Parallel Data Archive in HEC Environment

DISC

University of Minnesota
Digital Technology Center
Intelligent Storage Consortium

David DU

Supported by

Symantec, Sun Micro, LSI Logic Storage Systems, ETRE/Korea, ITRI/Taiwan

DOE, ONR, Cisco, Intel

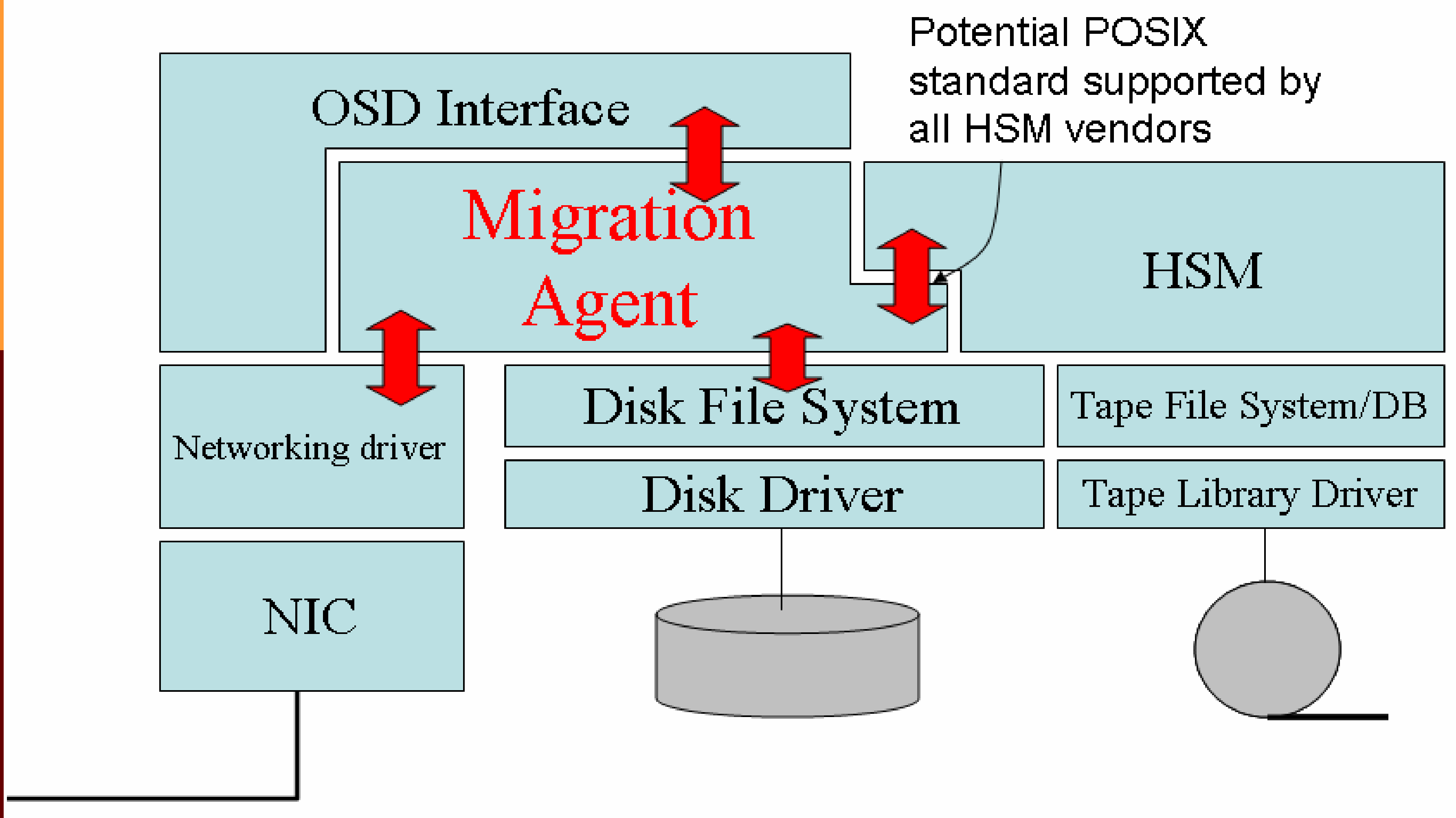
Current Research Projects in DISC

- Parallel Archive in HEC Environment
- Long-Term Key Management
- OSD Design and Implementation
- Massively Array of Idle Disks (MAID)
- QoS Specification and Enforcement for Remote Storage Accesses
- SQUAD: A unified framework for Storing and Querying Unstructured And Structured Data with intelligent storage

