

     Attribute Value - The value of the attribute to set.

EXAMPLE
     The following command will set the DM attribute “MIGRATE” to a value
     of “1”.

     setdmattr /var/hpss/hdm/hdm1/aggr/test.aggr/test.fileset/file.0 \
                                                                MIGRATE 1

FILES


SEE ALSO
     getdmattr(7) archivecmp(7) archivedump(7) archivelist(7)

SCCS
     man/setdmattr.7, gen, 4va 11/20/98 15:24:26
```

I.53 settapestats — Compute Tape Storage Server Statistics

```
NAME
     settapestats - Compute tape storage server statistics and update
                    metadata.

SYNTAX
     settapestats [ -p Utility Principal Name
                    -k Utility KeyTab Pathname
                    -g HPSS Generic Config Name
                    -d SS Descriptive Name
                    -s Storage Class(optional)
                    -u Update metadata (optional)]

DESCRIPTION
     This utility reads the given tape storage server's metadata
     and computes the global and per storage class statistics.  A
     statistics report is written to standard out.  Optionally, the
     global and/or storage class statistics fields in the server
     metadata files can be updated with the new values, and the
     storage class statistics flags in the tape storage maps can
     be updated.  A specific storage class can be selected for
     display and optional update.


OPTIONS

     -d <SS Descriptive Name>
                          The descriptive name of the HPSS Tape Storage
                          Server which manages the Volumes to be scanned.
                          The descriptive name of a storage server
                          can be found from SSM through: Admin, Configure
                          HPSS, Server Configuration screens.

     -g <HPSS Generic Config Name>
                          The pathname of the SFS generic server
                          configuration file.

     -p <Utility Principal Name>
                          The DCE principal name which this utility will run
                          under.  Usually run as SSM.  The Principal name of
                          the SSM can be found from SSM through: Admin,
                          Configure HPSS, Server Configuration, Security
                          Information screens.

     -s <Storage Class>   An optional argument.  If the storage class is
                          is specified, the output will be restricted to
                          volumes in that class.  If an update_tallys or
                          update_both is selected by the -u option, the
                          statistics for only the selected storage class
                          will be updated.  The default is to select
```

all storage classes.

-k <Utility KeyTab Pathname>
The path to the keytab file.

-u <Update metadata>
If omitted, the statistics are calculated, but the metadata files are not modified. If provided, the allowable values are:

update_global

The global statistics metadata records are updated.

update_tallys

The per storage class tally metadata records are updated and the storage map statistics flags (MAP_STAT_FREE and MAP_STAT_PARTIAL) are updated.

update_both

Both the global and per storage class metadata records are updated.

DEFAULTS

settapestats gets default values for the following from two sources. First, the execution environment of the program is checked for the environment variables listed below. For any that are defined, the values become the defaults. Second, for each environment variable not found in the first step, a value is assigned from the HPSS system default values.

SS Descriptive Name
If not supplied on the command line, defaults to the value of the HPSS environment variable HPSS_DESC_SSTAPE.

HPSS Generic Config Name
If not supplied on the command line, defaults to the value of the HPSS environment variable HPSS_CONFIG_SERVER.

Utility Principal Name
If not supplied on the command line, defaults to the value of the HPSS environment variable HPSS_PRINCIPAL_SSM.

Utility KeyTab Pathname

If not supplied on the command line, defaults to the value of the HPSS environment variable HPSS_KEYTAB_FILE_SERVER.

NOTES

Only the tape storage server storage maps are scanned by this utility. Statistics generated are computed solely from the information available in the storage maps using the same algorithm as the tape storage server. Disk statistics cannot be computed or modified with this program.

The global statistics are the “Total Bytes”, “Free Bytes” and “Used Bytes” fields found in the server’s specific configuration record and returned by the ss_SSrvGetAttrs() server function.

The storage class statistics are the “Total VVs” and “Free VVs” fields returned by the ss_GetStorageClassStats() server function and displayed in the storage class statistics SSM window.

The storage map statistics flags are MAP_STAT_FREE and MAP_STAT_PARTIAL which are used by the server to keep track of how each tape volume has contributed to the storage class statistics tally. These are new flags in HPSS version 4.1 and need to be set by settapestats before correct storage class statistics can be returned by the tape storage server.

If a specific storage class is selected with the -s option, statistics for that class only are reported. Global statistics are not calculated and cannot be updated.

It is preferable to update the storage server statistics when the server is not running, but the statistics can be updated while the server is running if necessary. settapestats can be run as often as desired without harming the tape storage server metadata.

This utility does not operate on disk storage server metadata. Any attempt to do so results in an error.

EXAMPLE

The following command lists the tape storage server statistics for the server named “Tape Storage Server”. Both the global and storage class statistics fields in the server metadata are updated.

The server descriptive name, generic config file name, utility principal name and keytab path are provided in the environment.

```
$ settapestats -u update_both

Statistics for ‘Tape Storage Server’

Found 1091 tape storage maps:

  Free:         152
```

```
  Allocated:      1
  EOM:          929
  Empty:          6
  Tally:          3
  Retired:        0

Summary for each storage class:

Storage    Total     Free  Partial      EOM    Empty  Retired    Storage
  Class      VVs      VVs      VVs      VVs      VVs      VVs   Segments

      2     1033      101        1      929        2        0    1451682
      4       50       45        1        0        4        0          3
      5        5        2        3        0        0        0          0


Storage
  Class          Total Bytes           Free Bytes           Used Bytes

      2   11,807,692,714,282    1,088,834,128,709    9,019,007,695,986
      4      581,870,354,198      483,183,820,801           10,304,330
      5       53,687,091,200       52,620,233,999                    0

Totals    12,443,250,159,680    1,624,638,183,509    9,019,018,000,316

Updating storage class 2 tally map.
Update for storage class 2 successful.

Updating storage class 4 tally map.
Update for storage class 4 successful.

Updating storage class 5 tally map.
Update for storage class 5 successful.

Note:  The storage class totals shown above may not match the totals
       shown below.  The Total, Free and Used bytes shown below are
       computed using a different formula from the storage class totals.

Global totals:
  Total bytes =    10,655,450,738,015
  Free bytes  =     1,642,824,990,720
  Used bytes  =     9,012,625,747,295


Updating the global statistics with the above.
Update successful.


Updating the storage map statistics flags ...

Set MAP_STAT_FREE in 148 maps.
Set MAP_STAT_PARTIAL in 5 maps.
Cleared MAP_STAT_xxxx in 0 maps.
```

```
     Storage map statistics flags update successful


SCCS
     man/settapestats.7, gen, 4va   11/23/98   13:58:56
```

I.54 sfsbackup — Generate SFS Data Volume Backups

```
NAME
     sfsbackup - Generate SFS data volume backups

SYNTAX
     sfsbackup <sfsServerName> <backupDir> {<volName> | all} \
                   <fileSize> <numBackupFiles>

DESCRIPTION
     This tool generates a specified number of incremental backup files for
     SFS data volume <volName> managed by the SFS server <sfsServerName>.  If
     the value of <volName> is “all”, backup files are generated for all
     Encina SFS data volumes.  <numBackupFiles> specifies the number of backup
     files that will be created.

     The generated TRB backup files are placed in the directory
     <backupDir>/<volName> where each backup file will contain <fileSize>
     bytes.  Note that <fileSize> should be specified in kilobytes with a
     trailing “k” and that the size should include 1 kilobyte for overhead.
     For example, to generate backup files in 16mb chunks, the <fileSize>
     argument would be 16mb+1kb=16385k.

     If multiple SFS servers reside on the same node, unique backup directories
     should be specified in case multiple SFS servers use common volume names.

     The user should be aware of how much disk space will be required for the
     newly generated backup files.

     This utility uses the “tkadmin backup lvol” command to generate all SFS
     backup files.

     The user must be logged into DCE as the administrator for the SFS server
     prior to invoking this utility.

     Once generated, SFS backup files should be copied/moved to offline
     storage.

     Before invoking the sfsbackup utility, ensure that SFS media archiving is
     enabled.  Use the “tkadmin query mediaarchiving -server <sfsServerName>”
     command to verify.  If it is disabled, use the “tkadmin enable
     mediaarchiving -server <sfsServerName>” command to enable media
     archiving.

EXAMPLE
     To generate 32 incremental backup files for volume sfsVol1 on SFS server
     /.:/encina/sfs/hpss in 16mb chunks, issue:

        % sfsbackup /.:/encina/sfs/hpss /archive/sfsbackups sfsVol1 16385k 32

     This will place the newly created backup files in the directory
```

```
     /archive/sfsbackups/sfsVol1.

     To generate 10 backup files for all volumes managed by SFS server
     /.:/encina/sfs/hpss in 32mb chunks, issue the following:

        % sfsbackup /.:/encina/sfs/hpss /archive/sfsbackups all 32769k 10

FILES


SEE ALSO

     “tkadmin backup lvol” command
     “tkadmin query lvol” command

SCCS
     man/sfsbackup.7, gen, 4va   10/11/97   17:13:25
```

I.55 shelf_tape — HPSS Shelf Tape Utility

```
NAME
     shelf_tape - HPSS Shelf Tape Utility

SYNTAX
     shelf_tape [-g HPSS Generic Server Path -c Utility Server(CDS) Name
       -p Utility Principal Name -k KeyTab Path -q Storage Class Id
       -f Input File Name -a -h]
       -r PVR Descriptive Name
       -o <age | access | count>
       -d Minimum Days Since Last Mount
       -n Number Of Tapes To Be Shelved


DESCRIPTION
     “shelf_tape” is a HPSS System Maintenance Utility. It is used by a HPSS
     system administrator to select and move tapes from a tape library to
     an off-line storage area.

