UCRL-WEB-200067

EZFILES

Table of Contents

Preface

Scope: EZFILES explains the hierarchical structure of directories and file systems on LC computers, the properties and limits of those directories, the role of the common home directory, and the default search paths. It also introduces some file-management software tools that monitor and report on important system features (such as quotas), and other tools that manipulate file permissions, transfer files, or perform basic file-handling tasks. Tools (such as GIVE, MOLE, and HOPPER) and file features (such as citizenship groups and quotas) unique to the LC computing environment receive special attention in EZFILES. Backup policies, purge policies, and cross references to details about how LC provides parallel file systems are included as well.

Availability: When the programs described here are limited by machine, those limits are included in their explanation. Otherwise, they run under any LC UNIX system.

Consultant: For help contact the LC customer service and support hotline at 925-422-4531 (open e-mail: lc-hotline@llnl.gov, secure e-mail: lc-hotline@pop.llnl.gov).

Printing: The print file for this document can be found at:

```
OCF: http://www.llnl.gov/LCdocs/ezfiles/ezfiles.pdf
SCF: https://lc.llnl.gov/LCdocs/ezfiles/ezfiles_scf.pdf
```

Introduction

This manual provides a basic guide to effectively managing your files and directories on LC computers. Its goal is to include the most important and relevant tools, techniques, and features, while making them easy to find and learn by avoiding more esoteric features or comprehensive descriptions of every possible feature.

EZFILES begins with a comparative look at how files and directories are organized on LC machines, with an emphasis on the important differences between each machine's home and work directories. We then look at the file-management implications of sharing the home directories across (most) open machines at LC. Included also are suggested techniques for working effectively in the public directory structure, a summary of which directories are backed up automatically and which are not, and a concise explanation of the default search path on different systems. EZFILES then introduces CHMOD to manage file permissions, and offers basic instructions for between-user file exchange (with GIVE and TAKE), home-directory and /nfs/tmp* quota monitoring (with QUOTA), and other related file-management tasks. EZFILES gives special attention to file-handling tools unique to LC (not found in most other UNIX environments), by explaining some (e.g., MOLE, HOPPER) and by linking to the local user manuals that explain others (e.g., HTAR).

Some UNIX file-management tools behave differently under the Linux version of UNIX or have extra options for added file-handling tasks. Consult LC's Linux Differences (URL: http://www.llnl.gov/LCdocs/linux) user guide for alerts about and details on these Linux-only features. For further details on the Linux-only parallel file system (called Lustre), see the manual called CHAOS: Linux from Livermore (URL: http://www.llnl.gov/LCdocs/chaos) or review the relevant sections of the I/O Guide for LC (URL: http://www.llnl.gov/LCdocs/ioguide).

Directory Structure and Properties

This section shows how the important public directories (and their underlying file systems) are organized on the prime LC computers, and compares the home and work file environments on each computing system. The diagrams and tables make it easy to see the differences between different kinds of computer, or between computers of the same kind but differently configured on the open and secure networks.

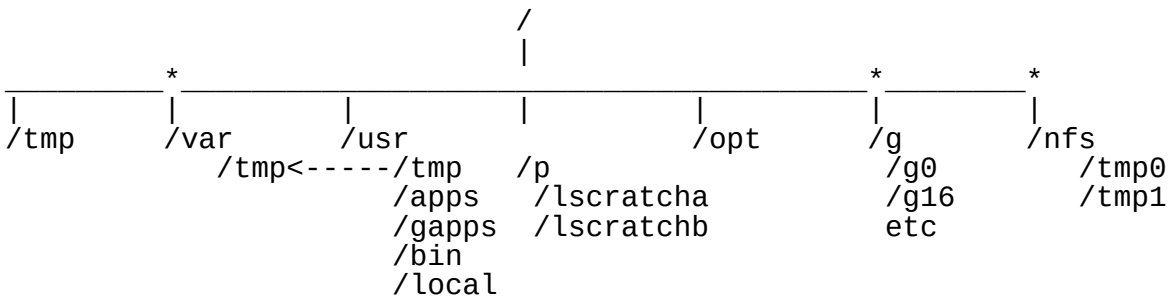
Linux/CHAOS Clusters (Open)

On the open network, LC offers several production Linux clusters. Some are ASC-sponsored (such as ALC) while others are M&IC-sponsored machines (such as Thunder). Differences in size, chip set, and available switch (interconnect) affect the scale and parallelism of applications for which each Linux cluster is best suited. See LC's frequently updated feature table of current OCF Resources (URL: http://www.llnl.gov/computing/hpc/resources/OCF_resources.html) for comparative technical details on the open LC Linux clusters.

All these computers run Red Hat Linux as modified for supercomputing at LLNL (CHAOS (URL: http://www.llnl.gov/LCdocs/chaos)). And all have the same basic file-system hierarchy and public-directory properties, described in the subsections that follow, except where local differences are explicitly noted.

File-System Hierarchy

Each machine among the open Linux/CHAOS clusters has a public directory structure of the form shown here (key directories are marked *):



where

/tmp — is just a link to /var/tmp (for system use).

/usr/tmp — is simply a convenient link pointing to /var/tmp (for general use).

/var/tmp — is the intended main working directory. You should create a subdirectory here (or in /usr/tmp if you prefer) and conduct most computational work in it. See the next section for a comparative look at the properties of /var/tmp (keyword: linux-properties (page 8)), and see the section on file management techniques (keyword: work-steps (page 25)) for a summary of sound work practices. Note that on diskless clusters /var/tmp uses real memory and so CHAOS purges it immediately after every batch job ends.

/nfs/tmp*n* are alternative working directories (currently /tmp0 and /tmp1) with properties similar to /var/tmp (keyword: linux-properties (page 8)). Access is slower than for /var/tmp, but each /nfs/tmp*n* directory is global, that is, shared among (NFS-mounted on) all open Livermore Computing production machines to eliminate the need for between-machine file transfers and to simplify batch-job preparations. You should create your own subdirectory here for your computational work (rather than run big jobs in your home directory). Whenever this directory becomes 70% full, files more that 10 days old are purged by age until it becomes no more than 50% full.

/g is a file system of globally available ("common") home directories on highly reliable RAID disks NFS-mounted on each open production machine. Your child subdirectory here (/g/g*nn*/*yourname*) is your default arrival directory and contains your startup and run-control dot files, but it is limited in size. See the next section (keyword: linux-properties (page 8)) for a comparison of home and /var/tmp properties, and see the section on common home directories (keyword: common-home (page 22)) for more details on the default contents of this directory.

/opt contains the Intel compilers (in subdirectories under /intel).

/usr/apps is a link to /usr/gapps/$SYS_TYPE (beginning in May, 2001). See /usr/gapps below.

/usr/gapps is a file system of globally available ("common") code-management directories on RAID disks NFS-mounted on each machine, just as with the /g common home directories. The /usr/gapps directories contain some important *non*commercial shared application codes and tools, such as ALE3D, BASIS, HYDRA, and PYTHON, segregated into subdirectories by operating system. See the Common Home Reference Manual (URL: http://www.llnl.gov/LCdocs/chome) for details on the complex internal structure of /usr/gapps.

/usr/bin contains hundreds of standard UNIX software tools (or their Linux counterparts, such as VIM) along with the C and Fortran compilers.

/usr/local contains some important *commercial* shared tools, such as TOTALVIEW, VALGRIND, MPIP, and the PATHSCALE compilers. Some tools may not be available on Linux machines; see the Linux Differences (URL: http://www.llnl.gov/LCdocs/linux) guide for an availability list.

/p is a set of *parallel* file systems for Linux (called Lustre), specifically designed to accommodate the need for fast parallel I/O and to elminiate parallel I/O to the more vulnerable /g (home-directory) file system. See the Lustre (page 10) subsection below for details.

Directory Properties and Limits

Because of their different roles, the home (/g) and work (/var/tmp, /nfs/tmp0) directories on the open Linux/CHAOS clusters have different properties, and understanding these differences can help you use each directory in the most appropriate way (some table entries have important explanatory notes that appear below the table):

	/g/gnn/uname	**/var/tmp**	**/nfs/tmp0,1**
Role:	home directory	working directory	working directory
Aliases:	~ $HOME	/usr/tmp (*)	[none]
Status:	NFS mounted, so shared	local to each machine	NFS mounted(#), so shared
Shared across machines?	yes	no	yes
Quotas?	yes	no	yes
...On file size:	16 Gbyte/user	120 Gbyte/machine	/nfs/tmp0: 400 Gbyte/user, 63 Tbyte total /nfs/tmp1: 100 Gbyte/user, 15 Tbyte total
...On file count(+):	no limit	no limit	1,000,000/user
Purge?	no	yes($)	yes, starts when 70% full
Files vulnerable:	never	after 3 days($)	after 10 days (5 days if usage is heavy)
Automatic backup?	yes, 4 copies, every half day	no, use storage	no, use storage

NOTES and COMMENTS on TABLE ENTRIES:

(*)WARNING: on the open Linux machines, the environment variable WRK evaluates to /g/*ggroup*/*yourname* rather than to /var/tmp, that is, to your home rather than to your work directory. So do NOT rely on the command cd $WRK to move to your work directory (for example, in batch scripts from elsewhere).

(#)WARNING: the /nfs/tmp*n* directories under AIX (only) have a 16-group file-access limit. See the "Using Groups" section (page 58) below for a full explanation.

(+)The limit here is strictly speaking on *inodes.* An inode is the "index node" by which UNIX file systems keep track of the (often scattered) disk blocks that comprise each file. Since inode size is fixed, a large file may require more than one inode to list all of its disk blocks. Hence, users with large files may find that slightly less than 1,000,000 files are allowed.

($)Some LC Linux/CHAOS clusters (such as Atlas, Zeus, Rhea, Hopi, and Yana) have compute nodes with no local hard disks. On these diskless nodes /tmp and /var/tmp use real memory rather than disk space.

So CHAOS quickly reclaims this memory as soon as you delete files from /tmp or /var/tmp. CHAOS also purges /tmp and /var/tmp *completely* immediately after each batch job that runs on diskless nodes. For job files here to endure at all, you must move them to /nfs/tmp*n*, to Lustre, or to archival storage before the job ends.

A later section (page 12) shows the same comparative table for SCF machines. See the Common Home section (page 22) below for a way that your application code can test for the presence of globally mounted directories on any LC machine where it runs. See the next section (page 10) for the role and properties of the Linux/CHAOS *parallel* file systems (Lustre).

Lustre Properties

Features.

On each LC Linux/CHAOS cluster that has a high-speed network "switch" (such as ATLAS) but *not* always on those without a switch (like ACE), a Lustre *parallel* file system meets the same needs as GPFS (page 14) does on IBM/AIX clusters. Lustre is mounted on every node within a Linux cluster. Starting in 2007, LC is experimenting with mounting Lustre file systems across multiple clusters as well (see below (page 20) for name changes related to this).

Lustre addresses the need for large-file parallel I/O by providing:

- the same directory naming scheme as GPFS (/p..., but see below (page 20) for naming details),
- the same portability aliases as GPFS (/p/glocal1, etc.),
- the same huge capacity (e.g., /p/lscratcha has 172 Tbyte available),
- the same general service constraints (no backup of files, no quotas, and a purge (but see the details below) of other large file systems),
- the same performance and access trade-off issues. As with GPFS, Lustre is specifically designed to support parallel I/O with automatic load balancing across disks. If you try extensive parallel I/O on a standard, globally mounted file system such as /nfs/tmp*n* or your /g home directory, you can seriously degrade performance not just for yourself, but for all users across all the machines where that file system is mounted.

For technical information on how LC implements its Lustre file systems (including the LLNL Lustre design strategy and current parallel file-system details grouped by Linux cluster served), as well as for advice on how to handle Lustre's known pitfalls and limitations, consult the Lustre sections in the online I/O Guide for LC (URL: http://www.llnl.gov/LCdocs/ioguide/index.jsp?show=s7). Parallel file systems often interact strongly with the MPI-IO performance of application codes, a concern also discussed separately in LC's I/O Guide. To minimize such problems, SLURM now flushes the Lustre page cache after every job ends.

Purge.

Starting in August, 2006, LC uses the policy below to purge all Lustre (Linux) parallel file systems (open and secure). Note the contrasts with GPFS (page 14) and see also the purge-comparison chart (page 30) elsewhere:

- Threshold:
  LC purges Lustre file systems on an on-going, as-needed basis, *without* promising that any specific usage level must be reached first to trigger a purge (this is different from both the GPFS and /tmp purges, which involve a prespecified usage threshold).
- Scope:
  All Lustre files not *accessed* for at least 60 days eligible to be purged at any time *regardless of size* (for GPFS the time scope is 90 days).
- Schedule:
  LC purges Lustre as soon as needed to maintain efficiency, *not* on a monthly or other periodic schedule (GPFS purges occur only on the third Tuesday of each month).