1. Creating OSD-Enabled Tape Library

2. High Performance File System

3. Migration Agent for Multi-Vendor HSMs

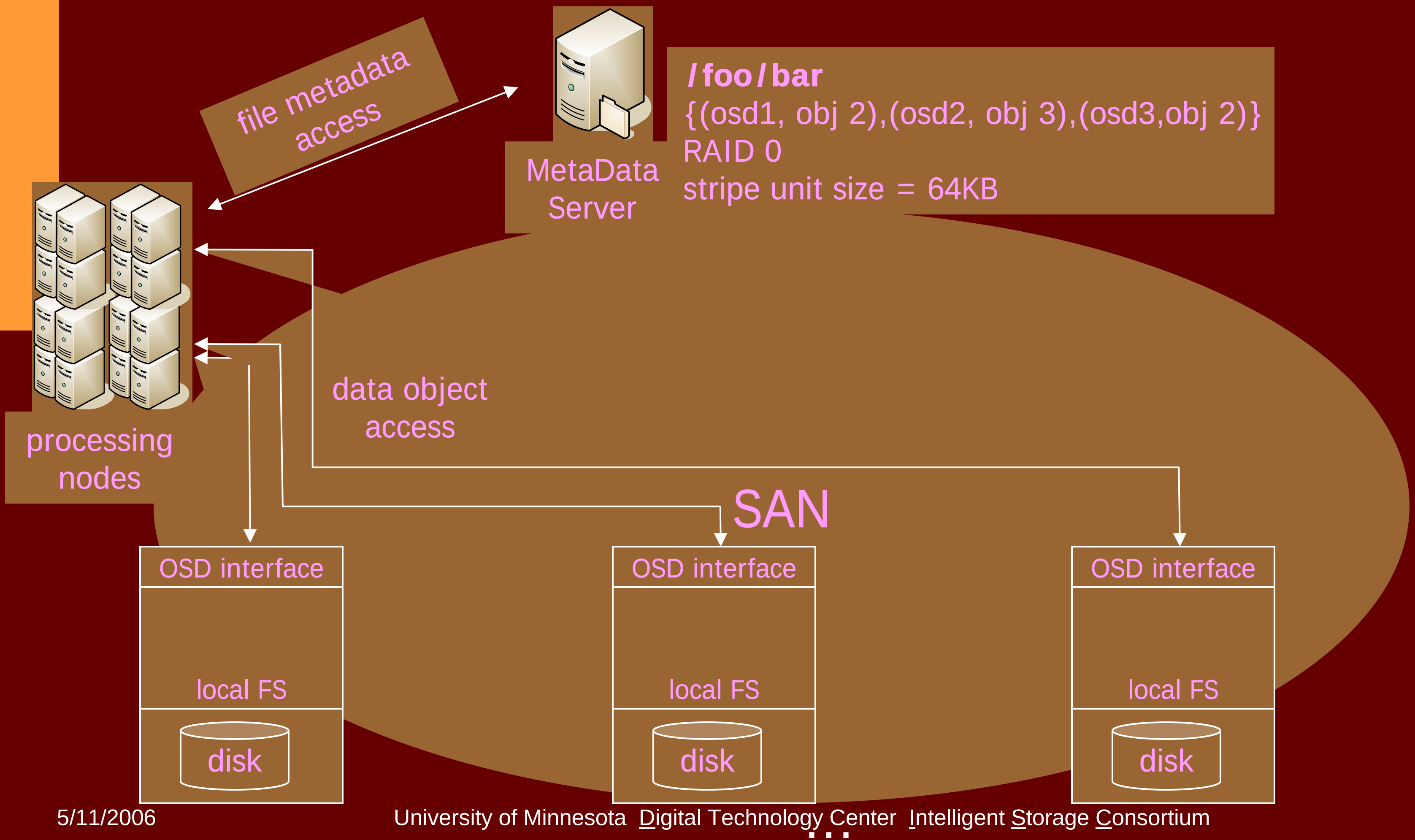


Demanding for Scalable, Global and Secure (SGS) file system

- Tri-Lab File System Path Forward RFQ
  - Global name space
  - Security
  - Scalable infrastructure for clusters and enterprise
  - No single point of failure
  - POSIX-like Interface
  - Work well with MPI-IO

…

Object-based Cluster File System (OCFS)