     An administrator can either provide the selection criteria and let the
     shelf_tape utility select the candidate tapes, or the administrator can
     specify the tapes to be shelved. In either case, a tape has to be
     DISMOUNTED and in EOM or RETIRED state before it can be selected or
     shelved from a tape library.

     If the selection is done by the shelf_tape utility, -r, -o, -d and -n
     parameters are required. If the selection shall be confined in a
     single Storage Class, then -q is required.

     If the candidate tapes are provided by the administrator, -f option
     with the input file name is required.

     An advisory mode option, -a, is provided to list the selected candidate
     tapes without removing the tapes from the library.  The advisory mode can
     be used to verify which tapes can be shelved when the tapes are
     provided in the -f option.

     An argument list is provided by entering the -h option.


OPTIONS

     -g <Generic Name>     The pathname of the SFS generic server
                           configuration file.
                           The default value is picked up from the
                           environment variable HPSS_CONFIG_SERVER.

     -c <CDS Name>         The Cell Directory Service name which this
                           utility will run under. Usually use SSM’s CDS name.
                           The Server CDS name can be found from SSM
```

```
                        through: Admin, Configure HPSS, Server
                        Configuration screens.
                        The default value is picked up from the
                        environment variable HPSS_CDS_SSMSM.

-p <Principal Name>     The server principal name which this utility
                        will run under.
                        The Principal name of the SSM can be found from
                        SSM through: Admin, Configure HPSS, Server
                        Configuration, Security Information screens.
                        The default value is picked up from the
                        environment variable HPSS_PRINCIPAL_SSM.

-k <Keytab Name>        The Keytab name which this utility will run with.
                        The Keytab name can be found from SSM through:
                        Admin, Configure HPSS, Server Configuration,
                        Security Information screens.
                        The default value is picked up from the
                        environment variable HPSS_KEYTAB_FILE_SERVER.

-r <PVR Name>           The descriptive name of the HPSS PVR which
                        manages the tapes to be shelved. The descriptive
                        name the PVR can be found from SSM through:
                        Admin, Configure HPSS, Server Configuration
                        screens.

-o <age | access | count>
                        The selection is either based on the age of the
                        data, last access time, or number of accesses
                        to the tape.

-d <Minimum Access Days>
                        This option precludes the tape that has been
                        accessed in the last specified days. If not
                        specified, the default value is one day.

-q <Storage Class>      The tape should be selected from this Storage
                        Class only.
                        The storage classes supported can be found
                        from SSM through: Monitor, Storage Class screens.
                        If not specified, the selection is from all the
                        Storage Classes.

-n <Number>             The number of tapes to be shelved. If not
                        specified, the default value is 10.

-f <Tape List>          The pathname to a file which contains the labels
                        of the tapes to be shelved.

-a                      Display the selected tapes on the screen without
                        removing the tapes from the tape library.

-h                      Display the help screen.
```

EXAMPLE

All the examples except the last one assume the CDS Name, HPSS Generic Server Configuration Path, Key Tab Path, and Utility Principal Name have been properly set. These environment variables are usually configured in the hpss/config/hpss_env script file. By sourcing this file, the required environment variables shall be properly set.

Shelve the 10 tapes which hold the files that were created the earliest and the tapes that have not been accessed in the past 5 days.

```
shelf_tape -r “3494 PVR” -o age -d 5 -n 10
```

Identify the 10 tapes which hold the files that were created the earliest and the tapes that have not been accessed in the past 5 days.

```
shelf_tape -r “3494 PVR” -o age -d 5 -n 10 -a
```

Shelve the 10 tapes which hold the files that were created the earliest and the tapes that have not been accessed in the past 5 days. These tapes should be selected from Storage Class 2 only.

```
shelf_tape -r “3494 PVR” -o age -d 5 -n 10 -q 2
```

Shelve the 10 tapes which have not been accessed for the longest time.

```
shelf_tape -r “3494 PVR” -o access -d 0 -n 10
```

Shelve the 10 tapes which have not been accessed for the longest time, but exclude the tapes which have been accessed in the last 2 days. (Note: If all the tapes have been accessed in the last 2 days, no tapes will be shelved.)

```
shelf_tape -r “3494 PVR” -o access -d 2 -n 10
```

Shelve the tapes specified in the /dir1/file1 file.

```
shelf_tape -r “3494 PVR” -f “/dir1/file1”
```

(Note: the contents of /dir1/file1 are as following

```
cat /dir1/file1

A00001
A00002
A00003
A00004
```

)

Shelve the 10 tapes which hold the files that were created the earliest

```
     and the tapes that have not been accessed in the past 5 days.
     (Note: This example assumes none of the required environment variables
            have been set.)

         shelf_tape -g /.:/encina/sfs/hydra/serverconfig -c /.:/hpss/ssmsm
         -p hpss_ssm -k /krb5/hpss.keytabs -r “3494 PVR” -o age -d 5 -n 10