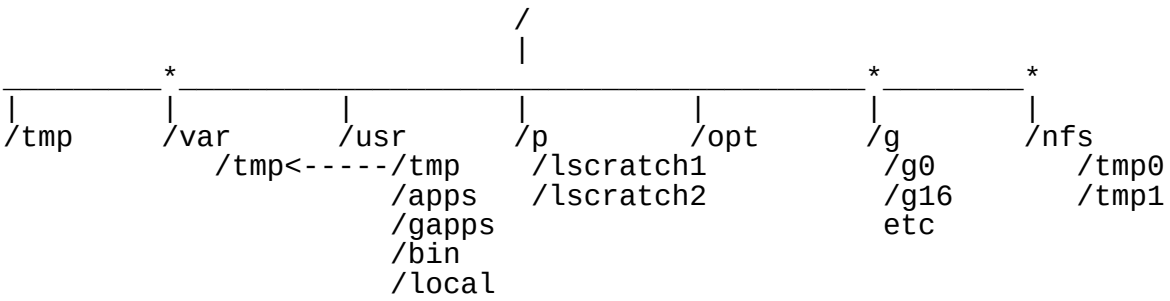
Linux/CHAOS Clusters (SCF)

On the secure network (SCF), LC offers several classified production Linux/CHAOS clusters, all ASC sponsored. As on OCF, differences in machine size, chip set, and available switch (interconnect) make different clusters suitable for different kinds of application. See LC's feature table of current SCF Resources (URL: http://www.llnl.gov/computing/hpc/resources/SCF_resources.html) for comparative technical details on Linux classified computers.

These machines have mostly the same file systems and public-directory properties as the open Linux/CHAOS clusters described above (page 6). This section therefore points out only the important differences you will encounter on the SCF machines, and refers you to the open Linux/CHAOS section above for relevant general background on file systems and directory properties.

SCF File-System Hierarchy

Each secure (classified) Linux cluster at LC has a public directory structure of the form shown here (key directories are marked *):



which is identical to the open Linux directory structure (page 6) EXCEPT that:

(1) the SCF parallel file systems for large-file, efficient parallel I/O use numerical suffixes (lscratch2, etc.) rather than the alphabetical ones (lscratchb, etc.) used on OCF, and

(2) your secure common home (page 22) directory involves /g and its children if your SCF account was first created after October 1, 1999 (/g here behaves just like /g on open machines). But if your SCF account was first created BEFORE October 1, 1999, then your default home directory was once a child of /u and was migrated to /g late in 2001. Where the /u files landed depends on whether you also had a /g directory: those already with a /g retained their old files in /g/g*nn*/*yourname*/forest; those without /g retained their files in a new /g itself. To avoid file-name collisions, these /forest subdirectories of /g persist (until you personally empty them), even though the SCF Forest cluster of machines for which they were named itself disappeared from SCF in October, 2002. See the Common Home Reference Manual (URL: http://www.llnl.gov/LCdocs/chome) for full migration details.

SCF Directory Properties and Limits

The home and work (/var/tmp) directories on the secure LC clusters differ just as they do on the open LC machines, with a few important exceptions marked (+) in this chart:

	/g/gnn/uname	**/var/tmp**	**/nfs/tmp0, /tmp1**
Role:	home directory	working directory	working directory
Aliases:	~ $HOME	/usr/tmp (*)	
Status:	NFS mounted	local to each machine	NFS mounted(#)
Shared across machines?	yes	no	yes
Quotas?	yes	no	yes
...On file size:	16 Gbyte total	56 Gbyte/node(+)	nfs/tmp0: 400 Gbyte/user /nfs/tmp1: 100 Gbyte/user
...On file count($):	no limit	no limit	1,000,000/user
Purge?	no	yes	yes, starts when 70% full
Files vulnerable:	never	after 10 days(+)	after 10 days (5 days if usage is heavy)
Backup?	yes, 4 copies, every half day	no, use storage	no, use storage

(+)Indicates an important DIFFERENCE between SCF Linux directories and the corresponding open directory properties.
(*)WARNING: on the SCF Linux/CHAOS machines, the environment variable WRK is not defined. So do NOT rely on the command cd $WRK to move to your work directory (for example, in batch scripts from elsewhere).
(#)WARNING: the /nfs/tmp*n* directories under AIX (only) have a 16-group file-access limit. See the "Using Groups" section (page 58) below for a full explanation.
($)The limit here is strictly speaking on *inodes*. An inode is the "index node" by which UNIX file systems keep track of the (often scattered) disk blocks that comprise each file. Since inode size is fixed, a large file may require more than one inode to list all of its disk blocks. Hence, users with large files may find that slightly less than 1,000,000 files are allowed.

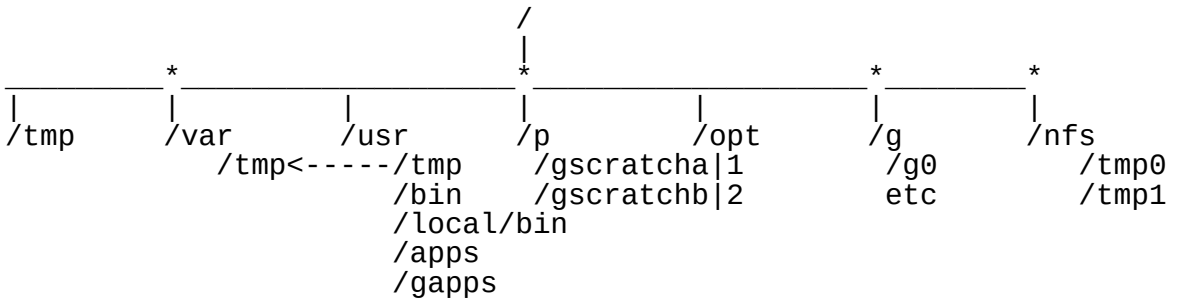
An earlier section (page 8) shows the same comparative table for open-network (OCF) machines. See the Common Home section (page 22) below for a way that your application code can test for the presence of globally mounted directories on any LC machine where it runs. See also the section (page 10) above on Lustre, the Linux parallel file system whose properties (but not directory names) are the same on SCF and OCF.

IBM AIX

LC's IBM machines are massively parallel sets of standard IBM processors configured for ASC production computing using the AIX operating system. Some have used IBM POWER3 chips, some use POWER4 chips, and some use POWER5 chips. For current configuration information on these IBM/AIX clusters, consult LC's tables of open (URL: http://www.llnl.gov/computing/hpc/resources/OCF_resources.html) and secure (URL: http://www.llnl.gov/computing/hpc/resources/SCF_resources.html) ASC-sponsored computing resources. (Note that Lilac, although an IBM machine, actually runs CHAOS/Linux (page 6) rather than AIX.)

IBM/AIX Directories

Each massively parallel IBM/AIX machine has a public directory structure of the form shown here (key directories are marked *):



This structure offers the same directories serving just the same roles (and quotas) as on the open Linux/CHAOS (page 6) clusters (described above), with only these differences:

/tmp	is a small directory for system use only (very transitory files).
/opt	contains IBM specialty tools (diagnostics, Tivoli, some freeware).
/p/g...	are General Parallel File Systems (GPFS) that fill the same role here as Lustre (page 10) parallel file systems fill on Linux/CHAOS machines. Note the /g rather than /l prefix to 'scratch' in the names (with alphabetical suffix for OCF, numerical for SCF as with Lustre). See the next section (page 14) for details. See Parallel File Systems (page 20) below for recent LC naming and mounting policies regarding GPFS.
/nfs/tmp*n*	has a 16-group file-access limit under AIX that is not seen under Linux/CHAOS. See the "Using Groups" section (page 58) below for a full explanation.

GPFS Properties

Each IBM machine has one or two "General Parallel File Systems" (GPFS) intended specifically for large-file parallel I/O. [See the section below (page 20) for important name and mount changes now phasing in regarding GPFS. Formerly, each GPFS name had the form /p/gx1 or /p/gx2, where *x* was a letter (or pair) that designated the host computer (so /p/gup1 was on UP, while /p/gum1 was on the UM cluster).]

On the IBMs you should continue to use the NFS-mounted directories (such as your common home directory (page 22) and /nfs/tmp*n*) for source files, executables, and other relatively small files (for better service and more efficient resource utilization). The GPFS directories have slow metadata operations and a large block size, so they are designed primarily for handling large data files that need high-speed parallel I/O access from all compute nodes. In fact, attempting parallel I/O to a standard, globally mounted file system such as /nfs/tmp*n* can easily degrade I/O performance for all users across all the machines that share that file system. See the I/O Guide for LC (URL: http://www.llnl.gov/LCdocs/ioguide) for locally relevant GPFS technical information and usage advice (for example, see the feature-by-feature comparison table (URL: http://www.llnl.gov/LCdocs/ioguide/index.jsp?show=s7.2) in the section called "Lustre and GPFS Compared"). To enable script portability when using GPFS, an alias for (symbolic link to) /p/gx1 called /p/glocal1 (etc.) is provided on each IBM/AIX machine.

This table summarizes the properties and limits of the (pair of) GPFS file systems available on each IBM/AIX machine (for easy comparison with the other directories similarly tabulated above (page 6)).

	GPFS (/p/gscratcha|1, etc.)
Role:	fast parallel I/O
Aliases:	/p/glocal1, 2
Status:	not NFS mounted, IBM AIX hosts only
Shared across nodes?	yes
Quotas?	no
On file size:	18.5 Tbyte total (OCF), 63 Tbyte total (SCF)
On file count:	no limit
Purge?	yes, summarized below(+)
Files vulnerable:	voluntary cleanup
Automatic backup?	no, use storage

(+)Using GPFS effectively and appropriately requires storing your files so that you avoid needlessly clogging the file system, and especially so that you avoid losing valuable data to the GPFS file purge (see the next section or the purge chart (page 30) later for comparisons with the very different Lustre purge policy).

- Threshold:
  LC purges GPFS when and only when usage exceeds 80%.
  Purging of files continues (oldest files first) until the file system is no more than 70% full (see schedule below).

- Scope:
  All GPFS files at least 13 weeks old (= 3 months) are eligible to be purged if the purge threshold is reached, but...
  All GPFS files 100 Kbyte or less in size are exempt from the purge regardless of their age.
- Schedule:
  If usage reaches the purge threshold during any month, then LC will start purging eligible GPFS files on the third Tuesday of that month (and continue until usage sinks to 70%).
  On the first Tuesday of every month, pre-purge logs are available for each user in a personal directory called

  `/p/g`*`xx`*`/purgelogs/`*`username`*

  (to help you anticipate which files are vulnerable for purge that month).

Former Compaq Cluters (Obsolete)

LC retired its last Compaq (once DEC) machines in January, 2007, so file-system and directory-name information for them is now obsolete. This section exists merely to gracefully preserve previously made inward links to that information. See instead Linux/CHAOS Clusters (Open) (page 6) and Linux/CHAOS Clusters (SCF) (page 11) for basic, current file-system and directory-name details for LC machines.

DFS and NFS Compared

LC provides its common home directories /g and its shared temporary disk space (/nfs/tmp0 and /nfs/tmp1 on both OCF and SCF) by using the Network File System (NFS) as implemented by a vendor called Network Appliance. Separately, and for more specialized purposes, LC provides other shared disk space by using a pilot implementation of the Distributed File System (DFS). Thus although they have similarities, at LC NFS and DFS are parallel projects with different detailed features deployed to meet different needs.

NFS (Network File System)

NFS (Network File System), in various implementations, is a widely used service that allows disks (file systems) to appear local to multiple machines so that overtly copying or moving files between those machines (such as with FTP) is no longer necessary. LC uses NFS to support the common home (page 22) (/g) and shared temporary (/nfs/tmp*n*) directories that appear local to all its open computers (separate NFS installations provide similar service for several separate clusters of secure machines as well). In 2004, LC significantly upgraded its open NFS capacity to increase the reliablity of its shared file systems and to expand the available shared disk space (SCF) to 77.2 TByte (divided by quota (page 44) among users).

I/O LOAD.
One drawback to mounting NFS file systems globally across all production machines involves performance under heavy I/O load. If just one user attempts massive parallel I/O to /nfs/tmp*n* or to their common home directory (instead of using GPFS or Lustre, local to each cluster and designed for that role), then all NFS users on all machines can experience seriously degraded performance. So plan your parallel I/O for only parallel file systems designed to support it.

SECURITY ISSUES.
Also, NFS has the potential for some security weaknesses that are of concern if its use is not limited to a LAN environment where overall network and system security are well managed. Most NFS compromises are the result of configuration errors. LC is cognizant of the various types of configuration errors and is extremely careful regarding the configuration of the various NFS servers. Configurations are reviewed by senior technical staff to ensure their correctness.

The foremost intrinsic NFS vulnerability relates to an object called a file handle. If discovered or otherwise captured, a file handle for a particular file system allows anyone from any platform to easily access any data on that file system. The challenge is to capture a file handle. Three known methods to do this are discussed below:

- Stealing. In the past, various versions of UNIX operating systems allowed file handles to be stolen either from kernel memory or from tricking system daemons such as portmap to disclose it. LC has attempted to steal file handles using well-known "hacker" tools from its current production systems and has been unable to do so. The server operating system (NetApp Data ONTAP) appears to restrict access to file handles appropriately. In addition, the client operating systems in use on LC systems (e.g., CHAOS, AIX, IRIX, and Solaris) appear to properly safeguard file handles.

- Snooping. Snooping a file handle from the network generally requires the compromise of a system (i.e., gaining root privilege). By policy, access to file systems from LC NFS servers are allowed only to LC managed and root-controlled systems. Further, snooping in general (e.g., from a PC) is not at all easy since LC makes extensive use of switch-based networks. NFS traffic to and from the servers from the production hosts are through private Fast Ethernet (switched) networks or through switched FDDI links. Snooping a file handle is extremely unlikely.
- Masquerading. Masquerading involves one system taking over the identity (i.e., IP address) of another system, typically when the other system is turned off. LC systems are operated on a 24-hour basis with a 24-hour operations staff; the possibility of masquerading is extremely remote, and NFS traffic is blocked from entering or leaving the OCF network by router filters.