Recent OCFS Solutions

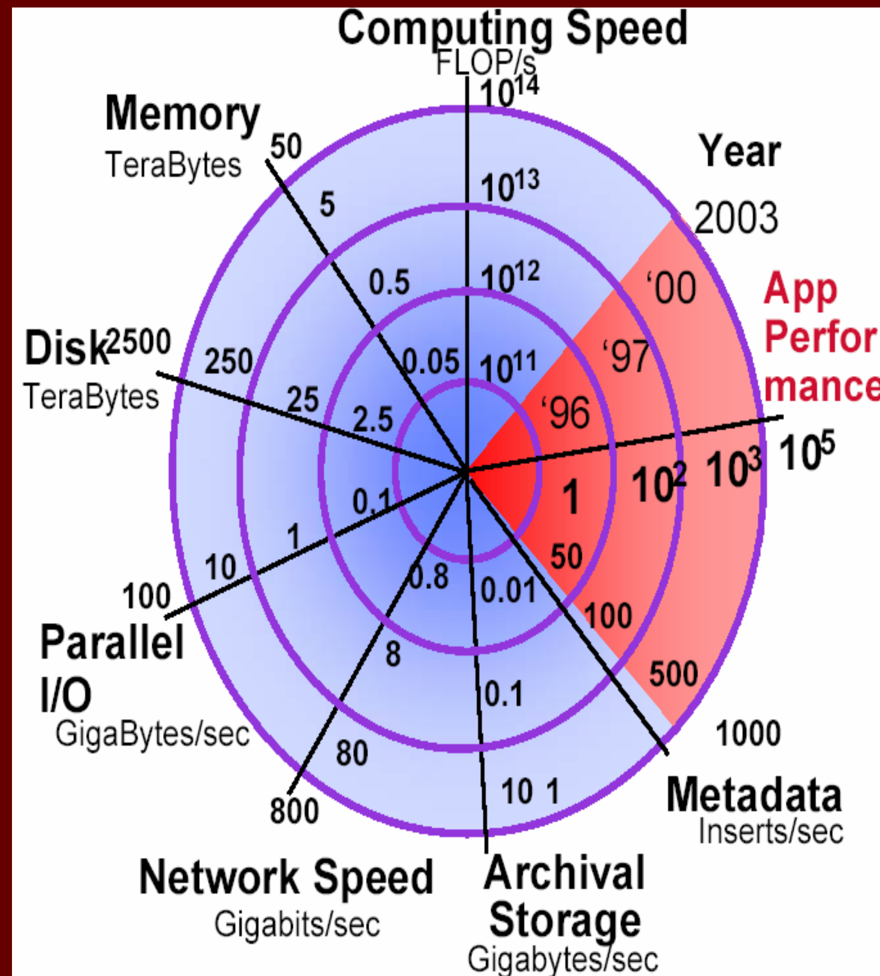
- Lustre File System of CFS, Inc.
  - LLNL runs Lustre on Multi-programmatic Capability Cluster (MCR)
  - 20 million files and 115.2TB
  - Aggregate I/O 22GBps
- ActiveScale File System of Panasas
  - LANL deploys three systems
  - The largest one has 200TB capacity and ~20GBps aggregate I/O

Why storage hierarchy in SGS file systems?

- Fast data generating rate in HPC environment
  - simulations generate one new file of multiple terabyte every 30 minutes
  - Post analysis reads such multiple terabyte files
- Cost effectiveness
  - Combination of expensive high-performance storage and more affordable low-performance storage
- Data lifecycle
  - Inactive data should not occupy precious resources
- Infinite storage
  - Any data may be useful in the future

Bottleneck in data migration throughput

- Enlarging gap between application parallel I/O and archival storage I/O
- Backup and Restore record in 2003
  - Achieved by SGI
  - 2.8GBps file-level backup
  - 1.25GBps file-level restore