FILES


SEE ALSO


SCCS
     @(#)38   1.1   man/shelf_tape.7, gen, 4.1.1, 4.1.1.1   4/2/99   10:00:53
```

Appendix J

IBM 3494/3495 PVR Information

J.1 Vendor Software Requirements

HPSS is designed to work with IBM 3494/3495 robots attached to an HPSS server with either RS-232, Ethernet, or Block Multiplexer (BMUX) control connections. Data paths to the drives will be SCSI-2 with RS-232 and Ethernet control paths, or BMUX if BMUX control paths are used. The HPSS PVR must run on a machine with the appropriate version of Library Manager Control Point (LMCP) device drivers installed.

J.2 Configuration Requirements

When configuring an HPSS PVR to manage an IBM robot, the Drive Address configuration entries correspond to the device number of the drive. Determine the device number by running the HPSS tool **GetESANumbers** for each tape drive in the robot.

The HPSS PVR requires one or two LMCP device special files to access the robot. For AIX systems, these files are usually named **/dev/lmcpX** and can be created using the System Management Interface Tool (SMIT). For a BMUX connected robot, the files can be defined to be a *Command Port* or an *Asynchronous (Async) Port*. The HPSS PVR requires both. By default, the PVR will expect the Command and Async ports to be named **/dev/lmcp0** and **/dev/lmcp1** respectively. Control connections must be made prior to configuration of the **/dev/lmcpx** devices or undefined errors may result. For Solaris systems, the symbolic library name defined in **/etc/ibmatl.conf** (e.g., **3494a**) should be used.

For RS-232 and Ethernet connected robots, the device special files support both command and async capabilities. If only one device special file is created, the environment variables or configuration should be set so that both the command and async ports point to that one special file. It is also possible to create two device special files and arbitrarily select one to be the command port and the other to be the async port. The PVR can then be configured to recognize the ports as described above in the BMUX case.

HPSS can share an IBM robot with other tape management systems. If a robot is shared, care must be taken to make sure that a drive is not used by any other tape management system while that drive is configured as unlocked in the HPSS PVL. This is important because HPSS periodically polls all of its unlocked drives even if they are not currently mounted. The two LMCP device special files (or possibly one file in the case of an RS-232 or Ethernet controlled robot) are not available to other

tape management programs. Additional device special files may be created for other programs. Other programs can send commands to the robot at the same time as HPSS through the additional device special file.

If the robot is placed in *pause mode* by an operator, an alarm will appear on the HPSS operator screen. All subsequent robot operations will silently be suspended until the robot is put back in automatic mode.

J.3 Cartridge Import and Export

When importing new cartridges into HPSS, the cartridges must be entered into the IBM robot before any HPSS import operations are performed. Cartridges placed in the convenience I/O port will automatically be imported by the robot.

When ejecting cartridges, HPSS will send cartridges to the convenience I/O port if it is available. If the port is not available and a high capacity I/O region is defined, the cartridges will be placed in that region. If no locations are available for the cartridge to be ejected, an alarm will appear on the HPSS operator screen and HPSS will retry the eject operation periodically.

J.4 Vendor Information

1. *IBM 3494 Tape Library Dataserver Operator's Guide*, GA32-0280-02
2. *Parallel and ESCON Channel Tape Attachment/6000 Installation and User's Guide*, GA32-0311-02
3. *IBM SCSI Device Drivers: Installation and User's Guide*, GC35-0154-01
4. *IBM 3495 Operator's Guide*, GA32-0235-02
5. *Parallel and ESCON Channel Tape Attachment/6000 Installation and User's Guide*, GA32-0311-02

Appendix K

StorageTek PVR Information

K.1 Vendor Software Requirements

HPSS is designed to work with Storage Technology Corporation's (STK's) Automated Cartridge System Library Software (ACSLS) Version 4.0 or 5.0.

ACSLS should be running on the workstation directly connected to the STK Silo. The HPSS STK PVR can run on any workstation that has a TCP/IP connection to the ACSLS workstation. The workstation running the HPSS STK PVR must also be running STK's Storage Serve Interface (SSI) software. This software will not be started by HPSS and should be running when HPSS is started. It is recommended that the SSI be started by the workstation's initialization scripts every time the workstation is booted.

The SSI requires that the system environment variables **CSI_HOSTNAME** and **ACSAPI_PACKET_VERSION** be correctly set. See the *STK Automated Cartridge System Library Software (ACSLS) System Administrator's Guide* for more information. Note that due to limitations in the STK Developer's Toolkit, if the SSI is not running when the HPSS STK PVR is started, or if the SSI crashes while the HPSS STK PVR is running, the HPSS STK PVR will lock up and will have to be manually terminated by issuing a **kill -9** command.

K.2 Configuration Requirements

When configuring HPSS to manage an STK Silo, the Drive Address configuration entries correspond to the ACS, Unit, Panel, Drive number used by ACSLS to identify drives. For example, the first drive in a typical Silo configuration has the Drive Address (**0,0,10,0**).

HPSS can share an STK robot with other tape management systems. If a robot is shared, care must be taken to make sure that a drive is not used by any other tape management system while that drive is configured as unlocked in HPSS. This is important because HPSS can be configured to periodically poll all of its unlocked drives even if they are not currently mounted or in use by HPSS. If a drive is being used by another tape management system, it must be configured as locked in HPSS.

HPSS periodically polls all drives in its pool, so these drives cannot be used by other tape management software when HPSS is running even if HPSS is not currently mounting tapes to those drives.

HPSS will use any Cartridge Access Port (CAP) in the STK Silo that has a priority greater than zero. When a CAP is needed by HPSS, HPSS will pick the highest priority CAP that is currently available. At least one CAP must be assigned a non-zero priority. See the *STK Automated Cartridge System Library Software (ACSLS) System Administrator's Guide* for procedures to set CAP priority.

K.3 Cartridge Import and Export

When importing new cartridges into HPSS, the cartridges must be entered into the STK Silo before any HPSS import operations are performed. See the *STK Automated Cartridge System Library Software (ACSLS) System Administrator's Guide* for procedures to enter cartridges. When exporting HPSS cartridges, the cartridges will be placed in the CAP for removal.

Cleaning cartridges are managed by the ACSLS software and should not be imported into the HPSS system.

STK supports cartridges without bar code external labels. These cartridges can be entered into the Silo using the **venter** command at the ACSLS console. These cartridges are fully supported by HPSS, but the Silo will eject the cartridges if they are scanned during an audit.

K.4 Starting the STK Client Processes

In order for the HPSS STK PVR to communicate with the STK robot, it is required that STK client side processes be running on the node where the STK PVR is executing. These client side processes are the Server System Interface (ssi) and the Toolkit event logger. These binaries and associated script files are distributed with the HPSS, but are maintained by the STK Corporation.

The binaries and script files for starting the STK client side processes are located in the **$HPSS_PATH/stk/bin** directory. Documentation files describing the files in the bin directory are located in the **$HPSS_PATH/stk/doc** directory. Refer to these doc files for additional information.

The **t_startit.sh** file must be executed to start the STK client processes on those nodes where the HPSS PVR is executing. The script will prompt for the following options:

1. Multi-Host or Single-Host - specify m for multiple host.
2. Server side or Client side - Specify c for client side.
3. Remote Host Name - specify the name of the host where the ACSLS software is running.
4. Remote Host Version - specify one of the following (probably 4).

ACSLS Release Number	Remote Host Version Number
3.x	2
4.x	3
5.x	4

5. Whether processes created should be listed (Y or N) - specify either y or n.

6. Whether **t_cdriver** should be started - specify n.

To terminate the STK client processes, the script **t_killit.sh** may be executed.

Sample output from t_startit.sh follows:

***** Welcome to t_startit.sh, Version 2.01 *****

This is the automated tester and startit script for the TOOLKIT. Simply answer the questions which follow.

Executables live in: /usr/lpp/hpss/stk/bin

Would you like Multi-Host or Single-Host testing?

Enter one of the following followed by ENTER:

M Multi-host testing

S Single-host testing

X eXit this script

Enter choice: m

Would you like to define the server side or client side for Multi-Host testing?

Enter one of the following followed by ENTER:

S Server side

C Client side

Enter choice: c

The Remote Host Name is the name of the server which has the ACSLS software (or simulator) running on it.

Enter Remote Host Name (CSI_HOSTNAME): jeep

The Remote Host Version number is the ACSLS Packet Version level which the server is expecting.

Here are the valid choices:

ACSLS Release Number Remote	Host Version Number
3.x	2
4.x	3

5.x 4

Enter Remote Host Version (ACSAPI_PACKET_VERSION): 4

Starting /usr/lpp/hpss/stk/bin/mini_el... Attempting startup of /usr/lpp/hpss/bin/mini_el ...

Starting /usr/lpp/hpss/bin/ssi... Attempting startup of PARENT for /usr/lpp/hpss/bin/ssi...
SIGHUP received Parent Process ID is: 17290 Attempting startup of /usr/lpp/hpss/bin/ssi...
SIGHUP received Parent Process #17290 EXITING NORMALLY

Initialization Done.

Do you want to see the processes created? (Y or N): y

17288 p4 S 0:00 /usr/lpp/hpss/stk/bin/mini_el

17292 p4 S 0:00 /usr/lpp/hpss/stk/bin/ssi 17290 50004 23

17295 p4 S 0:00 grep /usr/lpp/hpss/stk/bin

Do you want to start up t_cdriver? (Y or N): n

K.5 Vendor Information

1. *STK Automated Cartridge System Library Software (ACSLS) System Administrator's Guide*, PN 16716

2. *STK Automated Cartridge System Library Software Programmer's Guide*, PN 16718

Appendix L

ADIC Automatic Media Library Storage Systems Information

L.1 Vendor Software Requirements

HPSS is designed to work with ADIC Distributed Automated Media Library Server (DAS) software version 1.3 and the ABBA Management Unit (AMU) version 2.4.0. DAS is the ADIC software which consists of the Automated Media Library (AML) Client Interface (ACI) and the DAS server components. The AMU is the host computer software by which the ADIC Storage System manages the archive database, which is based on a DB/2 compatible database for an OS/2 system.

The AMU must run on a OS/2 PC host computer connected to the AML robot while the HPSS AML PVR can run on any RS/6000 workstation that has a TCP/IP connection to the OS/2 host computer. The workstation running the HPSS AML PVR must also contain the DAS/ACI software that is called by the HPSS AML PVR.

Refer to ADIC *DAS Installation and Administration Guide* and *Reference Guide AMU* for additional information.

L.2 Configuration Requirements

HPSS can share an AML robot with other tape management systems. If a robot is shared, care must be taken to make sure that a drive is not used by any other tape management system while that drive is configured as unlocked in HPSS. This is important because HPSS can be configured to periodically poll all of its unlocked drives even if they are not currently mounted or in use by HPSS. If a drive is being used by another tape management system, it must be configured as locked in HPSS. For robots that have more than one arm (such as the AML/2), users should configure the PVL Drive Information/Controller ID of each drive depending which arm is servicing it.

User needs to set the Server Name and Client Name, which are case sensitive, in the AML PVR Server Configuration panel to establish the connectivity between the HPSS software and the OS/2 controlling the robot. The Server Name is the name of the controller associated with the TCP/IP address, as defined in the TCP/IP HOST file, and the Client Name is the name of the OS/2 administrator client as defined in the DAS configuration.

Additionally users must configure the Insert/Eject ports for the AML PVR using the configuration files **/var/hpss/etc/AML_EjectPort.conf** and **/var/hpss/etc/AML_InsertPort.conf**. The reasons are that the AML robot can have multiple insert and eject ports, which have the capability to handle different media types. These two configuration files in conjunction with the AMU AMS configuration files specify which Insert/Eject areas or bins the tapes should be placed in for insertion into the archive and where they are ejected to when the HPSS export command is used.

L.3 Cartridge Import and Export

When importing new cartridges into HPSS, the cartridges must be entered into the AML Insert/Eject/Foreign (I/E/F) port before any HPSS import operations are performed. The HPSS AML PVR moves the cartridges into the towers or shelves before importing into HPSS. Users can then issue the HPSS cartridge import command to import the tapes into HPSS; users do not need to issue the **dasadmin insert** command prior to importing. See the ADIC *AML Administrator's Guide* for procedures to enter cartridges into the I/E/F bins. When an HPSS tape export command is issued, the cartridges will be placed in the I/E/F ports and must be removed before issuing an HPSS import command.

Cleaning cartridges are managed by the DAS software and should not be imported into the HPSS system.

L.4 Starting the DAS Server Processes

In order for the HPSS AML PVR to communicate with the AML robot, it is required that the DAS Server process be running.

Once the DAS, TCP/IP and AML configuration have been completed, the DAS server is initialized and started with the following steps:

1. Make sure the AMU archive management software is running and the hostname is resolved,

2. Select an OS/2 window from the Desktop and change the directory to C:\DAS,

   # C:> cd \das

3. At the prompt, type **tcpstart** and make sure that TCP/IP gets configured and that the port mapper program is started,

   # C:\das>tcpstart

4. Type **dasstart** to start the DAS server

   # C:\das> dasstart

In most cases, a startup file can be used to automatically start the DAS processor:

```
call tcpstart
das\tools\os2sleep 20
```

```
CD \AMU
START CON
START KRN
cd \das
tools\os2sleep 20
call dasstart
cd bin
start "DAS/2 AmuClient"
exit
```

L.5 Vendor Information

1. *Administration Guide Distribution AML Server, Order no. DOC F00 010-B*
2. *Reference Guide AMU, Order no. DOC E00 005*
3. *Interfacing Guide DAS, Order no. DOC F00 011*

Appendix M

HPSS Release 4.1.1

This appendix describes the new Release 4.1.1 features and components and discusses the considerations and recommendations for using them. The appendix also identifies Release 3.2 features which are no longer be supported. In addition, this appendix describes the procedures to allow sites running with HPSS 3.2 and 4.1 to upgrade to Release 4.1.1.

If you are upgrading from a previous version of HPSS, be sure to consult the documentation supplied with your upgrade. It will be located in the /hpss/prod directory in a file ending in ".README". As this file contains important information that is not otherwise provided, it is imperative that all sites review it before upgrading.

M.1 4.1 Features and Components

This section is provided for those sites currently running HPSS Release 3.2. Sites already running 4.1 will find no new information in this section.

1. DFS

   Release 4.1 provides distributed file system service by optionally allowing HPSS to interface with Open Group's DFS. DFS is a highly scalable distributed file system that provides a uniform view of file data to all users through a global name space. HPSS / DFS implementation includes:

   - Support for standard XDSM implementation to migrate DFS data into HPSS.
   - Support for interfaces to keep DFS and HPSS data and name space synchronized.
   - Support for migration and staging of data between Episode and HPSS.
   - Support for purging of data from Episode after it has been migrated to HPSS.
   - Support for two types of HPSS / DFS filesets:
     - Archived
       - Data is migrated from Episode into HPSS.
       - DFS and HPSS name spaces are not synchronized.

- Files are accessed through DFS only.
- Files are created in HPSS at migrate time.
- File deletes are queued for later HPSS deletion.
- Data transfer rate should be as fast as DFS rate except when data is staged from HPSS.

- Mirrored Filesets
  - Data is migrated from Episode into HPSS.
  - DFS and HPSS name spaces are in synchronized.
  - Data can be accessed and manipulated from either DFS or HPSS.
  - HPSS high transfer rate can be utilized.
  - Name space overhead increases to keep name spaces synchronized.

2. Filesets and junctions

   A fileset is a directory, administered as a unit, that can be mounted in the global name space. A junction is analogous to a symbolic link, allowing a fileset to be accessed.

   - Name Server supports HPSS-only and DFS filesets.
   - Name Server supports junctions to link / unlink filesets in / from HPSS name space.
   - Utility provided to create filesets for existing Release 3.2 systems.

3. File Families

   - Supports segregation of files by family onto tape virtual volumes assigned to the family.
   - Provides an administrative interfaces to create and delete file families.
   - A file family may be associated with a fileset when the fileset is created.
   - If a family ID is not associated with a fileset, files created in the fileset are assigned to family zero (0), the system default family.
   - The Tape Storage Server creates storage segments within a family on tapes assigned to that family. If no tape is available in the family, the server assigns a blank tape to the family, then creates segments on it.
   - All tape segment moves, repacks and COS changes take place within the family.
   - The family ID is stored in the file's migration records as part of the secondary key to minimize tape mount activity.
   - Reclaim resets the family ID of the volumes being reclaimed to zero.

4. DMAP Gateway Server

   - Translates calls between HDM and HPSS.
   - Migrates data from DFS into HPSS.
   - Invalidates data in DFS and HPSS.
   - Provides the following interfaces via SSM:
     - Server get / set attributes.
     - Create / delete filesets.
     - Get / set fileset attributes.

5. Location Server

   - Acts as an information clearinghouse enabling clients to locate servers and gather information from the local HPSS system.
   - Performs two main functions:
     - Allows a client to determine a server's location, its CDS pathname, and its object UUID. This allows a client to contact the appropriate Bitfile Server or Name Server.
     - Aids the client in selecting the appropriate Class of Service to use when creating bitfiles.

6. MPI-IO API

   The MPI-IO API is a subset of the MPI-2 standard. It provides applications written for a distributed memory programming model an interface that coordinates access to HPSS files from multiple processes. These processes can read and write data from a single file in parallel using HPSS's third-party transfer facilities. The interface also lets applications specify discontiguous patterns of access to files and memory buffers using the same "datatype" constructs that the Message-Passing Interface (MPI) offers. For the specific details of this interface, see the *HPSS Programmer's Reference Guide,* Volume 1, Release 4.1.

   MPI-IO provides the following functionality:

   - File open / close / delete / resize / preallocate.
   - Specification of file access patterns through MPI datatypes.
   - Collective and independent read / write.
   - Synchronous (blocking) and asynchronous (nonblocking) read / write.
   - Explicit and implicit (using individual or shared file pointers) file positioning for read / write.
   - Support for nonnative data representations for file interoperability.

- User “hints” specified at file-open to help optimize file access.
- Support for file consistency / synchronization across multiple processes.

7. Scalable Accounting
   - Improve accounting performance.
   - BFS creates accounting log records for create, open, read, write, clear, close, and unlink operations.
   - Background thread reads log records to create accounting summary records.
   - Accounting summary records support the following collection options during the accounting period:
     - By accounting index and COS, number of bitfiles stored
     - By accounting index and COS, number of accesses
     - By accounting index and COS, number of bytes transferred
     - By accounting index and COS, bytes used
     - By accounting index, COS and storage class, number of bytes transferred
   - Utilities are provided to create the initial set of account summary records.
8. FTP enhancements
   - Provides restricted Port specification and improved logging.
   - Support for network load-balancing through the use of multiple network interfaces and / or multiple nodes / independent systems for each child process for multi-network stripe transfers.
   - The “**quote site symlink filea fileb**” command has been added to allow creation of symbolic links.
   - Enhanced credential-based authentication including “Cross Cell” Authentication.
   - Improved COS determination.
9. Mover device initial disk offset

   Allows an initial starting offset to be configured for each disk device, such that the HPSS Mover will not access that device between offset zero and the configured starting offset. This allows for an initial non-HPSS volume label to be untouched (e.g., for AIX LVM labels, required when performing control operations on mirrored logical volumes).
10. Support for disk partitions greater than 2GB

    Allows for the support of Mover disk devices that are greater than 2GB in size. Note that AIX version 4.2.1 (or later) is a prerequisite for this capability.

11. Support pre-allocation of storage segments

    - BFS support option to pre-allocate storage segments when resetting the length of a file beyond its current length.
    - Client API provides resize interface to set the length of the file.

12. Support new COS hints

    - Add stripe length and width to the COS hints.
    - COS selection algorithm modified to use the new fields.

13. Faster disk file creates

    - Disk storage segments are created in batches for faster disk file creates.

14. Faster disk file purges

    - Disk storage segments are deleted in batches to keep the BFS disk segment unlink queue at a minimum.

15. Support for larger disk Virtual Volumes

    - The maximum number of disk Virtual Volume blocks has been raised to 16,384 from 8,192. This allows up to 16,384 disk storage segments to be created per volume.

M.2 New 4.1.1 Features and Components

1. Non-DCE Client Gateway

    - Allows network clients running in environments lacking DCE and/or Encina to make calls to the HPSS Client API using the Non-DCE Client API.

2. Non-DCE Client API

    - Provides standard HPSS Client API calls to client applications that run in environments without DCE and/or Encina.
    - Utilizes a Non-DCE Client Gateway to pass its API calls on to an HPSS system.

3. Shelf Tape support

    - **shelf_tape** utility provided that facilitates the removal of tapes from libraries while maintaining their HPSS metadata.
    - SSM will inform the operator when a shelved tape needs to be re-inserted into the library.

4. AML/2 robot support

    - New AML PVR is provided for interfacing with ADIC DAS/ACI libraries.

5. Movers now ported to IRIX 6.4

6. Support for new media types

   - Ampex DST-314 drives (see Section 2.4.4.1)
   - IBM 3590E tape drives (see Section 2.4.4.1)
   - MaxStrat disk from SGI

7. Client API and Utility Enhancements

   - hpss_Migrate is enhanced to handle multiple copies.
   - bfs_Purge, hpss_FileGetXAttributes, and bfs_BitfileGetXAttrs APIs are enhanced to support the locking of files into the top level (disk) of a hierarchy. Files so locked are not purgeable and are said to be purge locked.
   - hpss_FileGetXAttributes and bfs_BitfileGetXAttrs are enhanced to allow users to determine what levels of a hierarchy a file resides in.
   - hpss_PVrr is a new API that queries the number of retrieval requests on a storage media basis.
   - hpss_GetBFSStats and hpss_SetBFSStats are enhanced to allow users to query and reset the number of migrate, purge, stage, and delete requests.
   - **lsrb** is a new utility provided to output a list of iles based on date of last access.
   - **plu** is a new utility provided to output a sorted list of pathnames of files which have purge records.

8. Purge policies now support purging based on either purge record creation time, bitfile creation time, or time of last bitfile access.

9. Purge locked files are logged at purge time

   - **plu** has the ability to list files which have been purge locked longer than a configurable amount of time.