In summary, as long as NFS traffic is kept within a controlled LAN environment and the servers are properly configured, it is unlikely NFS can be compromised. There is probably a greater likelihood of compromising root on a given system, which LC does all it can to guard against.

DFS (Distributed File System)

To provide file sharing in an "enterprise" or WAN environment where security domains and administrative control areas are crossed, LC has been working on implementing DFS (Distributed File Service). Whereas NFS (page 17) is currently not considered a safe technology for sharing files outside a LAN environment, DFS is safe since it relies on the Distributed Computing Environment (DCE) security component to authenticate a user. DFS provides a uniform global name space and supports fine grain access control to files and directories using Access Control Lists (ACL).

SCOPE.
The LC DFS pilot environment currently has a disk capacity of 250 GB. DFS is *not* available on any LC Linux system. And IBM stopped supporting DFS with the release of AIX version 5.3 (which spread among LC production IBM systems during 2006). Hence, on LC networks, DFS is only accessible from the few remaining AIX 5.2 IBM systems or other special-purpose machines. DFS is a mature technology but was never broadly adopted, so its potential benefits were largely unrealized in the LC environment. In 2006, HPSS abandoned the DCE support required for DFS (with HPSS version 6.2). The PSUB utility to submit batch jobs has been revised to get the authentication needed to let batch jobs access DFS files.

ROLE.
LC's initial use for DFS was targeted at projects or groups interested in having a shared file space extended to their desktop or shared between the DOE laboratories. Extension to UNIX and Windows NT desktops has already been accomplished for members of an atmospheric sciences research group at LLNL. Another targeted use is to address concerns for protection of export controlled software and the protection of intellectual property. These projects use DFS for establishing the proper modes of access, by granting full access permissions for application developers, limiting access of application users, and denying access to others. LC has also been working on integrating Web access with DFS; this brings the security features of DFS to the Web and the Web data is location independent.

Begining in February, 2001, LC uses the standard DFS file-permissions mechanism (the "access control list" or ACL) to disable access to TOP-level DFS project directories for some users as part of its security

policy. For each file and directory in DFS, an access control list specifies for the following sets of users (called "ACL entries")

```
user_obj          the owner
group_obj         the group assigned to this file
other_obj         any user in the local DCE cell
foreign_other     any user from a DCE cell not locally
                  administered (e.g., Sandia and LANL
                  are "foreign" to LLNL's cell)
any_other         any user in the world
```

the access permissions (read, write, etc.) that those users have. Within DFS this ACL approach overrides the usual UNIX permissions.

DFS project directories have names of the form /dfs/proj/*codename*. The access restrictions on these directories now work as follows:
(1) The /dfs/proj directory now has no foreign_other entry at all in its ACL.
(2) Each of the 21 individual /dfs/proj/*codename* directories (such as /dfs/proj/ale3d) now has no foreign_other and no any_other entry in its ACL.
(3) The other_obj entry remains in the ACL for each /dfs/proj/*codename* directory, but no permissions (000) are assigned to it.

Permissions remain unchanged for all files and directories below the /dfs/proj/*codename* level, but if your access to them depends on your membership in the foreign_other or any_other sets of users, then your ability to read (or even to list) these low-level files may be disabled. (The former acl_edit utility is no longer available on LC production machines; it was seldom used.)

For current technical details on the tools and techniques needed to manage files under DFS at LC, consult the draft local documentation available online at this URL (OTP password required):

```
Open: https://lc.llnl.gov/computing/docs/dfsintro4.html
SCF:  [not yet available]
```

Parallel File Systems

Parallel file systems are specially designed to efficiently support large-file parallel I/O from every compute node in a cluster. At LC, the installed parallel file systems are Lustre (page 10) for Linux/CHAOS machines and GPFS (page 14) for IBM/AIX machines. LC is currently changing its approach to parallel file systems to introduce:

- new file-system *names* that generalize more easily and that emphasize the temporary nature of data placed on these devices, and
- the mounting of (some) parallel file systems *across multiple clusters* for greater convenience with less need to move files between like machines.

Eventually, this section will offer a unified summary of the parallel file system features (as changed by the two adjustments mentioned above, if they succeed) that are now discussed separately for IBM (AIX) and Linux (CHAOS) clusters for historical reasons. During this (probably year-long) transition, however, this section explains the in-coming naming scheme for parallel file systems.

Old Names:
Under the old naming scheme used through 2006, each LC parallel file system had a name of the form

/p/g*abbrnum*

where *abbr* was a one- or two-letter abbreviation for the machine on which the file system was mounted (e.g., b for BlueGene/L, um for UM) and *num* was the digit 1 or 2. For example, /p/gum1 was the parallel file system on UM.

New Names:
Under the new naming scheme phased in starting in mid 2006, each LC parallel file system gets a name of the form

/p/[l|g]scratch[*ocfletter*|*scfnumber*]

where

l	(literal lowercase el) indicates a Lustre (Linux/CHAOS) file system,
g	(literal) indicates a General Parallel File System or GPFS (IBM/AIX) file system,
ocfletter	is a unique one-letter indentifier for OCF systems (a, b, c, etc.), and
scfnumber	is a unique one-digit indentifier for SCF systems (1, 2, 3, etc.).

Example:

	OCF (Thunder)	SCF (Purple)
Old name	/p/gt2	/p/gp1
New name	/p/lscratchb	/p/gscratch1
Alias	/p/glocal2	/p/glocal1

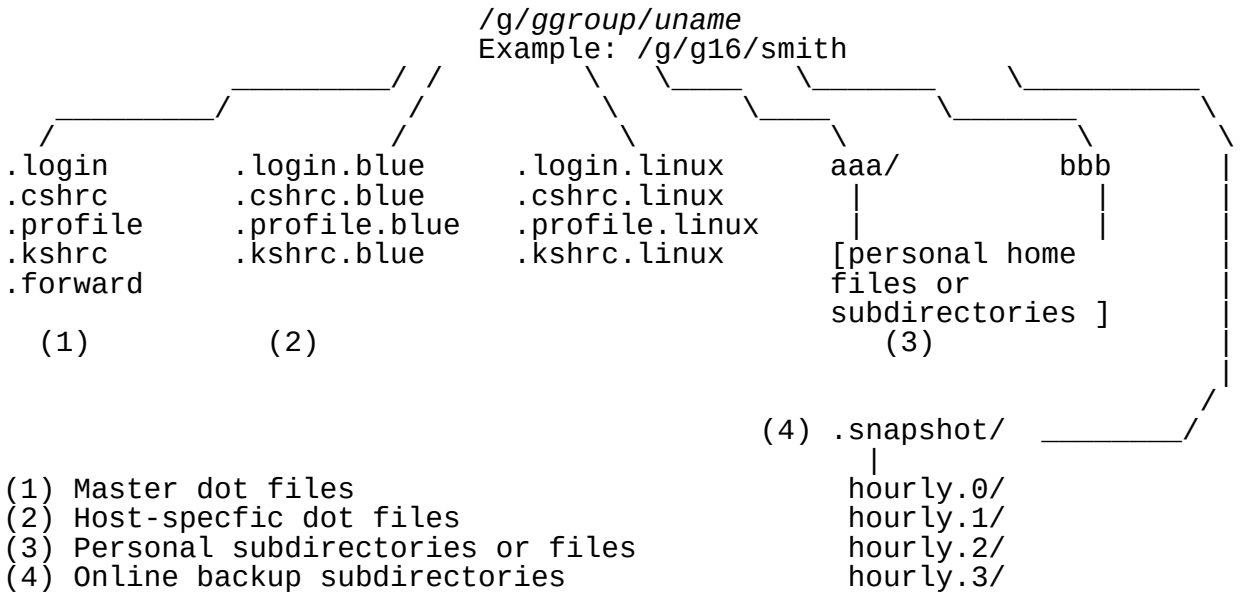
Note that the script-stabilizing alias names continue to follow the old, not the new, scheme (on the use of g and of numbers for OCF).

Warning:
To discover which naming scheme, old or new, is currently in place on any particular machine where you plan to run jobs, log on to that machine and try to CD into a directory with each alternative name. Note that just before each name change, the old directory remains for a few weeks in *read-only* status, so also check that you can actually create files once you are in any parallel file system.

Common Home Directories Summarized

All production nodes on all open network LC clusters (both Linux/CHAOS and IBM/AIX) share home directories that reside on global file system /g, NFS mounted from several dedicated servers. Likewise, all secure-network (SCF) production clusters share their own secure home directories on a global SCF /g file system. This too is NFS mounted across machines from multiple servers. (Legacy SCF users whose accounts were first created before October 1, 1999, have had their SCF home directory moved from the /u to the /g file system, as explained in the Common Home Reference Manual. (URL: http://www.llnl.gov/LCdocs/chome))

This "common home" arrangement makes keeping redundant home files (and doing redundant updates) on multiple machines unnecessary, and it allows the same pathname and directory structure for home to be shared on every host involved:



where

ggroup is an administrative subdirectory different from the UNIX groups that LS or GROUPS reports. On OCF, it is g0 for LC staff and g2 or above for other users; on SCF, it is g5 for LC staff and g10 or above for other users, and where

uname is your alphabetic login account name (not the numeral *uid* by which your file and block quotas are reported).

(1) Master dot files

are your basic startup and run-control files (copied from /gadmin/etc/skell). They detect the machine type you are on (and the operating system you are under) by evaluating the HOST_GRP and SYS_TYPE environment variables, and they then invoke the appropriate machine- (or system-)specific dot files (shown here at (2)).

Only customizations intended for ALL machine types and operating systems should go into these master dot files.

(2) Host-specific dot files

are the place for your customizations that apply only to individual hosts or to specific operating systems. "Host-group" names as suffixes (e.g., BLUE for all the open AIX machines, LINUX for all open CHAOS nodes) simplify sharing these customizations among like machines (only).

(3) Personal files and subdirectories

are whatever other files you want to have in your home directory, up to your quota, organized as you wish under /g/ggroup/uname. These files will appear to reside on every machine that shares the common home directory.

(4) Online backup subdirectories

contain four complete (but read-only) backup copies of every file and subdirectory in your common home directory, made automatically at noon and 7 p.m. every day. The most recent backup resides in .snapshot/hourly.0, with each earlier backup in a correspondingly numbered hourly.*n* directory. See the Backup section in the Common Home Reference Manual (URL: http://www.llnl.gov/LCdocs/chome) for technical details.

CAUTIONS.

Several cautions apply to the common home directories:

First, common home directories all have disk-space quotas: 16 Gbyte for general users (8 Gbyte for LC staff). Note also that the output from quota-monitoring tools varies somewhat from one brand of computer to another. Consult the QUOTA (page 44) section below for a comparison of disk-usage reporting formats.

Second, your common home directory by default allows access only to you (as the owner). You can widen this access (by running CHMOD, whose instructions appear in a later section, keyword: chmod (page 34)). But remember that the changed permissions will apply to ALL hosts sharing this directory, not just to the one host on which you make the change, as would usually be the case. Because of the danger of lost data or hidden changes to your login files, WORLD write access to your top-level common home directory (i.e., to /g/ggroup/uname) is NOT allowed on open machines. And world or even group write access to /g/group/uname disables SSH as well. You can enable world or group write access to your home subdirectories if you wish, but this too is risky. Consider using GIVE (page 39) or TAKE (page 42) instead.

Third, you cannot checkpoint (for restart) any job whose files reside on NFS-mounted disks. Since your common home directory is NFS-mounted, any batch job you run on a common-home machine that spawns a shell will access its dot files on those disks, and hence, that job will not checkpoint.

Fourth, your common home directory resides on an NFS file system *not* designed to handle high-volume parallel I/O. If just one user attempts massive parallel I/O to their common home directory (instead of using GPFS (page 14) or Lustre (page 10), mounted on each cluster and designed for that role), then all

NFS users on all machines can experience seriously degraded performance. So plan your parallel I/O for only parallel file systems designed to support it.

For additional technical details on the behavior of common home directories (and their quotas), on host-group names, and for advice if problems arise with specific software, consult the Common Home Reference Manual (URL: http://www.llnl.gov/LCdocs/chome) on the LC documentation web servers.

TEST.
If you have a single version of an application code (or utility) that you run on diverse machines, you may need to test for the presence or absence of LC common home directories to decide about where to locate (initialization or output) code-related files. To facilitate this test, each LC machine offers a system file called /etc/home.config, in which one line contains the keyword "FILE_SYS" (uppercase) followed by one of four configuration values (shown below, mixed case). Your code can reveal the content of this line, for use in conditional tests, by executing

```
grep FILE_SYS /etc/home.config
```

The four possible configuration values that this might return are:

global	means that all LC global file systems, including the common home directories of *all* users, are available.
LC_only	means that only LC staff common home directories are available; home directories for general users are *not* mounted here (e.g., LUCY).
green	means that only unrestricted-network (local) home directories are available (e.g., for "foreign national" users).
pl1	means that this SCF system at security level PL1 has *no* global (including common home) file systems available.

File Management Techniques

Typical Code-Development Scenario

This scenario suggests practical and appropriate steps for managing your files while you develop code on any open LC cluster. The same basic approach also applies while working on the nodes of the secure LC clusters:

(1) Log on. You will arrive in your home directory, which on the open clusters is the "common home" directory /g/ggroup/uname that all such machines share. Because it is not purged, your home directory is a good place for various background and support files. But because of its size quotas, developing large projects in your home directory is often impractical. Never perform massive parallel I/O in your home directory because this degrades home-directory service for *all* users on *all* machines (use a parallel file system instead, such as GPFS (page 14) or Lustre (page 10).