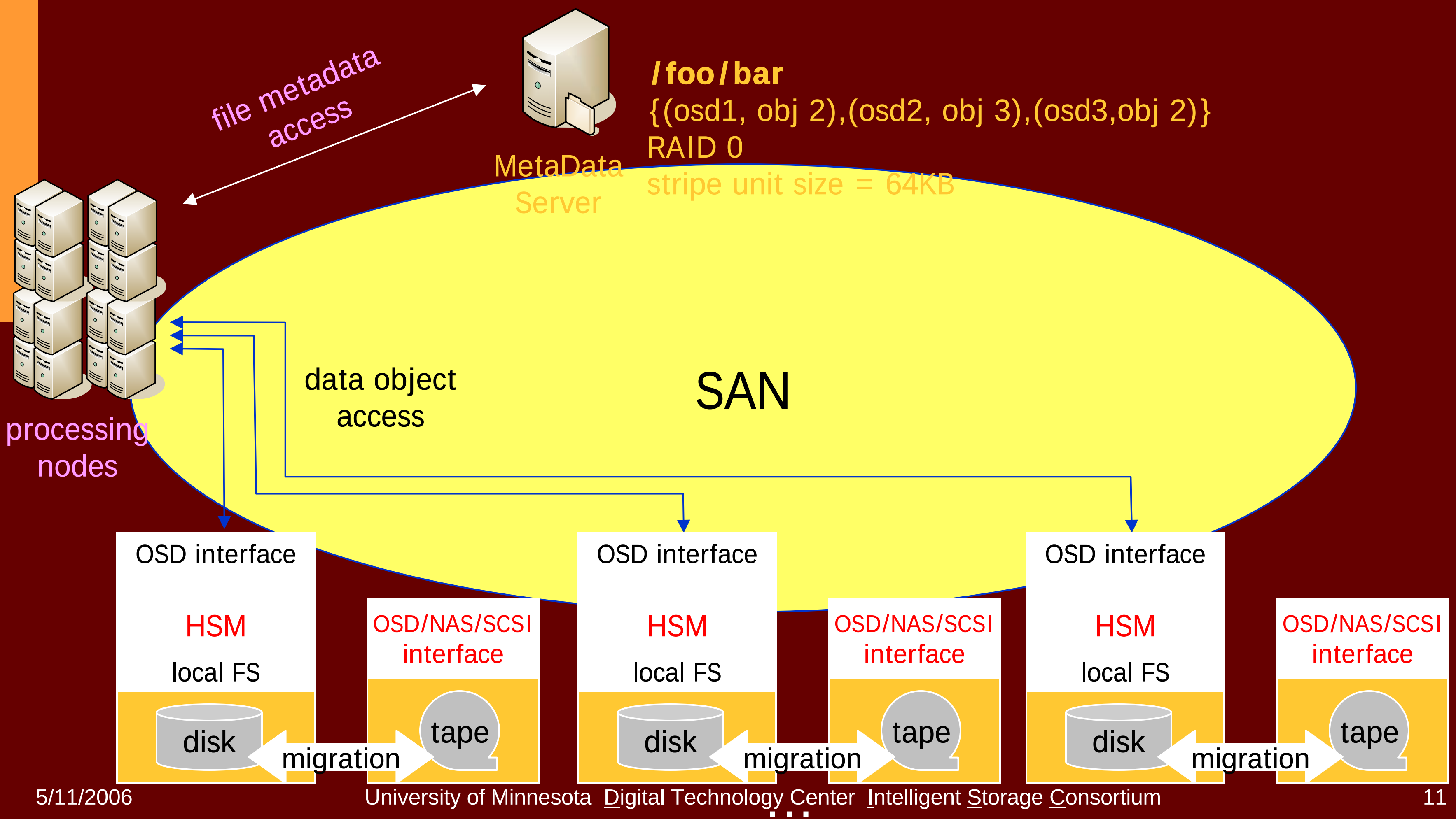
Objectives of Parallel Archive

- High data archive and restore throughput
- Scalable in archival bandwidth in addition to capacity
- Automated and transparent management of data migration in storage hierarch