   - The Purge Policy window supports a new field for the maximum number of minutes a file can be purge locked before messages are logged.

10. **repack** is enhanced to optionally repack volumes that have been checked out (shelved tapes).

11. The following have been ported to Solaris 2.6

    - Mover and Client API supported for both non-DCE and DCE environments.
    - Startup Daemon and Log Client

12. New utility for working with shelved tapes, **shelf_tape_util**.

M.2.1 New in 4.1.1.2 Patch

13. The following have been ported to Solaris 2.6
    - PVRs: STK, 3494, and Operator
    - MPI-IO
    - HDM for DFS
    - Parallel FTP Daemon and Client
14. New utilities for automating SFS backups. (See 6.11)
15. PFTP supported on Intel-based Linux

M.3 Obsolete 3.2 Features

The following features and capabilities are no longer supported:

- PIOFS Import/Export.
- Ampex PVR
- NSL Unitree Migration
- FTP no longer supports the HPSS Policy Manager.
- Migration policies no longer supports the option to dictate whether or not migration for a storage class will use the BFS migration records. As of Release 4.1, all migrations automatically use the migration records to optimize performance.

M.4 Considerations for Using 4.1 Features

This section is provided for those sites currently running HPSS Release 3.2. Sites already running 4.1 will find no new information in this section.

M.4.1 DFS

For the initial distribution of Release 4.1, sites should consider using DFS as a beta version until it has been used for a longer period in more robust environments.

Some recommendations for using the DFS/HPSS filesets:

- DFS only filesets:
  - Should be used for disk resident files, smaller store, etc.

        - Will need to use standard DFS backup tools, but backups could be placed in HPSS if desired.

    - DFS / HPSS archived filesets:

        - Should be used to extend DFS storage space.

        - Episode file data will migrate into HPSS files. Data will be purged from Episode disks if migrated into HPSS or not accessed recently from DFS.

        - DFS backups will still need to be made since HPSS does not contain enough information to recover the name space.

        - Two types of archived filesets:

            - Archive / delete removes files from HPSS immediately when removed from Episode.

            - Archive / rename renames files stored in HPSS, so they may be recovered if the Episode disk is lost.

    - DFS / HPSS mirrored filesets:

        - Should be reserved for large files or files with high performance demands.

        - DFS and HPSS name space changes are kept synchronized, so name space update rates are similar to HPSS name space rates.

        - Good for loading large files quickly with HPSS then using DFS to access pieces of the files as needed.

M.4.2 File Families

File Families impose partitions on the tape library and should be used with the understanding that a large number of unfilled tapes may accumulate in a library if the system is configured with a large number of file families. The Tape Storage Server will assign empty tapes from the free tape pool (family zero) to the various file families on an as-needed basis. If the number of files in these families is not sufficient to fill these tapes, the tapes will remain unfilled as they cannot be assigned to other families.

By the same token, a large number of file families can degrade tape migration performance. In any given migration run, a relatively large number of tapes may need to be mounted to complete the migration, if the files to be migrated span a large number of families.

M.4.3 Location Server

In 3.2, the Client API contacted the Bitfile Server and Name Server directly by using their CDS pathnames specified with environment variables or by using their default CDS pathnames. As of Release 4.1, the Location Server helps the Client API to locate the local Bitfile Server and Name Server as well as any other server the client API needs to contact, such as the DMAP Gateway server. Then the Client API only need to know the local Location Server's whereabouts. The Client API uses the DCE rpcgroup, a list of CDS pathnames of one or more local Location Servers that can

process client requests, to find a Location Server. This DCE rpcgroup is maintained by the Location Server.

In 4.1, the Location Server performs the Class of Service selection when files are created. This allows it to select a Class of Service and a Bitfile Server to create the file under. In 3.2 all Class of Service selection was performed by the Bitfile Server alone. In 4.1, the Bitfile Server continues to perform Class of Service selection for files that it already controls.

M.4.4 Mover device initial disk offset

In HPSS 4.1, each Mover disk device can be configured with an initial starting device offset (default value is zero). This allows part of the disk (at the start of the partition) to be left untouched by HPSS so that any control information that may be written at the start of the partition remains intact. Under AIX, the Logical Volume Manager (LVM) writes a header / control block on the first 512 bytes of each volume. This information is currently required when performing control operations on the volume (e.g., when removing a disk from a stripe group). By setting the initial starting device offset value beyond the LVM control information, it will be untouched by any HPSS processing.

M.4.5 Support for disk partitions greater than 2GB

In HPSS 4.1, the Mover includes support for disk partitions greater than 2GB, with the prerequisite that the Mover be running on an AIX 4.2.1 or later platform. Prior to AIX 4.2.1, asynchronous I/O support did not support accessing logical volumes greater than 2GB, and therefore the HPSS Mover contained checks to prevent having a disk device larger than 2GB. These checks are now replaced with checks for the maximum allowable disk partition size (currently 1TB).

M.5 Considerations for New 4.1.1 Features

M.5.1 Non-DCE Client API

The Non-DCE Client API attempts to be as accurate as possible in emulating the standard Client API. However, the current implementation of the NDAPI does not perform any authentication as to the client's identity. The client's UID from the his host system is translated directly into a DCE UID for the DCE cell in which the HPSS system is running. This means that, depending on the UID of the client, virtually any DCE identity (and corresponding permissions in HPSS) can be assumed.

M.6 Upgrading from Release 3.2 to Release 4.1.1

This section is only applicable to sites currently running with HPSS Release 3.2. Sites that are going to install and configure HPSS Release 4.1.1 from scratch will not need to perform these upgrade procedures.

For sites that are currently running with the initial distribution of HPSS Release 4.1, refer to Section M.7 for the procedures to upgrade to run with the 4.1.1 patch.

Due to the possible risk of losing HPSS metadata, it is strongly recommended that the upgrade procedures are planned and performed with the help of the site's HPSS Support Representative.

This section describes the changes which must be made to the existing 3.2 system to fully upgrade it to Release 4.1.1. If a site is currently running with the HPSS 3.2, it must perform the upgrade process described in the following sections prior to using the HPSS 4.1.1 capabilities.

M.6.1 What must be changed?

To support the Release 4.1.1 requirements, the data structures of the following SFS metadata files were changed and must be converted to 4.1.1 format:

- Accounting Policy (**accounting**)
- Name Server Configuration (**cns**)
- BFS Configuration (**bfs**)
- BFS Storage Segment Unlinks (**bfssunlink**)
- Bitfile Migration Records (**bfmigrec**)
- Bitfile COS Changes (**bfcoschange**)
- SS Disk Storage Maps (**storagemapdisk**)
- SS Tape Storage Maps (**storagemaptape**)
- Mover Devices Configuration (**moverdevice**)
- PVL Drives Configuration (**pvldrive**)
- PVR Configuration (**pvr**)

In addition, Release 4.1.1 requires the creation of the following new SFS metadata files:

- Accounting Summary Records (**acctsum**)
- Accounting Log Records (**acctlog**)
- Accounting Snapshot Records (**acctsnap**)
- DMAP Gateway Configuration (**dmg**)
- DMAP Gateway Fileset (**dmgfileset**)
- File Family (**filefamily**)
- Location Policy (**lspolicy**)
- NS Global Filesets (**nsglobalfilesets**)
- NS Fileset Attributes (**nsfilesetattrs**)
- Remote HPSS Sites (**site**)

- Non-DCE Client Gateway Configuration (**ndcg**)
- AML Cartridges (**cartridge_aml**)

Other important changes:

- Must upgrade OS to AIX 4.2.1 or 4.3.2
- Must upgrade to Encina TXSeries 4.2, if not already running with this version
- Changes in Sammi distribution and installation
- Must configure new 4.1.1 servers
- Changes in the BFS and NS Security Object ACLs
- Changes in the **/usr/lpp/hpss/config/hpss_env** environment file
- Changes in the FTP configuration files

M.6.2 Prerequisites for 4.1.1 Upgrade

It is mandatory that a full backup of the 3.2 SFS metadata is made available before upgrading the metadata to Release 4.1.1 format.

Existing HPSS 3.2 system must already run with the hpss.3.2_aix4.2_patch030428 patch and the Name Server text conversion required for this patch must have already been performed.

All 3.2 migration policies must have the Use Migration File flags set to ON. If this flag is not set ON for any storage class, modify the migration policy then run the mkmprec utility to generate the required migration and purge records.

All 3.2 purge policies must have the Use Purge File flags set to ON. If this flag is not set ON for any storage class, modify the purge policy then run the mkmprec utility to generate the required migration and purge records (if not already done above.)

1. Must have sufficient disk space, either on the SFS data volume or on another AIX disk volume, to save a copy of the 3.2 SFS metadata files via SFS backup or other means such as export/copy.
2. Must have sufficient SFS space to convert the following R3.2 SFS files to R4.1.1 metadata format:
   - Accounting Policy (**accounting**)
   - Name Server Configuration (**cns**)
   - BFS Configuration (**bfs**)
   - BFS Storage Segment Unlinks (**bfssunlink**)
   - Bitfile Migration Records (**bfmigrec**)
   - Bitfile COS Changes (**bfcoschange**)

- SS Disk Storage Maps (**storagemapdisk**)
- SS Tape Storage Maps (**storagemaptape**)
- Mover Devices Configuration (**moverdevice**)
- PVL Drives Configuration (**pvldrive**)
- PVR Configuration (**pvr**)

During the metadata conversion process, the 3.2 SFS files are renamed and the new files are created to store the converted metadata. Refer to Section 2.9.2 for more information on the sizing of the above files.