(2) Use CD to move to /var/tmp (or to /nfs/tmp0 or /nfs/tmp1 for more space). This puts /var/tmp, which is NOT on your default search path, into your search path as the "current working directory" (.), a convenience for executing your own programs. Create a subdirectory (usually named after your logon id for clarity, such as /var/tmp/jsmith), and CD into it. This makes the relatively large size of /var/tmp (or /nfs/tmp*n*) available for your work without mingling your files with others.

(3) Create symbolic links from pointers in your subdirectory of /var/tmp (or /nfs/tmp*n*) to your code (and perhaps to other key files) in your home directory, in this way:

```
ln -s origfile pointername
ln -s ~/proj2 proj2.tmp
```

These links must be symbolic (-s) because your home directory, /var/tmp, and /nfs/tmp*n* do not reside on the same file system.

(4) Edit, compile, test, and debug your code in your subdirectory of /var/tmp or /nfs/tmp*n*. Because of the symbolic links, your changes and updates end up mapped safely back to your home directory as a side effect of your work in /var/tmp. And because /var/tmp has no quotas, you enjoy much greater flexibility in using large data sets or making large output files. This approach also mitigates the occasional slowness of working on mounted file systems directly.

(5) Store your results and updates often. Use FTP (or, where available, NFT) and transfer copies to storage.llnl.gov (open or secure). Remember that /var/tmp (and /nfs/tmp*n*) on Livermore Computing production machines are NOT automatically backed up, and that all files older than a threshold age (which can be as short as 3 days during heavy use) can be purged with no possibility of recovery. With regular archival storage and good housekeeping you can delete unneeded files and leave more space for other users.

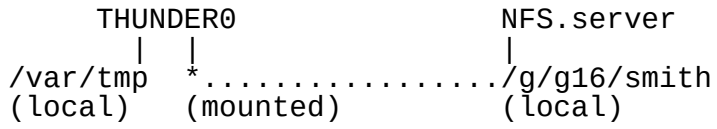
(6) There is no need to move files in your home directory (or in /nfs/tmp*n*) on Livermore Computing production machines between MACHINES because your home directory is shared among all open machines (on the open network) or among all secure machines (on the secure network). The other working directories (such as /var/tmp) are local to each machine, however, and you should use FTP (or, where available SCP

or NFT) to move work-directory files from /var/tmp on one machine to /var/tmp on another (if you need to).

(7) On IBM/AIX machines only, the batch system automatically sets the environment variable LOADL_STEP_ID and then creates a job subdirectory called /var/tmp/$LOADL_STEP_ID. If you run your batch job in that specific directory, then all of its files will automatically disappear when the job ends (so save them first!), guaranteeing that you will not hoard /var/tmp space on the IBM/AIX compute nodes. The LOADL_STEP_ID variable and directory is *not* available on IBM machines that use SLURM instead of LoadLeveler as their underlying batch system, however.

Transfer Rates for Mounted File Systems

To make files more readily available across machines, a file system local to one machine is sometimes "NFS-mounted" on (or "exported to") another machine. For example:



Here the home directory /g/g16/smith is really local to some NFS server machine, but it has been mounted on THUNDER0 (a Linux cluster node) and so it appears there as well, where you can use ordinary UNIX tools (LS, RM, CHMOD, etc.) to manipulate its files as if they were local to THUNDER0.

When TRANSFERRING large files between a mounted and local file system, however (for instance, between /g/g16/smith and /var/tmp on THUNDER0 here), efficiency issues can affect the best choice of transfer tool to use. Copying the files with CP is the most natural and simple approach, and is fine for small files (behind the scenes a remote copy is really taking place between machines). But for large or multiple files, using FTP for the transfer instead of CP, even though only one host seems to be involved, proves much more efficient.

To transfer a mounted file both FTP and CP must open the source file, read the file's bits, open a destination file, and write its bits. But the transmission rate for CP is severely limited by the NFS mounting mechanism, while FTP efficiently bundles the transmitted bits to take (better) advantage of the maximum bandwidth available. In the example above involving a file system cross mounted on two machines at LC, the typical (between-machine) transmission rate for CP is only 0.5 Mbyte/s but the typical rate for FTP is almost 5 Mbyte/s, ten times faster. So this significant difference justifies the extra setup that FTP requires if many or large files are being transferred. (Remember too that this difference applies ONLY to mounted file systems, and that CP has no disadvantage in speed and an obvious benefit in simplicity for ordinary file transfers done WITHIN a single file system.)

File-System Usage Comparison

On LC production machines, different file systems (appearing as different top-level directories) have different features and hence different intended uses. (Globally mounted standard file systems, such as /nfs..., promote easier *access* to files, while within-cluster parallel file systems, such as /p..., promote *parallel I/O* without degraded performance.) This chart summarizes the most suitable use for each LC file system, to help you compute efficiently and avoid problems. Each itemized file system is also linked to its detailed technical description elsewhere in EZFILES (or in another manual).

File System:	**Most Suitable For:**
Home directory	
/g/gnn/uname	Small, permanent files shared among machines (dot files, source code)
Local temporary directories	
/tmp	Only for system use (under Linux, links to /var/tmp)
/usr/tmp	A link to /var/tmp
/var/tmp	Temporary(*) storage of input or output local to each machine
Global temporary directories	
/nfs/tmp1	Same role as /nfs/tmp0
/nfs/tmp0	Large temporary storage for input or output shared among many machines
Parallel file systems (GPFS, Lustre)	
/p/gscratch... /p/lscratch...	Large data files, fast parallel I/O shared among cluster nodes (but not across clusters)
Storage directories	
[ftp storage]	Long-term (archival) storage of any size or type of file

(*)On LC's Linux/CHAOS clusters with diskless compute nodes (such as Atlas), /var/tmp uses real memory so CHAOS purges it completely immediately after every batch job ends.

File-Sharing Alternatives Compared

The standard UNIX technique for sharing access to files among several users is to enable (read, write, or execute) permissions (page 32) on those files (and their directory trees) for "world" or "other" users. On LC production machines, however, all top-level world permissions are automatically disabled (page 37) (set to 0) by monitoring software as a security policy. This effectively prohibits world-permission file sharing at LC. (An exemption requires specific approval of your LLNL Associate Director; contact the LC Hotline for details.)

Here are several alternative file-sharing techniques, each with its own strengths and weaknesses (and each linked to its execution details elsewhere in EZFILES):

- Run utilities GIVE (page 39) (to copy a file to another user) and TAKE (page 42) (to receive a GIVEn copy). This is well suited to sharing a few seldom-changed files with another specific user, but not to sharing large sets of files with many users, especially if the files change often. Also, the sharing users must all work on the same SCF or Linux/CHAOS machine; on OCF AIX or Tru64 machines you can GIVE files on one machine and TAKE them on another.
- Use file group (page 58) membership and permissions rather than world permissions. This is well suited to sharing large sets of files or whole subdirectories with a stable list of other users, and it allows across-machine sharing (if the files are in a globally mounted file system, such as the common home directories, and if the same user group exists on several machines). But an administratively approved form is required to create a group, and no LC user can belong to more than 32 groups at once.

  One variation on file sharing by group involves enabling group permissions on file(s) parked in /usr/gapps (common to all production machines). This requires completing the special USR_APPS form (because /usr/apps is just a link to /usr/gapps/$SYS_TYPE).

  A second variation involves enabling group permissions on STORED files. Storage groups and online groups are not the same, however, and group assignment changes when a file is stored. See the EZSTORAGE (URL: http://www.llnl.gov/LCdocs/ezstorage) basic guide for detailed instructions on sharing stored files.

- Use DFS (page 18) (Distributed File Service) with an access control list (discontinued for most purposes in May, 2007). This allows each shared file to have its own fine-grained list of sharing users. But the files must reside on a (still-developmental) DFS server, whose DFS file-management utilities are exotic and fairly complex.

Backup Policy Summary

Because some file systems contain valuable, frequently used information whose loss would be very disruptive, LC uses Legato Networker software to make backup copies of these file systems on tape. Full backup copies are made once each month, and incremental (changes-only) backups occur every night except Saturday and Sunday. LC copies over 400 Gbytes of unclassified data and over 90 Gbytes of classified data to backup tapes every month.

Other file systems, however, although heavily used for important work, are simply too large to allow practical backup copies to be made, or they are mounted on machines for which no appropriate backup software is commercially available. For example, neither the GPFS (IBM) nor Lustre (Linux) parallel file systems, with their multiple terabytes of capacity, are backed up, and there is no redundancy in their underlying "storage servers" (so that one hardware failure can make many distributed files unavailable). Also, files older than 90 days on GPFS (page 14) or 60 days on Lustre (page 10) can be (and usually are) purged, that is destroyed with no backup copies.

For files on these systems, no second copy automatically exists. Here your responsibility is to personally make storage copies of all crucial files in case you need to restore them on your own after disk problems. To easily store very large archives of very many files, consider using LC's highly efficient software tool designed for this specific task, namely HTAR (page 54).

This chart summarizes the backup status for each major file system on the LC production machines (those not listed are also NOT protected by backup):

File system	Backup status	
	Automatic backup On ALL machines	NO backup here
/g/gnn[*]	X	
/usr/local	X	
/usr/gapps[*]	X	
/usr/tmp		X
/var/tmp		X
/nfs/tmp0		X
/nfs/tmp1		X
/p/gscratcha|1 (GPFS)		X
/p/lscratcha|1 (Lustre)		X
/dfs	X	

[*]See the Backup section in the Common Home Reference Manual (URL: http://www.llnl.gov/LCdocs/chome) for how to retrieve these "layered" backups.

File Purge Policy

File systems at or near their capacity often show degraded performance, higher I/O error rates, or sometimes complete service failure. To make service more predictable and reliable, LC intentionally destroys ("purges") files on at-risk file systems intended for temporary storage (especially the large NFS-mounted termporary file systems and the GPFS and Lustre parallel file systems).

This chart summarizes and compares the current LC file-purge policies on those file systems where LC regularly purges user files (without backup (page 29)):

Purge Policy	**/nfs/tmp***	**GPFS (AIX)**	**Lustre (Linux)**
Purged?	Yes($)	Yes	Yes
Backed up?	No	No	No
Usage threshold that triggers a purge?	70%	80%	As needed
Purged down to what level?	50%	70%	As needed
Purge order?	Oldest files first	Oldest files first	Oldest files first
Eligible files:			
...Age (last accessed)?	Over 10 days(+)	Over 13 weeks = 91 days	Over 60 days
...Size?	Any size	Over 100 kbyte	Any size
Schedule:			
...Purge cycle?	Nightly (if needed)	Monthly, third Tuesday	As needed
...Prepurge inventory?	No	Yes, first Tuesday(#)	No

($)While /nfs/tmp is purged on a schedule, remember that on LC's Linux/CHAOS clusters with diskless compute nodes (such as Atlas), CHAOS purges /var/tmp *immediately* after every batch job ends.
(+)Over 5 days if usage reaches 90% since the previous day.
(#)Prepurge logs for each user are available at /p/gX1/purgelogs/*username*, where X is the relevant machine abbreviation (e.g., gum1) and *username* is your login name. Each log lists your specific files that would be purged unless you store and delete them beforehand.

Search Paths

When you try to execute a program by typing its (simple) file name, your UNIX shell searches through the file structure looking for a file with that name to execute. The order in which it searches directories is specified by your search path, a colon-delimited ordered list of directories stored in the environment variable PATH (all uppercase). If you use a program's absolute pathname, the shell ignores your search path. Search paths often differ drastically from one brand of computer to another because of operating-system differences. You can reveal your current search path on any LC machine by executing

```
echo $PATH
```

on that machine.

Permissions for Files and Directories

Kinds of Permission

All files and directories have an owner, usually the person who initially created the file or directory. The owner can assign UNIX permissions to other users, and these permissions control who can manipulate the files and directories.

There are three classes of users who may have different permissions for a file or directory:

```
u = user   (the owner)
g = group  (the owner's group)
o = others (everyone else)
```

There are three kinds of permissions (r, w, x) that may be assigned to any or all of the above classes of users. This table explains each permission and what it allows those who have it to do:

Permission	Actions allowed on files	Actions allowed on directories
R (read)	Read, copy (CAT, CP, LP, etc.)	List files in directory
W (write)	Edit, append	Create or remove files in directory
X (execute)	Run, execute	CD into directory

How to Discover Permissions

Run LS with the -l option to see the permissions assigned to your files and directories. Use the -a option (ls -la) to also see those files whose names begin with a dot (.). Here is an example of such output:

```
drw-------     1   tom     grp     1714   Mar 24 10:34  ABC
-rw-------     1   tom     grp     2966   Apr 14 16:48  filea
-rw-------     1   tom     grp    64269   Mar 13 15:51  filxyz
-rwxrw-r--     1   tom     grp   309938   Apr 10 11:19  MYFILE
 |________|
     |
 file permissions
 (the very first character is not part of the permissions)
```

The permissions assigned to the file MYFILE in this example can be interpreted as:

```
        rwx    rw-   r--    The 9-character field is divided into
         u      g     o     3 ownership classes. The 9 permissions
         |      |     |     together form the "mode" of the file.
         |      |     |
User has read,  |    Others have read
write, and      |    but not write or
execute rights  |    execute rights
                |
            Group has read,
            write, no execute
            rights
```

The default permissions for files on all LC machines are RW for the owner and no permissions for any other user (mode rw-------).