Design Rationales

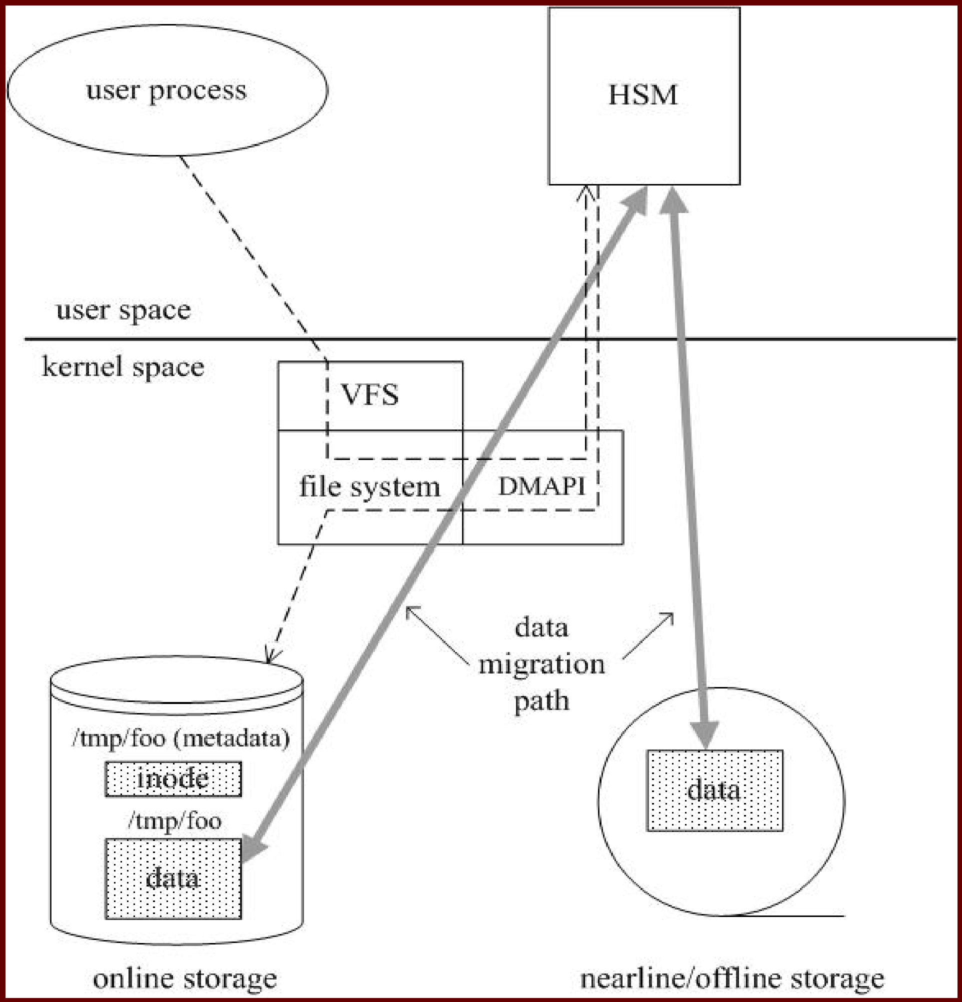
- Parallel archival storages
  - Explore aggregated parallel archival bandwidth
- Direct data migration between OSDs and their associated archival storages
  - OSDs are smart and powerful enough
- OSD embeds automated management of migrations