3. Must have sufficient SFS space to create the following 4.1.1 SFS files (Refer to Section 2.9.2.3 for more information on their sizing):
   - Accounting Summary Records (**acctsum**)
   - Accounting Log Records (**acctlog**)
   - Accounting Snapshot Records (**acctsnap**)
   - DMAP Gateway Configuration (**dmg**)
   - DMAP Gateway Fileset (**dmgfileset**)
   - File Family (**filefamily**)
   - Location Policy (**lspolicy**)
   - NS Global Filesets (**nsglobalfilesets**)
   - NS Fileset Attributes (**nsfilesetattrs**)
   - Remote HPSS Sites (**site**)
   - Non-DCE Client Gateway Configuration (**ndcg**)
   - AML Cartridges (**cartridge_aml**)
4. Must have already upgraded all machines to AIX Version 4.2.1 or AIX 4.3.2.
5. Must have already installed DCE Version 2.1 and / or DCE Version 2.2
6. Must have the Encina TXSeries 4.2 distribution tape.
7. Must have the Sammi (Version 4.1.2) distribution tape and license key.
8. Must have the HPSS 4.1.1 distribution tape.
9. Must have **root**, **cell_admin, hpss_ssm,** and **encina_admin** authority.

M.6.3 Preparation for 4.1.1 Upgrade

The following steps must be performed with the HPSS 3.2 code still in place. Ensure that all 3.2 upgrade procedures had been performed.

1. Shut down all 3.2 HPSS servers. Verify that all servers are down before proceeding.

2. Upgrade existing Encina version to TXSeries 4.2, if not already runing with this version. Contact Transarc for guidance on the upgrade procedures to minimize risks of losing metadata.

3. Install Sammi and set up its license key using the procedures provided by Kinesix. As of Release 4.1, Sammi is no longer bundled with the HPSS code.

4. Logon as **root**

5. **dce_login** using the **encina_admin** principal and password. Set the **ENCINA_SFS_SERVER** environment variable with the correct SFS server name.

6. Save a copy of the following 3.2 SFS files using the "**sfsadmin export file <fn>**" or the "**sfsadmin copy file <fn>"** command:

   - Accounting Policy (**accounting**)
   - Name Server Configuration (**cns**)
   - BFS Configuration (**bfs**)
   - BFS Storage Segment Unlinks (**bfssunlink**)
   - Bitfile Migration Records (**bfmigrrec**)
   - Bitfile COS Changes (**bfcoschange**)
   - SS Disk Storage Maps (**storagemapdisk**)
   - SS Tape Storage Maps (**storagemaptape**)
   - Mover Devices Configuration (**moverdevice**)
   - PVL Drives Configuration (**pvldrive**)
   - PVR Configuration (**pvr**)

7. Save HPSS 3.2 code and prepare for 4.1.1 installation, as follows:

   - cd to **/usr/lpp** directory (assuming that 3.2 was installed in **/usr/lpp/hpss**)
   - Move the **hpss** directory to **hpss_32** (for example: **mv hpss hpss_32**)
   - Create **hpss** and **hpss/config** directory (for example: **mkdir hpss hpss/config)**
   - cd to /**usr/lpp/hpss/config** directory

- Copy 3.2 **config** files into 4.1.1 **config** directory (for example **cp -pR /usr/lpp/hpss_32/config/* .**)

8. Save the 3.2 Unix configuration files in the **/var/hpss** directory, as follows:

   - cd to **/var** directory
   - Move the **hpss** directory to **hpss_32** (for example **mv hpss hpss_32**)

9. Save a copy of the **/krb5/hpss.keytabs** and **/krb5/hpssclient.keytab** files.

M.6.4 Procedures to Upgrade HPSS 3.2 to 4.1.1

The 4.1.1 upgrade procedures must be performed in the following order:

1. Install the HPSS 4.1.1 code. Use the installation procedures described in Section 3.3 of Chapter 3. This procedure assumes that the HPSS 4.1.1 code will be installed in the **/usr/lpp/hpss** directory.

2. Perform the following post installation procedures:

   - Copy the **/usr/lpp/hpss/config/hpss_env.default** to **hpss_env**. Review this file and update it using the data from the **/usr/lpp/hpss_32/config/hpss_env** file. Refer to Section 4.3 for more information on the **hpss_env** file.
   - In the special case where the HPSS 4.1.1 executables are to be rebuilt to meet the site's special needs, customize the **Makefile.macros** file as necessary. Refer to Section 3.4.2 for more information on the **Makefile.macros** file and the procedures to rebuild HPSS executables.
   - Run **mkhpss** to do the initial infrastructure setup as follow:
     - Invoke the **/usr/lpp/hpss/config/mkhpss** script.
     - Choose the **add-on** option.
     - Exit **mkhpss**

     This **mkhpss** option sets up the link for the **$SAMMI/bin** and **$SAMMI/data** directory. It also set up the ownership and permission for the HPSS files.
   - Restore and update the R3.2 Unix configuration files, as follows:
     - **cd /var**
     - **mv hpss hpss_41**
     - **cp -R hpss_32 hpss**
     - **cd /var/hpss/ftp/etc**
     - **cp /usr/lpp/hpss/config/templates/ftpaccess.template ftpaccess**

- Review the 4.1.1 **/var/hpss/ftp/etc/ftpaccess** file**.** Update it with the customized data from the saved /**var/hpss_32/ftp/etc/ftpaccess** file. Refer to Section 5.8.3 for more information on the supported **ftpaccess** options.

- Edit the /**etc/inetd.conf** file to remove the obsolete "**-BH**" flags from the **hpssftp** entry, then add the "**-L**" option to this entry. Refer to Section 5.8.3 for more information on the supported **inetd.conf** options

- Issue the "**refresh -s inetd**" after modifying the **inetd.conf** file.

- Copy the **/usr/lpp/hpss_32/sammi/ssmuser** directory and all of its contents to the **/usr/lpp/hpss/sammi** directory, as follows:

  - **cd /usr/lpp/hpss/sammi**

  - **cp -pR /usr/lpp/hpss_32/sammi/ssmuser .**

  - Ensure that the SSM user files have the appropriate ownerships and permissions. In addition, modify the permissions for the **/usr/lpp/hpss/sammi/ssmuser/<userid>** directories to allow the SSM Data Server to create the user's specific preference files.

- Restore the SSM **api_config.dat** file as follows:

  - **cd /usr/lpp/hpss/bin**

  - **cp /usr/lpp/hpss_32/bin/api_config.dat .**

- Restore the SSM **user_authorization.dat** file as follows:

  - **cd /usr/lpp/hpss/sammi/hpss_ssm**

  - **cp /usr/lpp/hpss_32/sammi/hpss_ssm/user_authorization.dat .**

- Replace the R3.2 **rc.hpss** script, as follows:

  - **mv /etc/rc.hpss /etc/rc.hpss_32**

  - **cp /usr/lpp/hpss/tools/start/rc.hpss /etc/rc.hpss**

3. Set up the DCE identities for the Location Server, DMAP Gateway and Non-DCE Client Gateway.

   - Invoke the **/usr/lpp/hpss/bin/create_server_identity** script to create the DCE accounts, principals and keytab entries for the Location Server and the DMAP Gateway. The two new keys will be added to the existing server keytab file with known passwords. The utility then will prompt whether the keytab entries in the keytab file are to be randomized and registered in the DCE registry.

   - Invoke the **/usr/lpp/hpss/bin/create_ndcg_identity** script to create the DCE account, principal and keytab entry for the Non-DCE Client Gateway. The new key will be added to the existing server keytab file with known passwords. The utility then will prompt whether the keytab entry in the keytab file is to be randomized and registered in the DCE registry.

4. **dce_login** as **encina_admin** to convert the SFS files described in step 5 through step 16.

5. Convert the Accounting Policy SFS file (**accounting**).

   - Invoke the "**sfsadmin query file accounting**" command and note the number of 3.2 record. There should only be one entry.

   - Invoke the **/usr/lpp/hpss/bin/convert_acctconfig** utility to convert the 3.2 entry into 4.1.1 metadata format. If there are any errors, retry the conversion by following the instructions output by the utility. Contact the HPSS Support Representative if the problem persists.

   - If the conversion is successful, invoke the "**sfsadmin query file accounting**" command and verify that the 4.1.1 file has the same number of records as the 3.2 file. If not, rename the **accounting.convert.old** SFS file to **accounting** and retry the conversion. Contact the HPSS Support Representative if the problem persists.

6. Convert Name Server Configuration SFS File (**cns**) and its CDS Security Object.

   - Invoke the "**sfsadmin query file cns**" command and note the number of 3.2 records.

   - Invoke the **/usr/lpp/hpss/bin/convert_nsconfig** utility to convert the 3.2 entries into 4.1.1 metadata format. If there are any errors, retry the conversion by following the instructions output by the utility. Contact the HPSS Support Representative if the problem persists.

     Note that as part of this conversion, the utility will rename the **cns**, **cnsobjects**, **cnstext** and **cnsacls** SFS files to **nsconfig**, **nsobjects**, **nstext** and **nsacls**, respectively.

   - If the conversion is successful, invoke the "**sfsadmin query file nsconfig**" command and verify that the 4.1.1 file has the same number of records as the 3.2 file. If not, rename the **cns.convert.old** SFS file to **cns** and retry the conversion. Contact the HPSS Support Representative if the problem persists.

   - If the conversion is successful, **dce_login** as **hpss_ssm** (using the **-k /krb5/hpss.keytabs** option, if needed) then invoke the **/usr/lpp/hpss/bin/convert_nssecacls** utility to set the ACLs for the NS Security Object appropriately for Release 4.1.1. If there is any errors, the **/usr/lpp/hpss/bin/restore_nssecacls** utility may be run to restore the Name Server's Security Object to 3.2 format. Contact the HPSS Support Representative if the problem persists. Destroy the **hpss_ssm** credential and verify that the **encina_admin** credential is active before proceeding with the next step.

It is very important that the convert_nssecacls script be run to set the ACLs for the NS Security Object appropriately. If this script is not run, some users may be inadvertently granted full root privileges to the HPSS name space. After the script is run, invoke the "dcecp -c acl show -entry <NS CDS Security Object Name>" DCE command to review the ACLs. The output from this command should be as follows:

$ dcecp -c acl show -entry /.:/hpss/cns/Security

{unauthenticated r--t-}

{user hpss_ssm rwdtc}

{user hpss_dmg rwdtc}

{user hpss_nfs2 r-dtc}

{user hpss_ndcg r-dtc}

{user hpss_bfs r-dtc}

{user hpss_ftp r-dtc}

{group subsys/dce/cds-admin rwdtc}

{group subsys/dce/cds-server rwdtc}

{any_other r--t-}

7. Convert the BFS Configuration SFS file (**bfs**) and its CDS Security Object.

   - Invoke the "**sfsadmin query file bfs**" command and note the number of 3.2 records.

   - Invoke the **/usr/lpp/hpss/bin/convert_bfsconfig** utility to convert the 3.2 entries into 4.1.1 metadata format. If there are any errors, retry the conversion by following the instructions output by the utility. Contact the HPSS Support Representative if the problem persists.