To obtain a list of all your files that have certain specific permissions, you need to use not LS but a tricky combination of options for the FIND utility. You also need to use the octal method of representing permissions, explained in the next section (page 34). As an example of this technique, the execute line

```
find . -perm -770 -follow -print
```

reports all and only the files in your current directory (.) and all of its subdirectories that have access permission 770 (user rwx, group rwx, others none).

How to Change Permissions (CHMOD)

Only the owner of a file or directory (or the super user) can change a file's permissions (mode). The changes are made with the CHMOD utility. There are two forms of CHMOD syntax: one specifies the desired permissions as an absolute (octal numeric) value; the other is symbolic and changes permissions incrementally.

```
chmod  nnn  files              (absolute form)
chmod  j operator k  files     (symbolic form)
```

(To execute CHMOD using a graphical user interface on LC production machines, see the HOPPER subsection below (page 55).)

Absolute (Octal) Form of CHMOD

In the absolute form of CHMOD a 3-digit octal number *nnn* specifies the desired permissions for a file or directory, without regard for its previous permissions:

```
chmod nnn filename(s)
```

You construct the appropriate enabling number *nnn* by ORing (adding) the digits vertically for each permission you want to allow for each class of user (u, g, o) shown in the chart below:

	u	g	o
r	4	4	4
w	2	2	2
x	1	1	1

```
For example:        6    4    4    Gives user read and write,
                                    everybody else read only

                    7    5    5    User has rwx; group and
                                    others have rx rights

                    6    0    0    User has rw rights,
                                    others have no rights
```

For instance, to assign the permissions (mode) rwxrw-rw- (766) to MYFILE, type the line

```
chmod 766 MYFILE
```

which works regardless of what the former permissions on MYFILE were.

Symbolic Form of CHMOD

The symbolic version of CHMOD uses letters and symbols to specify incremental changes to the existing permissions on a file (add a right, delete a right, change all the rights) and which user classes receive the change (user, group, others). This chart shows the pattern:

```
      COMMAND      HOW TO ALTER MODE      FILE(S) TO CHANGE
       chmod         j operator k            filename
_____________________________________________________________
```

```
   where           j is  u (user)
                         g (group)
                         o (others)
                         a (all of the above)

                   operator is  - (remove permission)
                                + (add permission)
                                = (reset/assign permissions;
                                  overrides all current per-
                                  missions)
                                  on all user classes)

                   k is  r (read)
                         w (write)
                         x (execute)
```

For example, to change the permissions of the file MYFILE from rw-rw-r-- (user rw; group rw; others r) to rwxrw-rw- (user rwx, group rw, others rw), you need to add execute permission for the user and write permission for others. So you would type:

```
chmod   u+x,o+w   MYFILE
```

More than one change may be made, but they must be separated by commas and NO spaces, as shown here. To later remove execute (x) permission for the user (with this symbolic CHMOD syntax), type:

```
chmod u-x MYFILE
```

Citizenship and Permissions (SCF)

On the secure (SCF) production computers (such as UM, UV, and Lilac), user citizenship can affect file sharing and the assignment of file permissions.

System administrators create groups of users (usually at the request of projects or collaborators) so that group members can use CHGRP and CHMOD to allow shared access to files among, but only among, everyone in the group. The GROUPS utility, run without options, reports the names of the groups to which you currently belong, just as LS reports the group to which a file has been assigned. On the SCF production machines, every user belongs to an "extra" group that reflects that user's citizenship (for example, every U.S. citizen belongs to the group us_cit). You can therefore restrict the access for any file to fellow citizens by using CHGRP to assign the file to your citizenship group, such as

```
chgrp us_cit myfile
```

and then using CHMOD to open group permissions but limit (or eliminate) world permissions on that file, such as

```
chmod 750 myfile
```

On SCF production machines, system administrators also use these citizenship groups in more subtle ways, along with the segregation of home directories by citizenship, to manage file access generally. If you think your file management activities call for more details on the interaction of citizenship with file permissions, contact the LC hotline at 925-422-4531 with specific questions.

Top-Level World Permissions Disabled

World (or "other") permissions on top-level files and directories invite unauthorized access and other security problems. Therefore, beginning in April, 1999, an automatic monitoring process systematically disables (converts to permission 0) all world permissions (read, write, and execute) on TOP-level user directories and files in the following file systems

- common-home (/g/g*nn*),
- NFS-mounted temporary (on OCF and SCF, /nfs/tmp0 and /nfs/tmp1),
- /var/tmp (sometimes called /usr/tmp),
- /tmp,
- GPFS (/p/g...) and Lustre (/p/l...),
- /usr/gapps (linked from former /usr/apps)

on each LC production machine where they reside (permissions on files below the top level remain unchanged). Access to TOP-level DFS (distributed file system) (page 18) project directories is similarly restricted, but by using the access control list (ACL) mechanism instead of UNIX permissions (see the DFS section above for details).

Because disabling all top-level world permissions (in the specified file systems) is a security policy, exceptions will require a justification memo with the signature of the relevant LLNL Associate Director. Part of that justification must include a statement that the files or directories do not contain, nor will they ever contain, export controlled or sensitive unclassified information. Only system-owned top-level directories (those with UIDs below 1000) avoid this requirement. (Contact the LC Hotline if you want to apply for a specific exemption to the restrictions on world access.)

Several alternative ways to share files safely are available on LC machines, each with its own strengths and weaknesses. See the File Sharing (page 28) section in this guide for a systematic comparison of these alternatives. When planning such safe file sharing, you may want to discover all your files with a specified permission set. Instructions for doing this with FIND appear in the How to Discover Permissions (page 33) section above.

File-Management Tools

This section summarizes the crucial instructions for several file-management tools that are either unique to the local computing environment (e.g., GIVE and TAKE) or that have special significance here where so many resources are shared by many users (e.g., QUOTA). These software tools are generally available on LC production machines. Because you may want to use these tools on several platforms from different vendors or in combination (even though they were written by different people at different times), note also the sometimes-troublesome unit discrepancies in the last column. (For LC's special storage tools, see EZSTORAGE (URL: http://www.llnl.gov/LCdocs/ezstorage). For LC's special tools to manage the striping of files in the Lustre parallel file system, see the I/O Guide (URL: http://www.llnl.gov/LCdocs/ioguide).)

Tool:	Function:	Units:
GIVE	Offers files to another user to TAKE, even across machines	
TAKE	Accepts GIVEn files from another user, even across machines	
QUOTA	Reports your current local and global disk usage and limits (includes common home directories)	mega/gigabytes (with -v)
LIMIT	Summarizes (and sets) machine configuration limits	kbytes
DU	Reports disk usage (for current or specified directory)	512-byte blocks (default) 1024-byte blocks (with -k)
DF	Reports free disk space (for current or specified file system)	1024-byte blocks
MOLE	Displays text files with better control, better status information, better handling of nonstandard content than MORE	lines, characters
HTAR	Bundles files into TAR-format archives or extracts archive members, for archives on *remote* machines	

NOTE: on LC Linux machines, some standard UNIX file-handling tools have many useful extra options (e.g., DU). All LC-developed file tools have now been ported to Linux. See also the subsection on LC's file-management GUI, called HOPPER (page 55).

GIVE

How to Run GIVE

GIVE transfers files to another specified user (i.e., changes the owner) on the same machine (sometimes also across machines, see below) by copying the files to a holding directory from which the intended recipient (but no one else) can retrieve the files by running TAKE (page 42). GIVE thus lets any user do what normally only a UNIX superuser can do with CHOWN.

To GIVE a file, type

```
give [-i] [-r] [-l] takername flist
give -l [takername] | -u takername [flist]
```

where

takername is the login name of the user you intend to receive the files. You cannot specify several TAKErs with one execution of GIVE, though you can specify several files.

flist is the name, space-delimited list of names, or file filter that specifies the file(s) or directories to GIVE to *takername*. Pathnames are accepted for files not in the current directory.

When you GIVE a file, you always automatically retain the original. Because the transferred copy waits in a special temporary subdirectory it may be purged if the TAKEr delays retreiving it longer than the local purge interval (sometimes as short at 10 days). GIVE options let you confirm (-l) or cancel (-u) GIVEn files not yet TAKEn. If you try to GIVE two files with the same name, only the first is copied to the holding directory (you get a warning about the second). GIVEn files always arrive with the TAKEr as owner and group, and with 600 or 700 permissions, regardless of how you set the group and permissions on the file before you GIVE it.

On OCF IBM machines (only), GIVEn files are copied to the file system /usr/give, which has a whole-system (not per-user) quota of 10 Gbyte of disk space. If you plan to GIVE a few large files (or even many small ones) on an IBM machine, therefore, you should probably confirm that you will not violate this quota by first running

```
df -k /usr/give
```

to get a current report on how much of that 10 Gbyte is still available free space in /usr/give (see the DF section (page 49) below for more background on using DF). There is no quota on the number of files that you can GIVE.

On both OCF and SCF production machines (but not on some special-purpose machines) and for all LC operating systems (AIX and Linux/CHAOS), the transfer directory /usr/give is globally mounted. This means that a GIVEr can run on one machine and a TAKEr can run on another, for across-machine file transfers. Across-machine GIVEs and TAKEs can transfer files between nodes within one cluster (from one /usr/tmp to another, for example) or between entirely separate, dissimilar clusters (such as UP and THUNDER).

GIVE Options

The -i and -r options control file transfers; the -l and -u options manage files already GIVEn.

-i *takername flist*

causes GIVE to interactively prompt about transferring each file in *flist*, to which you respond y (to GIVE) or n (to omit).

-r *takername flist*

recursively GIVEs the contents of all directories in *flist*, including all files in all of their subdirectories. You must use -r if *flist* contains any directories at all.

-l [*takername*]

lists the files you have GIVEn to *takername* that they have not yet retrieved from the holding directory. Without *takername*, lists all unretrieved files you have GIVEn to any user (grouped by user).

-u *takername* [*flist*]

ungives (withdraws, removes) the specified files (or, by default, all files) that you have GIVEn to *takername* that they have not yet retrieved from the holding directory.

GIVE Examples

[1]

GOAL: To give a file to another user on THUNDER0, later confirm that it is available for that user to TAKE, then (still later) change your mind and "ungive" the file.

STRATEGY: (1) Run GIVE and specify the user to receive the file (here DMASON) and the file to transfer (here JLOG2). GIVE confirms your donation automatically.
(2) Later use GIVE's -l option to list your unTAKEn but previously GIVEn files.
(3) Then use GIVE's -u option to revoke the transfer to DMASON of JLOG2. This ungive is NOT automatically confirmed, but you can check by using the -l option again, as shown.

```
give dmason jlog2                        ---(1)
   1 file given.

give -l                                  ---(2)
   dmason has been given:
             1736704 Jan 29 10:30 jlog1

   jsmith has given:
             1640205 Jan 25 11:12 jlog1

give -u dmason jlog2                     ---(3)
   1 files removed.
```

```
give -l
   You have given 0 files.
```

TAKE

How to Run TAKE

TAKE transfers files from another specified user (i.e., changes the owner) on the same machine (sometimes also across machines, see below) by copying the files from a holding directory where the donor has previously put copies by running GIVE (page 39) (and indicating you as the intended TAKEr). TAKE, together with GIVE, thus lets any user do what normally only a UNIX superuser can do with CHOWN.

To TAKE a file, type

take [-i] [-r] *givername* [*flist*]

take [-l [*givername*]]

where

givername is the login name of the user who previously ran GIVE to transfer files to you. To actually retrieve any files you must specify the user who gave them, and you cannot TAKE files from several users at once (though you can TAKE several files from one user).

flist is the name or space-delimited list of names (but NOT a file filter) that specifies the file(s) or directories to TAKE from *givername*. TAKEn files arrive in the directory where you run TAKE, and for TAKEn directories the whole subdirectory structure will be reproduced in your current directory if that structure does not already exist.

When you TAKE a file (a copy), the GIVEr always automatically retains the original. Because the transferred copy waits in a special temporary subdirectory it may be purged if you delay retreiving it longer than the local purge interval (sometimes as short at 10 days). TAKE options let you list (-l) GIVEn files awaiting retrieval or control how files are retrieved (-i, -r). If you try to TAKE a file with the same name as one already in your current directory, TAKE warns you that "xxx already exists locally" and does NOT overwrite the existing file, a safety feature. TAKEn files arrive with you as the owner and group (a contrast with CP or MV) and with default 600 permissions (RW for owner only) or 700 permissions (RWX for owner only), regardless of how the GIVEr set the group and permissions on the original file.

On both OCF and SCF production machines (but not on some special-purpose machines) and for all LC operating systems (AIX and Linux/CHAOS), the transfer directory /usr/give is globally mounted. This means that a GIVEr can run on one machine and a TAKEr can run on another, for across-machine file transfers. Across-machine GIVEs and TAKEs can transfer files between nodes within one cluster (from one /usr/tmp to another, for example) or between entirely separate, dissimilar clusters (such as UP and THUNDER).

TAKE Options

The -i and -r options control file transfers; the -l option reports files already GIVEn.

-i *givername* [*flist*]

causes TAKE to interactively prompt about transferring each file in *flist*, to which you respond y (to TAKE) or n (to omit).

-r *givername* [*flist*]

recursively TAKEs the contents of all directories in *flist*, including all files in all of their subdirectories. You must use -r if *flist* contains any directories at all.

-l [*givername*]

lists the files that *givername* has already GIVEn you that you have not yet retrieved from the holding directory. Without *givername*, lists all unretrieved files you have been GIVEn by any user (grouped by user). The -l option prevents actual file transfers and just reports available files; typing TAKE or TAKE -l yields the same report.