  - Policy based

Parallel Archive Architecture



Eliminating dependency on DMAPI/XDSM

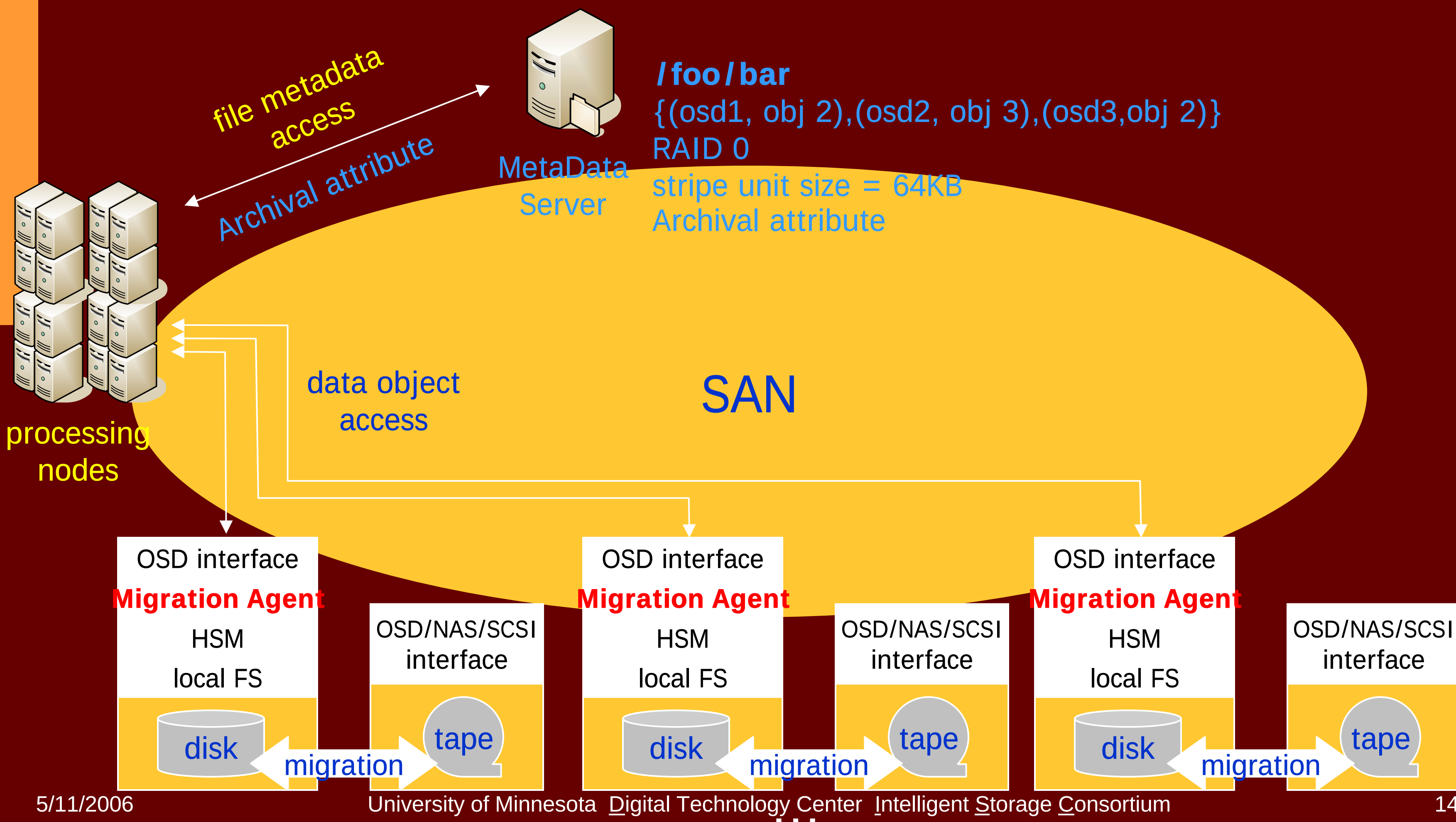
- Heavy kernel implementation
- Most functions are unused by HSM
- Not widely supported by popular file systems
- Not scalable from past experience