   - If the conversion is successful, invoke the "**sfsadmin query file bfs**" command and verify that the 4.1.1 file has the same number of records as the 3.2 file. If not, rename the **bfs.convert.old** SFS file to **bfs** and retry the conversion. Contact the HPSS Support Representative if the problem persists.

   - If the conversion is successful, **dce_login** as **cell_admin** and invoke the **/usr/lpp/hpss/bin/convert_bfssecacls** utility to set the ACLs for the BFS Security Object appropriately for Release 4.1.1.The updated ACLs, as displayed from the conversion output, should be as follows:

     Updated ACLs for /.:/hpss/bfs/Security:

     {unauthenticated r--t-}

     {user hpss_ssm rwdtc}

     {user hpss_nfs2 rwdtc}

     {user hpss_mps rwdtc}

     {user hpss_dmg rwdtc}

     {user hpss_cns rwdtc}

     {group subsys/dce/cds-admin rwdtc}

     {group subsys/dce/cds-server rwdtc}

     {any_other r--t-}

     Destroy the **cell_admin** credentials and verify that the **encina_admin** credentials are active before proceeding with the next step.

8. Convert the BFS Storage Segment Unlinks SFS file (**bfssunlink**).

   - Invoke the "**sfsadmin query file bfssunlink**" command and note the number of 3.2 records.

   - Invoke the **/usr/lpp/hpss/bin/convert_ssunlink** utility to convert the 3.2 entries into 4.1.1 metadata format. If there are any errors, retry the conversion by following the instructions output by the utility. Contact the HPSS Support Representative if the problem persists.

- If the conversion is successful, invoke the "**sfsadmin query file bfssunlink**" command and verify that the 4.1.1 file has the same number of records as the 3.2 file. If not, rename the **bfssunlink.convert.old** SFS file to **bfssunlink** and retry the conversion. Contact the HPSS Support Representative if the problem persists.

9. Convert the Bitfile Migration Records SFS file (**bfmigrrec**).

   - Invoke the "**sfsadmin query file bfmigrrec**" command and note the number of 3.2 records.

   - Invoke the **/usr/lpp/hpss/bin/convert_migrfile** utility to convert the 3.2 entries into 4.1.1 metadata format. If there are any errors, retry the conversion by following the instructions output by the utility. Contact the HPSS Support Representative if the problem persists.

   - If the conversion is successful, invoke the "**sfsadmin query file**" command and verify that the 4.1.1 file has the same number of records as the 3.2 file. If not, rename the **bfmigrrec.convert.old** SFS file to **bfmigrrec** and retry the conversion. Contact the HPSS Support Representative if the problem persists.

10. Convert the Bitfile COS Change SFS file (**bfcoschange**).

    - Invoke the "**sfsadmin query file bfcoschange**" command and note the number of 3.2 records.

    - Invoke the **/usr/lpp/hpss/bin/convert_coschange** utility to convert the 3.2 entries into 4.1.1 metadata format. If there are any errors, retry the conversion by following the instructions output by the utility. Contact the HPSS Support Representative if the problem persists.

    - If the conversion is successful, invoke the "**sfsadmin query file**" command and verify that the 4.1.1 file has the same number of records as the 3.2 file. If not, rename the **bfcoschange.convert.old** SFS file to **bfcoschange** and retry the conversion. Contact the HPSS Support Representative if the problem persists.

11. Convert the SS Disk Storage Maps SFS File (**storagemapdisk**).

    - Invoke the "**sfsadmin query file storagemapdisk**" command and note the number of 3.2 records.

    - Invoke the **/usr/lpp/hpss/bin/convert_diskmaps** utility to convert the 3.2 entries into 4.1.1 metadata format. If there are any errors, retry the conversion by following the instructions output by the utility. Contact the HPSS Support Representative if the problem persists.

    - If the conversion is successful, invoke the "**sfsadmin query file storagemapdisk**" command and verify that the 4.1.1 file has the same number of records as the 3.2 file. If not, delete the **storagemapdisk** file, rename the **storagemapdisk.convert.old** SFS file to **storagemapdisk** and retry the conversion. Contact the HPSS Support Representative if the problem persists.

12. Convert the SS Tape Storage Maps SFS File (**storagemaptape**).

    - Invoke the "**sfsadmin query file storagemaptape**" command and note the number of 3.2 records.

- Invoke the **/usr/lpp/hpss/bin/convert_tapemaps** utility to convert the 3.2 entries into 4.1.1 metadata format. If there are any errors, retry the conversion by following the instructions output by the utility. Contact the HPSS Support Representative if the problem persists.

- If the conversion is successful, invoke the "**sfsadmin query file storagemaptape**" command and verify that the 4.1.1 file has the same number of records as the 3.2 file. If not, delete the **storagemaptape** file, rename the **storagemaptape.convert.old** SFS file to **storagemaptape** and retry the conversion. Contact the HPSS Support Representative if the problem persists.

- If the conversion is successful, invoke the **/usr/lpp/hpss/bin/settapestats** utility as follows to calculate and set the statistics for the Release 4.1 Tape Storage Server.

  % **. /usr/lpp/hpss/config/hpss_env**

  % /usr/lpp/hpss/bin/settapestats -g /.:/encina/sfs/hpss/serverconfig -u update_both

  Refer to Section I.53 in the Appendix I for more information on how to invoke the **settapestats** utility.

13. Convert the Mover Device Configuration SFS file (**moverdevice**).

    - Invoke the "**sfsadmin query file moverdevice**" command and note the number of 3.2 records.

    - Invoke the **/usr/lpp/hpss/bin/convert_mvrdevice** utility to convert the 3.2 entries into 4.1.1 metadata format. If there are any errors, retry the conversion by following the instructions output by the utility. Contact the HPSS Support Representative if the problem persists.

    - If the conversion is successful, invoke the "**sfsadmin query file**" command and verify that the 4.1.1 file has the same number of records as the 3.2 file. If not, rename the **moverdevice.convert.old** SFS file to **moverdevice** and retry the conversion. Contact the HPSS Support Representative if the problem persists.

14. Convert the PVL Drive Configuration SFS file (**pvldrive**).

    - Invoke the "**sfsadmin query file pvldrive**" command and note the number of 3.2 records.

    - Invoke the **/usr/lpp/hpss/bin/convert_pvldrive** utility to to add the secondary index, **PVLMountedVolIndex**, to the SFS file. If there is a need to undo this conversion, the "*sfsadmin delete index pvldrive PVLMountedVolIndex*" command can be used to delete the secondary index. Contact the HPSS Support Representative if the problem persists.

    - If the conversion is successful, invoke the "**sfsadmin query file**" command and verify that the 4.1.1 file has the same number of records as the 3.2 file. Contact the HPSS Support Representative if the problem persists.

15. Convert the PVR Configuration SFS file (**pvr**).

    - Invoke the "**sfsadmin query file pvr**" command and note the number of 3.2 records.

    - Invoke the **/usr/lpp/hpss/bin/convert_pvrconfig** utility to convert the 3.2 entries into 4.1.1 metadata format. If there are any errors, retry the conversion by following the instructions output by the utility. Contact the HPSS Support Representative if the problem persists.

- If the conversion is successful, invoke the "**sfsadmin query file pvr**" command and verify that the 4.1.1 file has the same number of records as the 3.2 file. If not, rename the **pvr.convert.old** SFS file to **pvr** and retry the conversion. Contact the HPSS Support Representative if the problem persists.

16. Create new 4.1.1 SFS files.
    - Invoke **managesfs** to create the following files:
        - Accounting Summary Records (**acctsum**)
        - Accounting Log Records (**acctlog**)
        - Accounting Snapshot Records (**acctsnap**)
        - DMAP Gateway Configuration (**dmg**)
        - DMAP Gateway Fileset (**dmgfileset**)
        - File Family (**filefamily**)
        - Location Server Policies (**lspolicy**)
        - NS Global Filesets (**nsglobalfilesets**)
        - NS Fileset Attributes (**nsfilesetattrs**)
        - Remote Site Configurations (**site**)
        - Non-DCE Client Gateway Configuration (**ncdg**), if required
        - AML Cartridges (**cartridge_aml**), if required
17. Start the SSM servers and bring up an SSM session.
18. Bring up the NS Basic Server Configuration window and change the **Server Configuration File (SFS)** field from "**cns**" to "**nsconfig**".
19. Configure the following Release 4.1.1 entities via SSM:
    - Location Server (required).
    - Location Policy (required)
    - AML PVR (if desired)
    - File Families (if desired)
    - Filesets (if desired)
    - Junctions (if desired)
    - Configure DFS (if desired)

- DMAP Gateway (if DFS is to be used)
- Non-DCE Client Gateway (if Non-DCE Client is to be used)

20. Review the following metadata:

    - **Accounting Policy**. Verify that the new fields are set to the desired value. In addition, verify that other fields are unchanged.
    - **NS specific configuration**. Verify that all SFS file names are correct and that other fields are unchanged.
    - **BFS specific configuration**. Verify that all SFS file names are correct and that other fields are unchanged.
    - **Mover basic and specific configuration**. For each DCE Mover, verify that the **Execute Hostname** field in the Mover Basic Configuration window and the **Hostname** field in the Mover Specific Configuration window are identical.
    - **PVR specific configuration**. Verify that all SFS file names are correct and that other fields are unchanged.
    - **Disk device configurations**. Ensure that the disk **Starting Offset** fields are set to zeroes.

21. Remotely install HPSS code on all HPSS (Mover and HDM) machines by running the **mkhpss** program. Refer to Section 4.6.5 for more information on how to install HPSS on remote nodes.

22. Bring up the Startup Daemon(s) and then all HPSS servers. The 4.1.1 HPSS system is fully operational at this time. The upgrade verification procedures described in Section M.6.5 must be performed before allowing the HPSS users to use the Release 4.1.1 system.

23. If HPSS accounting is used, invoke the **/usr/lpp/hpss/bin/acct_convert** utility to create the HPSS accounting summary information. When started, the utility will output some information on the selected parameters and begin the conversion. During the run, HPSS will continue to be available. Upon completion, the utility will give a summary of the conversion along with additional instructions if errors were encountered. If desired, the utility can be invoked as follows to reduce the work load on the HPSS system:

    % **acct_convert -s <# of microseconds to sleep between bitfiles>**

    Issue the "**acct_convert ?**" command for more information on invoking the utility.

The account conversion utility will read every bitfile record in the HPSS system and could easily take several days to run on a large system. Also, the large amount of metadata activity will cause an increased number of log archive (LA) files to be generated, and the filesystem containing the LA files should be carefully monitored during the conversion to ensure that it doesn't fill up and cause an SFS crash. Finally, it is recommended to use a sleep factor other than 0 to keep from bogging down the system. The exact value will depend mainly on the number of bitfiles in the HPSS system, since there will be a linear increase in the time it takes to perform the account conversion that depends solely on the number of bitfiles. For each million files in the system, each unit of the sleep factor causes the run time to be one second longer. (e.g. A sleep factor of 1000 for a system with 5 million files will increase the conversion time by 5000 seconds, or ~83 minutes).

M.6.5 Release 4.1.1 Upgrade Verification

After all 4.1.1 upgrade procedures are completed, use each of the configured user interfaces to create new files and access files which were created in 3.2 to verify that the conversions were performed successfully. In addition, monitor the migration and purge operations to ensure that they are working properly.

M.6.6 Post Release 4.1.1 Upgrade Cleanup

After the 4.1.1 system has been operational for an extended period, delete the following files:

- All backed up SFS files.
- All ***.convert.old** SFS files.
- Delete the HPSS 3.2 code saved in the **/usr/lpp/hpss_32** directory
- Delete the **/var/hpss_32** and **/var/hpss_41** directories.
- Other files backed up for the 4.1.1 upgrade.

M.7 Upgrading from Release 4.1 to 4.1.1

The procedure to upgrade HPSS Release 4.1 to 4.1.1 consists of a subset of the steps described in Section M.6. They must be performed in the following order:

M.7.1 Prerequisites for 4.1.1 Upgrade

It is mandatory that a full backup of the 4.1 SFS metadata is available before upgrading the metadata to Release 4.1.1 format.

1. Must have the HPSS 4.1.1 distribution tape.
2. Must have already upgraded all machines to AIX Version 4.2.1 or AIX 4.3.2.
3. Must have **root** and **encina_admin** authority.

M.7.2 Preparation for 4.1.1 Upgrade

The following steps must be performed with the HPSS 4.1 code still in place.

1. Shut down all 4.1 HPSS servers. Verify that all servers are down before proceeding.
2. Logon as **root**
3. **dce_login** using the **encina_admin** principal and password. Set the **ENCINA_SFS_SERVER** environment variable with the correct SFS server name.

4. Save a copy of the following 4.1 SFS files using the "**sfsadmin export file <fn>**" or the "**sfsadmin copy file <fn>"** command:

   - Bitfile COS Changes (**bfcoschange**)
   - PVR Configuration (**pvr**)

5. Save HPSS 4.1 code and prepare for 4.1.1 installation, as follows:

   - cd to **/usr/lpp** directory (assuming that 4.1 code was installed in **/usr/lpp/hpss**)
   - Move the **hpss** directory to **hpss_41** (for example: **mv hpss hpss_41**)
   - Create **hpss** and **hpss/config** directory (for example: **mkdir hpss hpss/config)**
   - cd to /**usr/lpp/hpss/config** directory
   - Copy 4.1 **config** files into 4.1.1 **config** directory (for example **cp -pR /usr/lpp/hpss_41/config/* .**)

6. Save the 4.1 Unix configuration files in the **/var/hpss** directory, as follows:

   - cd to **/var** directory
   - Move the **hpss** directory to **hpss_41** (for example **mv hpss hpss_41**)

7. Save a copy of the **/krb5/hpss.keytabs** and **/krb5/hpssclient.keytab** files.

M.7.3 Procedures to Upgrade HPSS 4.1 to 4.1.1

1. Install the HPSS 4.1.1 code. Use the installation procedures described in Section 3.3 of Chapter 3. This procedure assumes that the HPSS 4.1.1 code will be installed in the **/usr/lpp/hpss** directory.

2. Perform the following post installation procedures:

   - Run **mkhpss** to do the initial infrastructure setup as follow:
     - Invoke the **/usr/lpp/hpss/config/mkhpss** script.
     - Choose the **add-on** option.
     - Exit **mkhpss**

     This **mkhpss** option sets up the link for the **$SAMMI/bin** and **$SAMMI/data** directory. It also set up the ownership and permission for the HPSS files.
   - In the special case where the HPSS 4.1.1 executables are to be rebuilt to meet the site's special needs, customize the **Makefile.macros** file as necessary. Refer to Section 3.4.2 for more information on the **Makefile.macros** file and the procedures to rebuild HPSS executables.
   - Restore and update the 4.1 Unix configuration files, as follows:

- **cd /var**
- **mv hpss hpss_411**
- **cp -R hpss_41 hpss**

- Copy the **/usr/lpp/hpss_41/sammi/ssmuser** directory and all of its contents to the **/usr/lpp/hpss/sammi** directory, as follows:
  - **cd /usr/lpp/hpss/sammi**
  - **cp -pR /usr/lpp/hpss_41/sammi/ssmuser .**
  - Ensure that the SSM user files have the appropriate ownerships and permissions. In addition, modify the permissions for the **/usr/lpp/hpss/sammi/ssmuser/<userid>** directories to allow the SSM Data Server to create the user's specific preference files.
- Restore the SSM **api_config.dat** file as follows:
  - **cd /usr/lpp/hpss/bin**
  - **cp /usr/lpp/hpss_41/bin/api_config.dat .**
- Restore the SSM **user_authorization.dat** file as follows:
  - **cd /usr/lpp/hpss/sammi/hpss_ssm**
  - **cp /usr/lpp/hpss_41/sammi/hpss_ssm/user_authorization.dat .**

3. Set up the DCE identity for the Non-DCE Client Gateway.
   - Invoke the **/usr/lpp/hpss/bin/create_ndcg_identity** script to create the DCE account, principal and keytab entry for the Non-DCE Client Gateway. The new key will be added to the existing server keytab file with known passwords. The utility then will prompt whether the keytab entry in the keytab file is to be randomized and registered in the DCE registry.
4. Convert the Bitfile COS Change SFS file (**bfcoschange**).
   - Invoke the "**sfsadmin query file bfcoschange**" command and note the number of 4.1 records.
   - Invoke the **/usr/lpp/hpss/bin/convert_coschange** utility to convert the 4.1 entries into 4.1.1 metadata format. If there are any errors, retry the conversion by following the instructions output by the utility. Contact the HPSS Support Representative if the problem persists.
   - If the conversion is successful, invoke the "**sfsadmin query file**" command and verify that the 4.1.1 file has the same number of records as the 4.1 file. If not, rename the **bfcoschange.convert.old** SFS file to **bfcoschange** and retry the conversion. Contact the HPSS Support Representative if the problem persists.
5. Convert the PVR Configuration SFS file (**pvr**).

- Invoke the "**sfsadmin query file pvr**" command and note the number of 4.1 records.
- Invoke the **/usr/lpp/hpss/bin/convert_pvrconfig** utility to convert the 4.1 entries into 4.1.1 metadata format. If there are any errors, retry the conversion by following the instructions output by the utility. Contact the HPSS Support Representative if the problem persists.
- If the conversion is successful, invoke the "**sfsadmin query file pvr**" command and verify that the 4.1.1 file has the same number of records as the 4.1 file. If not, rename the **pvr.convert.old** SFS file to **pvr** and retry the conversion. Contact the HPSS Support Representative if the problem persists.

6. Create new 4.1.1 SFS files.
   - Invoke **managesfs** to create the Non-DCE Client Gateway Configuration (**ncdg**) file, if Non-DCE Client is to be used.
7. Start the SSM servers and bring up an SSM session.
8. Configure the Non-DCE Client Gateway (if Non-DCE Client is to be used) via SSM.
9. Review the converted PVR specific configuration to ensure that the configuration data is correct.
10. Review the Mover basic and specific configuration. For each DCE Mover, verify that the **Execute Hostname** field in the Mover Basic Configuration window and the **Hostname** field in the Mover Specific Configuration window are identical.
11. Remotely install HPSS code on all HPSS (Mover and HDM) machines by running the **mkhpss** program. Refer to Section 4.6.5 for more information on how to install HPSS on remote nodes.
12. Bring up the Startup Daemon(s) and then all HPSS servers. The 4.1.1 HPSS system is fully operational at this time. The upgrade verification procedures described in Section M.6.5 must be performed before allowing the HPSS users to use the Release 4.1.1 system.
13. Post Release 4.1.1 Upgrade Cleanup

    After the 4.1.1 system has been operational for an extended period, delete the following files:
    - All backed up SFS files.
    - All ***.convert.old** SFS files.
    - Delete the HPSS 4.1 code saved in the **/usr/lpp/hpss_41** directory
    - Delete the **/var/hpss_41** and **/var/hpss_411** directories.
    - Other files backed up for the 4.1.1 upgrade.

Appendix N

Developer Acknowledgments

HPSS is a product of a government-industry collaboration. The project approach is based on the premise that no single company, government laboratory, or research organization has the ability to confront all of the system-level issues that must be resolved for significant advancement in high-performance storage system technology.

HPSS development was performed jointly by IBM Worldwide Government Industry, Lawrence Livermore National Laboratory, Los Alamos National Laboratory, NASA Langley Research Center, Oak Ridge National Laboratory, and Sandia National Laboratories.

We would like to acknowledge Argonne National Laboratory, the National Center for Atmospheric Research, and Pacific Northwest Laboratory for their help with initial requirements reviews.

We also would like to acknowledge Cornell Information Technologies of Cornell University for providing assistance with naming service and transaction management evaluations and for joint developments of the Name Server component.

We also wish to acknowledge the many discussions, design ideas, implementation and operation experiences we have shared with colleagues at the National Storage Laboratory, the IEEE Mass Storage Systems and Technology Technical Committee, the IEEE Storage System Standards Working Group, and the storage community at large.

Finally, we want to acknowledge the Cornell Theory Center and the Maui High Performance Computer Center for providing a test beds for the initial HPSS release.