TAKE Examples

[1]

GOAL: To discover which files have been GIVEn to you by a user on THUNDER0, and then retrieve a copy of those files.

STRATEGY: (1) Run TAKE with the -l option to list the files you have been GIVEn but have not yet retrieved.
(2) Then run TAKE with the giver's login name (here DMASON) to actually retrieve the files. You cannot TAKE files without specifying the giver.

```
take -l                                  ---(1)
   dmason has given:
      1270 Feb 5 15:04  log2
      7340 Feb 6 09:15  log3
   2 files.

take dmason                              ---(2)
   Taking files.
       2 file(s) taken.
```

QUOTA

How to Run QUOTA

QUOTA reports your current disk usage both in bytes and in files (sometimes called inodes), as well as the current byte and file limits imposed on you on each file system. In the past, the format of and labels on QUOTA reports varied from one computer vendor or platform to another, and some systems reported disk usage in 1024-byte "blocks." Because those reports were hard to interpret and awkward to display as the numbers grew very large, simple megabyte (M) and gigabyte (G) units now appear in all QUOTA output on LC production machines.

To run QUOTA, type

```
quota [-v]
```

On open and secure IBM/AIX and Linux/CHAOS production machines, QUOTA with -v reports common-home directory usage and limits (along with those of other file systems, if applicable) directly in megabytes (M) or gigabytes (G). The default report (without -v) is null, and no unit conversion is necessary. Thus, QUOTA with -v now replaces the former GLOBAL_QUOTA utility for reporting your LC common-home directory usage and limits on all LC production machines.

WARNING:
On both OCF and SCF, QUOTA *cannot* report data for /nfs/tmp1 or /nfs/tmp0 (the actual current NFS global file systems). LC is pursuing this problem; see DF (page 49) and BDF below.

QUOTA Options

Most QUOTA options (not mentioned here) can be used only by system administrators for managing file systems.

-v	On both IBM/AIX and Linux/CHAOS machines, -v displays disk usage and limit information on mounted global file systems (such as common home directories and /nfs/tmp*) only. Note that "parallel file systems" at LC, such as GPFS on IBM clusters or Lustre on Linux/CHAOS clusters, are *not* considered global (even though LC has begun mounting them on multiple clusters at once), and do *not* support file quotas. Hence QUOTA never includes them in its reports.

QUOTA Examples

[1]

GOAL: To report your current disk usage (including your common-home directory usage) and see how close to the allowed limits you are.

STRATEGY: Run QUOTA with the -v option.
On LC's IBM/AIX and Linux/CHAOS machines, this report is directly in megabytes (M) or gigabytes (G) and covers ONLY the common home directories and the /nfs shared work directories. (On SCF and OCF, /nfs/tmp1 and /nfs/tmp0 are *not* now reported by QUOTA, a known anomaly.)

```
quota -v

Disk quotas for jsmith:
Filesystem  used    quota  limit  timeleft  files  quota   limit  timeleft
/g/g5       721.7M  8.0G   8.0G             2874   n/a     n/a
/g/g17      -0-    16.0G  16.0G             0      n/a     n/a
/g/g50      -0-    16.0G  16.0G             0      n/a     n/a
          [many others too]
```

LIMIT

How to Run LIMIT

LIMIT displays a brief summary of the current machine configuration limits, or, to some extent, sets those limits. To run LIMIT, just type

`limit`

LIMIT Options

In general, any field reported by LIMIT (e.g., coredumpsize) can also be set if you use that field name as a LIMIT option followed by a numerical value. Details vary by operating system (this includes which fields are really setable, exact field names, relevant units, and allowed numerical ranges).

LIMIT Examples

[1]

GOAL: To report the current machine configuration on THUNDER3 in the open Linux/CHAOS cluster.

STRATEGY: Run LIMIT, whose report includes the relevant units for each feature reported.

```
limit

cputime         unlimited
filesize        unlimited
datasize        unlimited
stacksize       unlimited
coredumpsize    16 kbytes
memoryuse       unlimited
vmemoryuse      unlimited
descriptors     1024
memorylocked    128 kbytes
maxproc         1024
```

DU

How to Run DU

DU ("disk usage") reports the amount of disk space used by a specified directory, and by default, the space used by each of the individual subdirectories of that directory as well. If you have many (sub)directories this can be a helpful aid in planning and monitoring your use of disk space, because DU's report is much more fine-grained than are the summed reports from QUOTA.
To run DU, type

du [*options*] [*directory*]

where

options control the units that DU uses and the level of detail in its disk-space reports. By default, DU reports in 512-byte (= 0.5-kbyte) blocks and overtly itemizes all the subdirectories of the target directory.

directory specifies (with a pathname) which directory to report on. By default, DU reports on the directory you are in when you run it.

DU Options

DU's options vary somewhat from one vendor (and hence platform) to another, but these two options are especially helpful and are supported on all local machines.

-k causes DU to report all disk usage in 1024-byte (= 1-kbyte) blocks (the default unit is 512-byte blocks).

-s eliminates the default report of itemized subdirectories and just prints the total disk usage for the one directory you specify on the execute line.

On LC Linux machines only, DU has a dozen useful extra options (mostly to control the block size used or to request totals along with detailed reports).

DU Examples

[1]

GOAL: To see how your disk usage is divided among the subdirectories of your home directory (here, on THUNDER3).

STRATEGY: Run DU and specify your home directory (~) as the target for its report. If you use the -k option, as shown here, the report units will be 1-kbyte blocks (the numbers along the left). Note that the total appears at the end of the list, and the children of each subdirectory are listed above it.

```
du -k ~
```

```
900     /g/g0/trg/.dt
180     /g/g0/trg/.elm
4       /g/g0/trg/.netscape/archive
3       /g/g0/trg/Mail
429     /g/g0/trg/anduin/new
434     /g/g0/trg/anduin
704     /g/g0/trg/banks
4032    /g/g0/trg/baxter
164     /g/g0/trg/chome
     [...many more subdirectories, alphabetically...]
15      /g/g0/trg/eom
719     /g/g0/trg/fis
5818    /g/g0/trg/mail
1021    /g/g0/trg/nmg
2119    /g/g0/trg/test71a
1144    /g/g0/trg/verify
2       /g/g0/trg/nft/case1/case2/case3
3       /g/g0/trg/nft/case1/case2
1       /g/g0/trg/nft/case1/soft
5       /g/g0/trg/nft/case1
1095    /g/g0/trg/nft
856     /g/g0/trg/sgml2fc
224     /g/g0/trg/sybase
284     /g/g0/trg/jcd
37292   /g/g0/trg
```

DF (and BDF)

How to Run DF

DF ("disk free") reports the free space left on any mounted device (file systems and their associated directories), both in absolute terms and as a percentage of the total space available to users. To run DF, type

```
df [options] [filesystem]
```

where

options control the units that DF uses and the level of detail in its disk-space reports. By default, DF reports in 512-byte (= 0.5-kbyte) blocks on IBM/AIX machines, and 1024-byte (= 1-kbyte) blocks on Linux/CHAOS nodes.

filesystem specifies (with a pathname) which file system to report on. By default, DF reports on all currently mounted file systems (but it excludes any automounted file systems, such as the "global" file system supporting the common home directories). See QUOTA (page 44) (discussed above) as a supplement. You can use either the physical device name (e.g., /dev/vartmp on IBM nodes or /dev/hda9 on Linux nodes) or the more familiar name of the associated directory (e.g., /var/tmp).

Details of DF's reports vary by vendor (and hence by platform). Note that when "capacity" (really usage) is reported as a percentage (e.g., 65%), this is not simply a ratio of the used blocks to total blocks (each reported in other columns) because typically some fraction of the total space (often 10%) on a file system is reserved for system use and is not available to users. "Capacity" represents the percentage of this smaller, actually available space that is now *used* (on Linux nodes, this value is more appropriately called "Use%").

Because typical DF output lists huge, often awkwardly aligned byte counts, LC has deployed on each production machine a Perl script called BDF that reports the same columns as DF (total space, amount used, amount free, and "capacity") but in easy-to-read standard units (PB, TB, GB, MB). The example below illustrates this difference in usability.

DF Options

DF's options vary somewhat from one vendor (and hence platform) to another, but one especially helpful option is supported on all local machines.

-k causes DF to report all disk usage in 1024-byte (= 1-kbyte) blocks (the default unit is 512-byte blocks on IBM nodes).

DF Examples

[1]

GOAL: To see how much user space remains on the working file system supporting /var/tmp (on THUNDER3 in the open THUNDER Linux/CHAOS cluster).

STRATEGY: Run DF with the -k option (to get a report in 1-kbyte units) and specify /var/tmp as the target for the report. For an analogous report in much easier to use units, run BDF (shown here below DF).

```
df -k /var/tmp

Filesystem           1k-blocks      Used    Avail      Use%  Mounted on
/dev/sda10            42915560   777672   39957868      2%    /tmp


bdf  /var/tmp

Filesystem               total      used      free       cap  mounted on
/dev/sda10                 41G      760M         39G      2%    /tmp
```

MOLE

How to Run MOLE

MOLE displays text files using MORE-like commands. But MOLE has been enhanced to provide:
(1) forward *and backward* movement within each file as well as among multiple files,
(2) status information on file name, size, and creation date (in a header that begins each displayed file) along with current position (in a prompt at the bottom of each screen), and
(3) counts of hidden lines, signals (bell) for nonASCII characters, and no grotesque attempts to display binary files.
MOLE also lets you jump to a specified line number in the open file, in either direction.

To run MOLE, type

mole [*flist*]

where

flist is the name, space-delimited list of names, or file filter that specifies the file(s) that you want MOLE to display. Pathnames are accepted for files not in the current directory. If run without *flist*, MOLE displays a 10-line help message and ends.

Once started, MOLE steps through each file in 40-line (default) or 20-line (if you toggle with the * command) screens, or one line at a time (using RETURN), with optional backward steps (L command) or jumps between files (with + and -). Like MORE, MOLE does not echo typed commands on the screen and does not require a RETURN after each command to execute it. See the MOLE section (URL: http://www.llnl.gov/LCdocs/fis/index.jsp?show=s6.4) of the File Interchange Service (FIS) Manual for more details on interpreting MOLE's prompt and file header, and for MOLE's role in helping ADCs review files for secure-to-open transfer.

MOLE Options

The first four options here MOLE shares with MORE; the others are unique to MOLE.

RETURN (same as MORE) displays the next line in the open file.

SPACE (same as MORE) displays the next screen (40 lines by default).

? (same as MORE) displays a brief help message (command summary).

q (same as MORE) terminates MOLE (with a report of the number of files reviewed).

* toggles between whole-screen (40-line, the default) display and half-screen (20-line) display steps.

L (ell, either case) displays the previous (40-line or 20-line) screen again.

*nnn*RETURN displays the line numbered *nnn* as well as the (40 or 20) lines that immediately precede line *nnn* (only this option takes a RETURN).

-	displays again the previous file (if you have specified several on MOLE's execute line). A one-line header indentifies each file displayed when the display starts.
+	displays the (sequentially or alphabetically) next file (if you have specified several on MOLE's execute line). A one-line header identifies each file displayed when the display starts.

MOLE Examples

[1]

GOAL: To display first one file (test1), then another (test2), including jumps to a specific line and display of hidden lines.

STRATEGY: (1) Run MOLE with both file names on its execute line.
(2) Toggle to the 20-line step size and request the next 20 lines.
(3) Jump to line 400 in test1 (and the 20 lines that precede it). As always, MOLE's prompt reveals this new position by line number.
(4) Move to the "next" file (test2). Note that MOLE reports in its one-line file header that test2 contains 8 hidden lines, and any that fall near the front will automatically appear along with the first 20 lines displayed here.
(5) Quit MOLE (note the final report).

```
User: mole test1 test2                                  ---(1)
Rtne: [test1]: 408L/21957 Jan 5, 1998 11:03             [MOLE's header]
      ...first 40 lines of test1 displayed here...      [text]
      ? l/f space L # + - * Q  *** 40/A test1 ***       [MOLE's prompt]
User: *                                                 ---(2)
Rtne: [bell]                                            [MOLE acknowledges]
User: [space]                                           [request next screen]
Rtne:
      ...next 20 lines of test1 displayed here...       [text]
      ? l/f space L # + - * Q  *** 60/A test1 ***       [MOLE's prompt]
User: 400RETURN                                         ---(3)
Rtne:
      ...20 lines of test1 ending at line 400...
      ? l/f space L # + - * Q  *** 400/A test1 ***      [note new
                                                        location]
```

```
User: +                                                  ---(4)
Rtne: [test2]: 110L/5331 Apr 9, 2001 3:17 *** 8 hidden lines***
                                                         [MOLE's header]
      ...first 20 lines of test2 displayed here,         [text]
      including any hidden lines in this range...
      ? l/f space L # + - * Q  *** 20/A test2 ***        [MOLE's prompt]
User: q                                                  ---(5)
Rtne: [2 ASCII files reviewed]                           [final status report]
```

HTAR

HTAR is basically a (locally developed) interface to LC's (open or secure) storage system HPSS. But it behaves like (and shares much execute-line syntax with) the traditional UNIX TAR utility, so in that sense HTAR is also a file-bundling and archive-management tool. Also (like LC's NFT (URL: http://www.llnl.gov/LCdocs/nft)), you can request HTAR to create remote archives on or extract member files from a remote archive on *any* LC production machine if you wish (by using HTAR's -F option).