Functions of DMAPI Need to be Replaced

- Catching access events
  - Accessing objects not in online storage
- Transparent namespace
  - As if files are always there
  - File stub is always kept in the FS managed by HSM

Replacing DMAPI/XDSM



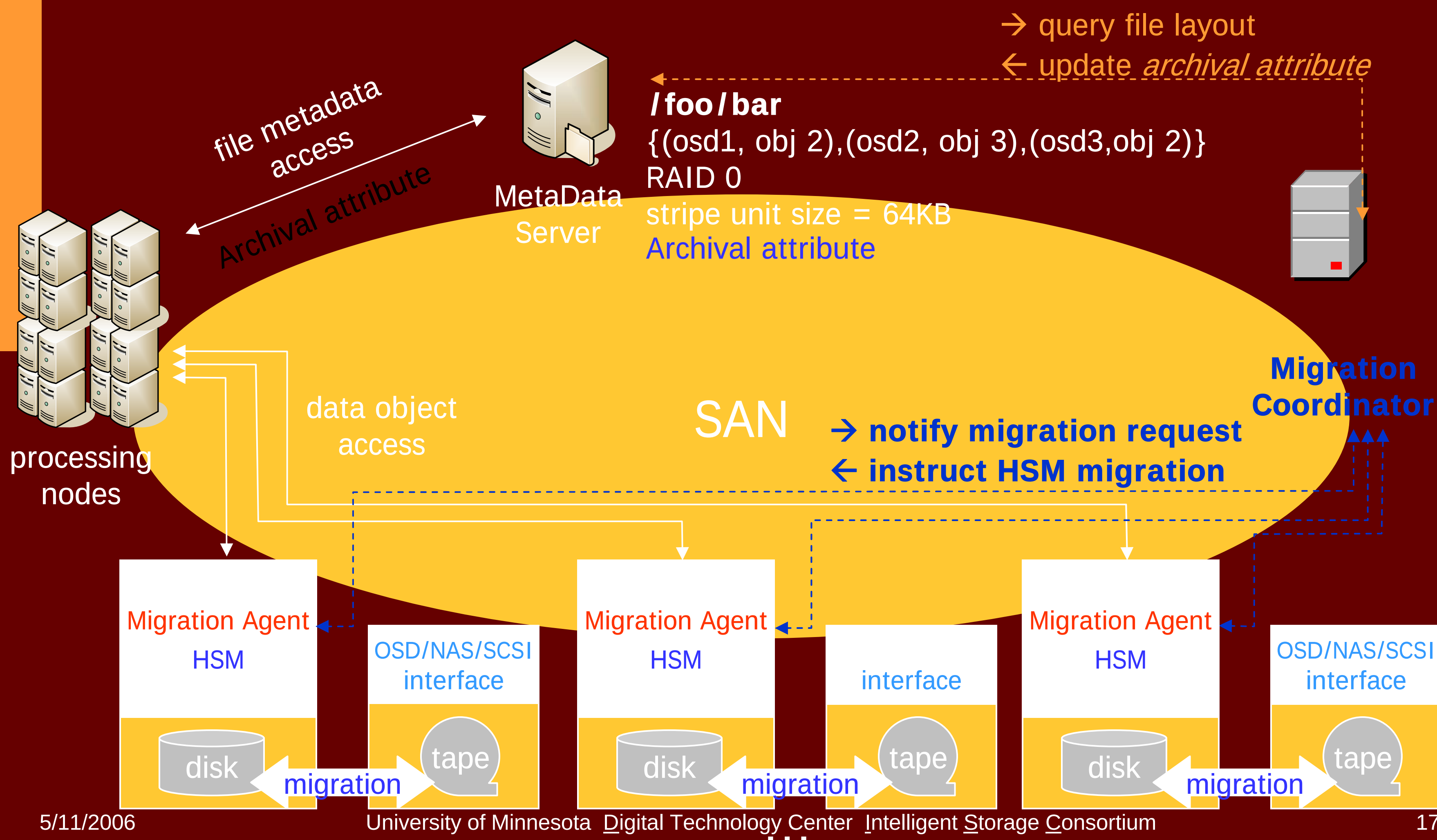
Local HSMs Need to be Coordinated

- Striping of single large files on multiple OSDs
  - Terabyte files are common in HPC
- File sets of many related files on multiple OSDs
  - Used by the same application
- Accessing of archival storage are typically sequential
- "Synchronous" migrations between multiple pairs of OSD and archival storage
  - True high aggregated migration bandwidth for single file or file set

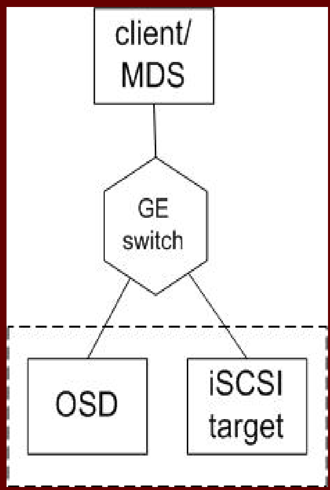
Design Rationales

- Separated migration control path and migration data path
  – OSDs do not involve in complicated migration coordinating task
- Centralized coordinating authority
  – Possible for intelligent decisions across requests