HTAR's specially tuned features include:

- Uses a TAR-like syntax and supports TAR-compatible archive files by relying on the POSIX 1003.1 TAR file format.
- Bundles files in memory using multiple concurrent threads and transfers them into an archive file built *directly* in storage by default (or on any *remote* LC machine that you specify), to avoid needing extra local online disk space.
- Takes advantage automatically of available parallel interfaces to storage to provide fast file transfers (measured at as high as 150 Mbyte/s, which exceeds 30 times the typical rate for transferring small files separately).
- Uses an external index file to easily accommodate thousands of small files in any archive, and to support retrieval of specified files from within a remote archive *without* first retrieving the whole archive from HPSS (or from another remote machine). You can even request that HTAR index remote TAR archives that it did not create.
- Imposes no limit on the total size of the archives that it builds and accepts input files (archive members) as large as 8 Gbyte.
- Allows easily building (with -n) and storing *incremental* archives (consisting of only recently changed files).

When the storage system (HPSS) is up and available to users you can execute HTAR with a command line that has the general form

htar *action archive* [*options*] [*filelist*]

and the specific form

htar -c|t|x|X|K -f *archivename* [-BdEFhHILmMoOpSTvVw] [*flist*]

where exactly one action and the *archivename* are always required, while the control options and (except when using -c) the *filelist* can be omitted (and the options can share a hyphen flag with the action for convenience). Users familiar with TAR can guess how to run HTAR from this model (although there are some tricky syntax differences). Others should consult the HTAR Reference Manual (URL: http://www.llnl.gov/LCdocs/htar) for usage suggestions, annotated examples, technical tips, full option details, and tips for circumventing known problems. The HOPPER controller (page 55) can run HTAR graphically if you wish.

HOPPER: A File-Management GUI

SERVICES:

HOPPER offers an alternative, graphical interface to (a GUI controller for) these file-transfer services at LC:

- FTP, SFTP, and NFT (on its CONNECT menu, as a pull-down on the CONNECT TO REMOTE option),
- HTAR (on its CLIPBOARD menu to create an archive; *double* click on an existing archive's directory entry to report its contents, then copy wanted items to HOPPER's CLIPBOARD),
- "Storage HPSS" (really opens an FTP connection to your top-level directory at storage.llnl.gov by default), and
- File Interchange Service (FIS) for transfers between LC's open and secure networks.

FEATURES:

By invoking HOPPER you can do many file-management tasks graphically, including:

- Use one common, desktop-like interface to run a variety of otherwise disparate file-handling tools.
- Avoid learning the syntax and specific options for tools that are usually executed by a UNIX-style command line with explicit arguments.
- Pick visually an *alphabetically scattered* set of files to transfer from a larger set, where file filters won't help.
- Pick *graphically* from a set of already HTAR-archived files a subset to extract (sometimes tricky with a command line).

HOPPER's SELECT menu also offers a USE WILDCARD option that gives you the convenience of a file filter within its graphical framework to accelerate those cases where filtering meets your file-selection needs.

STRATEGY:

HOPPER is written in Java. It has been installed on all LC production machines, open and secure, and you can find instructions on the support website (below) for downloading the source for use on desktop computers. You use HOPPER by:

- Executing the program,
- Picking graphically (CTRL-CLICK with your mouse) the *files* on which to operate from its displayed list or table of those in your chosen directory, and then
- Specifying (from a menu) which *operation* to perform on those files (such as put into or retrieve from a remote directory). You can also define a set of UNIX commands in advance and then "launch" any of them as (additional, customized) operations on your selected target file(s). See the LAUNCH COMMANDS subsection below for instructions.

PERFORMANCE:

Because transferring many files and displaying much graphical data, both perhaps remotely, can in some circumstances slow performance you will find HOPPER use more satisfying the closer to the top of this spectrum of conditions you can arrange to execute it:

- Ideal--run HOPPER on your local desktop machine (where you want it to display), even when moving files on remote production hosts.
- Adequate--connect to a remote production host by using SSH on your local desktop machine, and run HOPPER remotely to display back to your local host over X11. On Windows computers, use XWin-32's built-in SSH.
- Least favorable--connect to a remote production host by using F-Secure SSH on your local desktop, and run HOPPER remotely to display back to your local host over X11.

LAUNCH COMMANDS:

To expand the set of operations that HOPPER lets you perform on a file, HOPPER includes the ability to define a customized set of UNIX commands and then (later) execute any of those commands on a target file (so long as the target file resides on a UNIX system, even if you run HOPPER elsewhere).

- Defining a "launch command":
  To define a customized command, work down this chain of menus starting at FILE:
  File
  Preferences
  Operations
  Launch Command
  Fill in the displayed dialog box with a mnemonic label ("command name," shows in a later menu) and "command line" (exact UNIX syntax to execute). Use the string %s to parameterize the command. For example, supplying name PRINT and line
  lp -d p280 %s
  lets you (later) click on a PRINT menu entry to execute the above specific UNIX printer command on your current HOPPER target file. Click on the ADD button to save each newly defined command.
- "Launching" a previously defined command:
  Select (*right* click) a target file from HOPPER's directory table (or choose "Ops on Selected Entries" from HOPPER's OPS menu). HOPPER offers a drop-down menu of possible operations (not alphabetized) on that file, including "Launch Command." Select Launch Command to see a second drop-down menu (alphabetized) of your previously defined UNIX customized command names (such as PRINT, from above). Click on any name to execute the corresponding command on the UNIX machine (local or remote) where your target file resides.

SUPPORT:

More general background information on HOPPER is available at the (OCF-only) project web site:

`http://www.llnl.gov/hopper`

See "Getting Started" for instructions on how to download HOPPER to your local desktop machine.

Tools for Obsolete File Types

To help you properly identify still-needed files that were made on LC's former CRAY and CDC 7600 computers, and to (optionally) convert some files that can be rescued, LC provides several tools for managing obsolete file types. These special tools usually perform familiar tasks (such as listing file formats or unpacking libraries) but on file types ignored by standard UNIX tools. See also the "Cray File Conversion" section of the EZOUTPUT (URL: http://www.llnl.gov/LCdocs/ezoutput) basic guide for how IBM's (but *not* Compaq's) TRANS tool can sometimes help with such legacy files.

Among the available obsolete-file tools on LC production machines are:

lft [-c|-n] *filelist*

> reports for each file in *filelist* (a blank delimited list of files or a UNIX file filter) two columns that specify each file's name and its file type (ascii, directory, lib, tar, unrecognized). LFT reports GIF, PDF, and PS files as ascii. With -n, LFT reports on CRAY and CDC 7600 files without conversion. With -c, LFT converts each (specified) CRAY or CDC 7600 text file to UNIX format (but leaves all other files unchanged).

lib76 *libname* l|x|b|a [*filelist*]

> processes CDC 7600 LIB-format and LIX-format library files. Specify the library with *libname* and the files to process within it with *filelist*. Here option l lists the files, x extracts them as text, b extracts them as binary files, and a extracts them as ascii and converts them to UNIX format. With no *filelist*, LIB76 processes all files in the specified library. WARNING: odd-number length files will gain 4 extra bits, and LIB76 will overwrite without warning any existing local files with the same name as those it extracts.

Using Groups

BASIC ADVICE:

A group is a named set of users created by a system administrator to enable easier file sharing among group members. Every user at LC belongs by default to a group (of one) with the same name as their login name, and your newly created files are assigned by default to that unique group. But if you belong to other groups as well, you can change your default group (for the current session) or change the group to which any of your files is assigned, to take advantage of the group permissions (page 32) explained in a previous section. (On LC machines, software constraints limit every user to membership in no more than 32 groups.)

This table shows how to perform the most common group-related tasks:

Group-related task:	**Command:**
Reveal who belongs to a specified group	uinfo group *grpname* grep *grpname* /etc/group
Reveal all groups to which you belong	groups uinfo user *username*
Reveal all groups to which *username* belongs	groups *username* uinfo user *username*
Change your default group to *grpname*	newgrp *grpname*
Restore your original default group	newgrp
Change a file's group assignment	chgrp *grpname filename*
Change a file's group permissions	See chmod
Create or join a group at LC	Contact LC Hotline

WARNINGS:

(1) Storage Groups. Your group membership on LC's archival storage systems (storage.llnl.gov and storage.scf.cln) is likely to differ from your group membership on LC's production machines (which also often differ among themselves). You can use group assignments to control the sharing of stored files, but only if you discover who belongs to which storage groups, and only if you assign a file to a group AFTER you store it (since group assignment does NOT persist during file transfer). To reveal your group memberships (or the members of a specified group) on the LC storage systems you must run the complex LDAPSEARCH utility as explained in the "Using Storage Groups" section of the EZSTORAGE (URL: http://www.llnl.gov/LCdocs/ezstorage/index.jsp?show=s5.1) guide.

You can still change the group to which a stored file is assigned by using NFT's **`chgrp`** option or FTP's **`quote site chgrp`** option, but only if you belong to the target group. (Despite its name, NFT's **`group`** option begins asynchronous file transfers and has nothing to do with managing file-permission groups.)

(2) AIX 16-Group Limit on /nfs/tmp*n*. On LC AIX machines *only* (not under Linux), the globally mounted /nfs/tmp*n* directories have a limit of 16 user groups. Attempting to access files in these directories on AIX machines if you belong to more than 16 groups may cause unpredictable access-denied errors for your applications. These NFS-mounted directories process group membership in the order in which group information comes from the AIX /etc/group system file, and this may *not* be the order in which the groups were created (or in which you joined them). Sometimes joining an unimportant seventeenth (or more)

group causes /nfs/tmp*n* to ignore your membership in much more important "earlier" group(s), with very disruptive file-access results. Some AIX users have reported this problem intermittently with their NFS-mounted home directories as well. Linux/CHAOS handles group verification differently and avoids this 16-group limit. GPFS (the AIX parallel file system) does not share this problem.

(3) Lustre Groups. Support for group access to files on the Lustre (Linux/CHAOS) parallel file systems is sometimes faulty or absent for other than your primary group. See the "Lustre Groups" section (URL: http://www.llnl.gov/LCdocs/ioguide/index.jsp?show=s7.4.6) of the I/O Guide for LC for both an interactive and a batch-oriented way to work around this known limitation.

(4) DFS Groups. LC's Distributed File System (DFS) handles groups differently than the UNIX operating system does. As with storage, a file's previous mainframe group assignment reverts to your default group (the one-member group that matches your login name) when you move or copy a file into DFS. Under DFS, a file can be assigned to many groups (not just one as with UNIX), all posted by the **aclmod** utility on the file's access control list (ACL). If you belong to several groups, your DFS permissions for a file are the logical sum (union) of the permissions of every group you belong to that is assigned to the file. And finally, under DFS you can use **aclmod** to assign a group to a file even if you do not belong to that group yourself (in contrast with **chgrp**). Because of these differences, you should carefully review the instructions for and examples of **aclmod** in LC's DFS documentation before you rely on group behavior under DFS. (Note that DFS is currently mounted only on LC IBM POWER3 nodes (running AIX 5.2) but *not* on any other AIX clusters, Linux nodes, or on LC's special-purpose Suns.)

RELEVANT FORMS:

To create, delete, or change the membership of a group on any open or SCF machines, you must submit one of these (signed) forms to the LC Hotline (L-63).

```
SCF-2  Create/Update Group
SCF-9  Delete Group
```

The blank forms are available by request from the Hotline (page 4) and from these LC web sites:

```
Open: https://www.llnl.gov/lcforms
SCF:  https://www.scf.cln/lcforms
```

The section once linked from this reference no longer exists.

Disclaimer

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial products, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government thereof, and shall not be used for advertising or product endorsement purposes.

(C) Copyright 2007 The Regents of the University of California. All rights reserved.

Keyword Index

To see an alphabetical list of keywords for this document, consult the next section (page 64).

Keyword	Description
entire	This entire document.
title	The name of this document.
scope	Topics covered in EZFILES.
availability	Where these programs run.
who	Who to contact for assistance.
introduction	Role and goals of EZFILES.
directories	How public directories are organized.
linux-directories	Directories, properties on open clusters.
linux-structure	How open cluster directories are organized.
linux-properties	Home, work dirs. compared on clusters.
lustre-properties	Parallel file systems for Linux.
linux-scf-directories	Directories, properties on secure clusters.
linux-structure-scf	How SCF cluster directories are organized.
linux-properties-scf	Home, work dirs. compared on SCF clusters.
ibm-directories	Directories, properties on IBM SPs.
ibm-structure	How IBM/AIX directories are organized.
gpfs-properties	Parallel file systems for IBM/AIX.
dec-directories	Directories on former Compaq/DEC clusters.
dfs-nfs-comparison	Contrasts DFS and NFS shared disks.
nfs	Network File System properties.
dfs	Distributed File System properties.
parallel-file-systems	Names, policies for LC parallel file systems.
common-home	Home directories shared across machines.
file-management	Techniques and issues in work with files.
work-steps	Typical code-development scenario.
transfer-rates	CP vs. FTP from mounted file systems.
suitability	Intended-use comparison of file systems.
file-sharing	File-sharing alternatives compared.
backup	LC file backup policy by directory.
purge	When and how LC purges file systems.
search-paths	Search path features analyzed.
permissions	Managing UNIX file permissions.
access-rights	Managing UNIX file permissions.
permission-kinds	R, W, X for U, G, O explained.
permission-reports	How to discover file permissions.
chmod	How to change file permissions.
chmod-octal	Absolute, numeric CHMOD syntax.
chmod-symbolic	Symbolic, incremental CHMOD syntax.
citizenship-groups	How citizenship affects SCF permissions.
permission-disabled	Where top-level world perms disabled.
file-tools	Local software tools for file handling.

Alphabetical List of Keywords

Keyword	Description
a	The alphabetical index of keywords.
access-rights	Managing UNIX file permissions.
availability	Where these programs run.
backup	LC file backup policy by directory.
bdf	Reporting free space by file system.
chmod	How to change file permissions.
chmod-octal	Absolute, numeric CHMOD syntax.
chmod-symbolic	Symbolic, incremental CHMOD syntax.
citizenship-groups	How citizenship affects SCF permissions.
common-home	Home directories shared across machines.
date	The latest changes to EZFILES.
dec-directories	Directories, properties on Linux clusters.
df	Reporting free space by file system.
df-examples	Sample annotated uses of DF.
df-execute-line	How to run DF.
df-options	Options to control DF use.
dfs	Distributed File System properties.
dfs-nfs-comparison	Contrasts DFS and NFS shared disks.
directories	How public directories are organized.
du	Reporting disk usage by (sub)directory.
du-examples	Sample annotated uses of DU.
du-execute-line	How to run DU.
du-options	Options to control DU use.
entire	This entire document.
file-management	Techniques and issues in work with files.
file-tools	Local software tools for file handling.
file-sharing	File-sharing alternatives compared.
give	Donating files to another user.
give-examples	Sample annotated uses of GIVE.
give-execute-line	How to run GIVE.
give-options	Options to control GIVE use.
gpfs-properties	Parallel file systems for IBM/AIX.
groups	Revealing or changing file groups.
hopper	Graphically manipulating files.
htar	Making, managing file archives.
ibm-directories	Directories, properties on IBM SPs.
ibm-structure	How IBM/AIX directories are organized.
index	The structural index of keywords.
introduction	Role and goals of EZFILES.
lft	Tool for obsolete (legacy) file types.
lib76	Tool for obsolete (legacy) file types.
limit	Reporting machine configuration limits.
limit-examples	Sample annotated uses of LIMIT.
limit-execute-line	How to run LIMIT.
limit-options	Options to control LIMIT use.
linux-directories	Directories, properties on open clusters.
linux-properties	Home, work dirs. compared on clusters.
linux-properties-scf	Home, work dirs. compared on SCF clusters.
linux-scf-directories	Directories, properties on secure clusters.
linux-structure	How cluster directories are organized.
linux-structure-scf	How SCF cluster directories are organized.

Date and Revisions

Revision Date	Keyword Affected	Description of Change
14Jun07	linux-properties	/var/tmp purged on diskless nodes.
	suitability	/var/tmp purged on diskless nodes.
	purge	/var/tmp purged on diskless nodes.
	lustre-properties	SLURM flushes Lustre page cache.
01May07	linux-directories	16-group AIX /nfs/tmp warning added.
	ibm-directories	16-group AIX /nfs/tmp warning added.
	groups	16-group AIX /nfs/tmp limit explained.
29Jan07	linux-directories	Structure diagram, property table now in Linux section, updated.
	ibm-directories	Subdivided and updated discussion.
	dec-directories	Old Compaq section now just a placeholder with cross refs.
	lustre-properties	New subsection with Lustre details.
	gpfs-properties	New subsection with GPFS details.
	common-home	Compaq details, examples replaced.
	file-management	Compaq details removed or replaced.
	backup	Parallel f.s. names changed.
	search-paths	Compaq example deleted.
	quota	Compaq details replaced.
	limit	Compaq example replaced.
	bdf	BDF tool, DF comparison added.
	hopper	Text details updated.
	index	New keywords for new sections.
19Sep06	parallel-file-systems	New section on name, policy changes.
	index	New keyword for new section.
15Aug06	ibm-directories	Contrasts with Lustre purge noted. White references removed.
	linux-directories	New Lustre purge policy explained.
	work-steps	Cross refs to GPFS, Lustre added.
	purge	Purge comparison section added.
	index	New keyword for new section.
18Jul06	nfs	Parallel I/O warning expanded.
	common-home	Parallel I/O warning expanded.
	dfs	AIX 5.3 does not support DFS.

	work-steps	LOADL_STEP_ID not under SLURM.
24May06	ibm-directories	GPFS/Lustre comparison link added.
	linux-directories	Implementation details added.
	backup	Parallel file system concerns added.
	groups	Lustre group-problem link added.
05Dec05	hopper	Launch-command feature added.
12Oct05	ibm-directories	Parallel I/O warning added.
	linux-directories	Parallel I/O warning added.
	nfs	Parallel I/O warning added.
	file-tools	Disk-striping tool cross ref added.
04Aug05	directories	NFS tmp0, tmp1 replace all others, many details updated throughout.
	common-home	Machine list updated.
	htar	New features noted.
	hopper	New section on local GUI.
	index	New keyword for new section.
20Jun05	dec-directories	TC2K deleted throughout.
	dec-scf-directories	SC reduced to 8 nodes.
	linux-directories	Adelie, Emperor no longer GA.
23May05	ibm-directories	UV, Tempest noted.
	linux-directories	Lilac noted (goes GA).
	limit	Setting limits clarified.
	index	New keyword added for Linux directories.
07Feb05	pentium-directories	Lustre parallel f.s. added.
	ibm-directories	GPFS details, alias added.
	suitability	Lustre added to chart.
	quota	No parallel file system quotas.
22Nov04	dec-directories	Temp directory details updated.
	ibm-directories	SCF machine changes noted.
	backup	Temp directory details updated.
	quota	SCF /nfs/tmp reporting flaws persist.
30Aug04	common-home	Blue replaced in examples.
	htar	Features, details updated.
06Apr04	pentium-directories	ACE up to 128 nodes (SCF).
	quota	SCF tmp1, tmp3 ignored by QUOTA.
	dec-scf-directories	

		/nfs/tmp1 and tmp3 replace 0 and 2 on SCF.
	ibm-directories	
		/nfs/tmp1 and tmp3 replace 0 and 2 on SCF.
	suitability	/nfs/tmp1 and tmp3 replace 0 and 2 on SCF.
08Dec03	pentium-directories	
		ACE SCF details added.
	file-tools	GIVE, TAKE now cross all machines.
	give	General across-machine GIVEs ok.
	take	General across-machine TAKEs ok.
01Oct03	dec-directories	
		LX cluster departs.
	dec-scf-directories	
		Furnace departs, ICF noted.
	pentium-directories	
		MCR and other details added.
	common-home	Furnace departs, MCR added.
08Sep03	common-home	Larger SCF home directories.
	quota	QUOTA output now uniform everywhere.
04Aug03	directories	/nfs/tmp subnames, numbers updated.
	work-steps	/nfs/tmp3 added.
	backup	/nfs/tmp3 added.
	permission-disabled	
		/nfs/tmp3 added.
	htar	Between-machines role (-F) added.
09Jul03	dec-directories	
		/nfs/tmp* purge clarified.
	dec-scf-directories	
		/nfs/tmp* purge clarified.
	pentium-directories	
		More ILX nodes added.
09Jun03	common-home	How to test for availability.
	htar	Now available on Linux too.
	give	Some across-machine GIVEs ok.
	take	Some across-machine TAKEs ok.
	search-paths	Tru64 order changed, ECHO noted.
16Apr03	common-home	Name changes, new OCF quotas.
	backup	Clarifying cross ref added.
	quota	Col heads, default output elaborated.
	df	Between-machine differences clarified.
24Mar03	introduction	Cross ref to CHAOS manual added.
	dec-directories	
		Machine names, details updated.
	dec-scf-directories	
		LC/Furnace match noted.
	pentium-directories	
		ILX cluster added, with CHAOS links.
	transfer-rates	ILX replaces LX in example.
	suitability	On Linux, /tmp links to /var/tmp.
	file-tools	Local tools now on Linux too.
01Oct02	dec-scf-directories	
		Clarified because Forest departs.

	pentium-directories	DFS not on any Linux nodes.
	dfs	DFS not on any Linux nodes.
	backup	DFS has auto backup.
	groups	DFS access limits noted.
29Jul02	ibm-directories	GPFS purge details added.
	work-steps	IBM /var/tmp subdirs noted.
01Jul02	lft	New section on legacy-support tool.
	lib76	New section on legacy-support tool.
	groups	UINFO roles added.
	index	New keywords for new section.
21May02	dec-scf-directories	/var/tmp size variations noted.
	pentium-directories	New section on SCF Pentium clusters.
	common-home	Linux dot file added.
	index	New keyword for new section.
05Mar02	directories	New clusters replace old ones (Compass out, GPS in, etc.).
	common-home	Example and details updated.
	file-tools	Many example details revised.
	index	Separate TERA section deleted.
04Dec01	dec-scf-directories	Home /g replaces /u on Forest.
	ibm-directories	Purge started on SCF GPFS.
	file-tools	Some LC tools not on Linux. DU has extra Linux options.
01Oct01	give	IBM-only GIVE quota explained.
06Sep01	file-tools	Table updated re HTAR.
	htar	New section on new local tool.
	index	New keyword for new section.
17Jul01	dec-scf-directories	SCF /nfs/tmp1 links to /nfs/tmp2.
	ibm-directories	SCF /nfs/tmp1 links to /nfs/tmp2.
	dfs-nfs-comparison	SCF /nfs/tmp1 links to /nfs/tmp2.
	work-steps	SCf /nfs/tmp1 links to /nfs/tmp2.
	groups	Now 32 groups per user.
04Jun01	dec-properties	Limit on /nfs/tmp files/user added.
	dec-properties-scf	Limit on /nfs/tmp files/user added.
	ibm-directories	Limit on /nfs/tmp files/user added.
	quota	/nfs/tmp limit not reported.
17May01	mole	New section on new local tool.
	index	New keywords for new subsections.

03May01	entire	/usr/gapps replaces /usr/apps (OCF, SCF). /nfs/tmp0 and tmp1 replace tmp2 (OCF).
12Mar01	dfs	DFS access restrictions explained.
	permission-disabled	Cross ref to DFS access limits.
04Dec00	directories	/nfs/tmp2 replaces /nfs/tmp1 on OCF.
	suitability	/nfs/tmp2 added to comparison.
	work-steps	/nfs/tmp2 replaces /nfs/tmp1 on OCF.
	backup	/nfs/tmp2 not backed up.
	permission-disabled	/nfs/tmp2 added to list.
14Nov00	search-paths	New DEC/Compaq search path. How to add application setup files.
21Sep00	file-sharing	New section compares ways to share files.
	suitability	New section compares file-system roles.
	ibm-directories	New section on SPs, GPFS.
	groups	Only 16 groups per user.
	backup	GPFS not backed up.
	index	New keywords for new sections.
08Mar00	entire	All CRAY references deleted, including sects on CRAY dirs, search paths.
14Dec99	dec-scf-directories	/g now optional SCF common home.
	j90-directories	/g now optional SCF common home.
	common-home	/g mostly replaces /u (SCF).
	dfs-nfs-comparison	New URL for DFS details.
12Aug99	dec-structure-scf	Added /nfs/tmp1 to diagram.
	dec-properties-scf	Added /nfs/tmp1 col to table.
20Jul99	dec-scf-directories	Two subsections added.
	index	New keywords for new sections.
11May99	permission-reports	How FIND supplements LS.
	permission-disabled	World permission restrictions.
	index	New keyword for new section.
25Mar99	introduction	GLOBAL_QUOTA deleted.
	common-home	SCF clarified, GLOBAL_QUOTA gone.
	file-tools	Table updated re QUOTA. GLOBAL_QUOTA sections deleted.
	quota	Role expanded, examples revised.
	index	All GLOBAL_QUOTA keywords deleted.
20Jan99	common-home	/gadmin/etc replaces /g/etc.

15Dec98	common-home	World write access forbidden.
18Nov98	dec-directories	Shared /nfs/tmp1 dir. explained.
	dec-scf-directories	Absence of /nfs/tmp1 noted.
	file-management	Role of /nfs/tmp1 included.
	dfs-nfs-comparison	New section, details added.
	index	New keywords for new section.
18Sep98	groups	Storage group info updated.
	quota	New role on open DECs.
	global-quota	Now SCF use only.
	give	Permissions, groups clarified
	take	Permissions, groups clarified
	file-tools	Comparative chart updated.
	common-home	Backup added, details updated.
13Aug98	groups	Groups advice section added.
22Jun98	tera-cluster	New section on new machines.
	citizenship-groups	SCF citizenship effects noted.
	backup	Y-MP, migration comments deleted.
	index	New keywords for new sections.
	entire	Y-MP details deleted throughout.
04Feb98	dec-directories	/u deleted, details clarified.
	dec-scf-directories	New section on Forest added.
	common-home	Forest SCF comments added.
	work-steps	Open/SCF clarifications.
	dec-path	Open/SCF clarifications.
	global-quota	Forest SCF quotas noted.
	index	New keyword for new section.
06Jan98	dec-properties	Size quotas updated.
	who	Link/ref to doc matrix added.
28Oct97	global-quota	File quota gone, DU note added.
	dec-properties	File (count) quota gone.
03Oct97	global-quota	Usage details, example added.
	common-home	Link, server names updated.
	dec-properties	Home quotas, servers updated.
04Jun97	global-quota	Initial use on SKY noted.
18Feb97	entire	First edition of LC EZFILES manual.

TRG (14Jun07)

UCRL-WEB-200067
Privacy and Legal Notice (URL: http://www.llnl.gov/disclaimer.html)
TRG (14Jun07) Contact: lc-hotline@llnl.gov