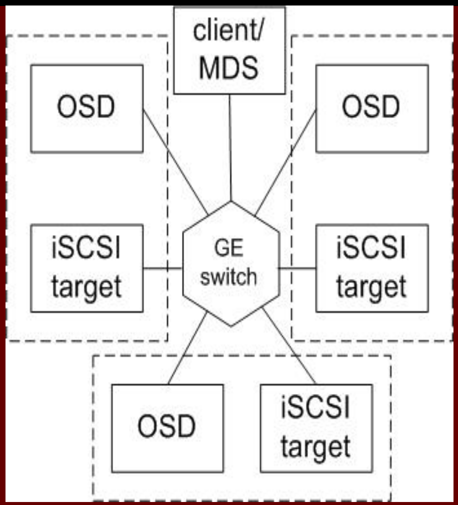
Coordinating Parallel HSM



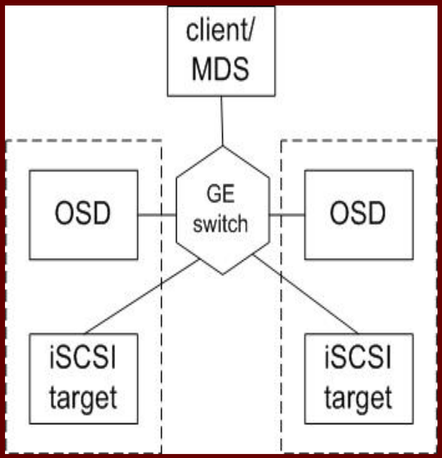
Experiment setup



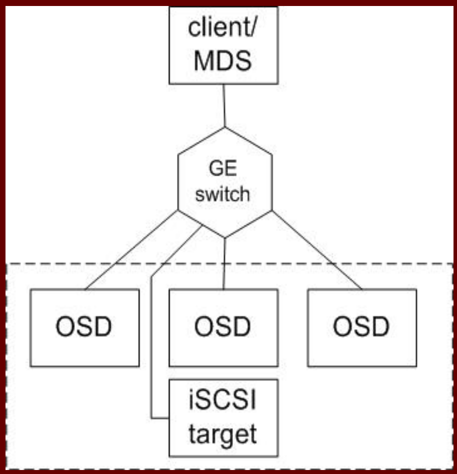
single-pair



triple-pair



dual-pair



single-backup

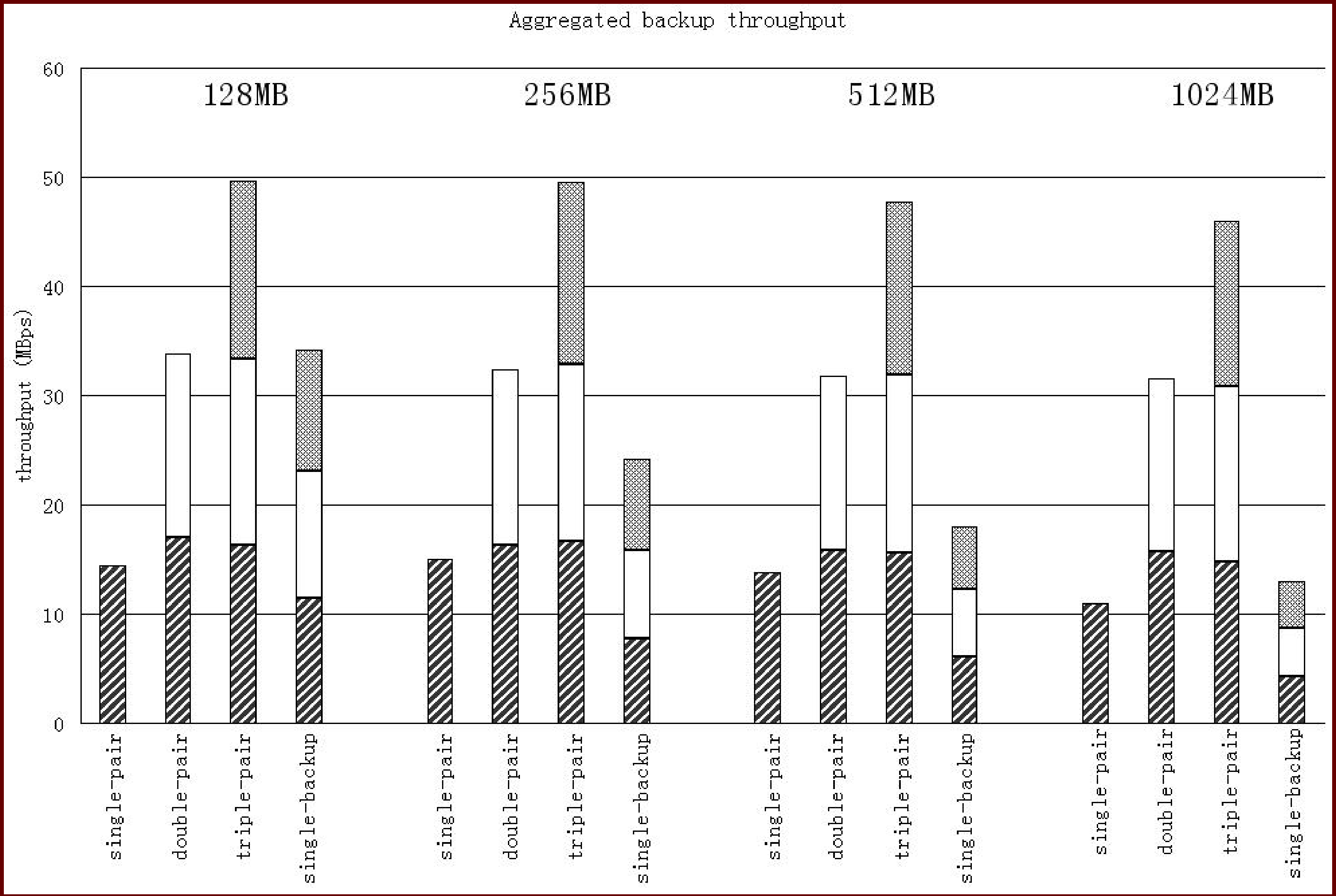
OSD host configurations

CPU	Two Intel XEON 2.0G
Memory	256MB DDR DIMM
SCSI interface	Ultra 160 SCSI (160MBps)
HDD speed	10,000RPM
Avg. seek time	4.7ms
NIC	Intel Pro/1000MF

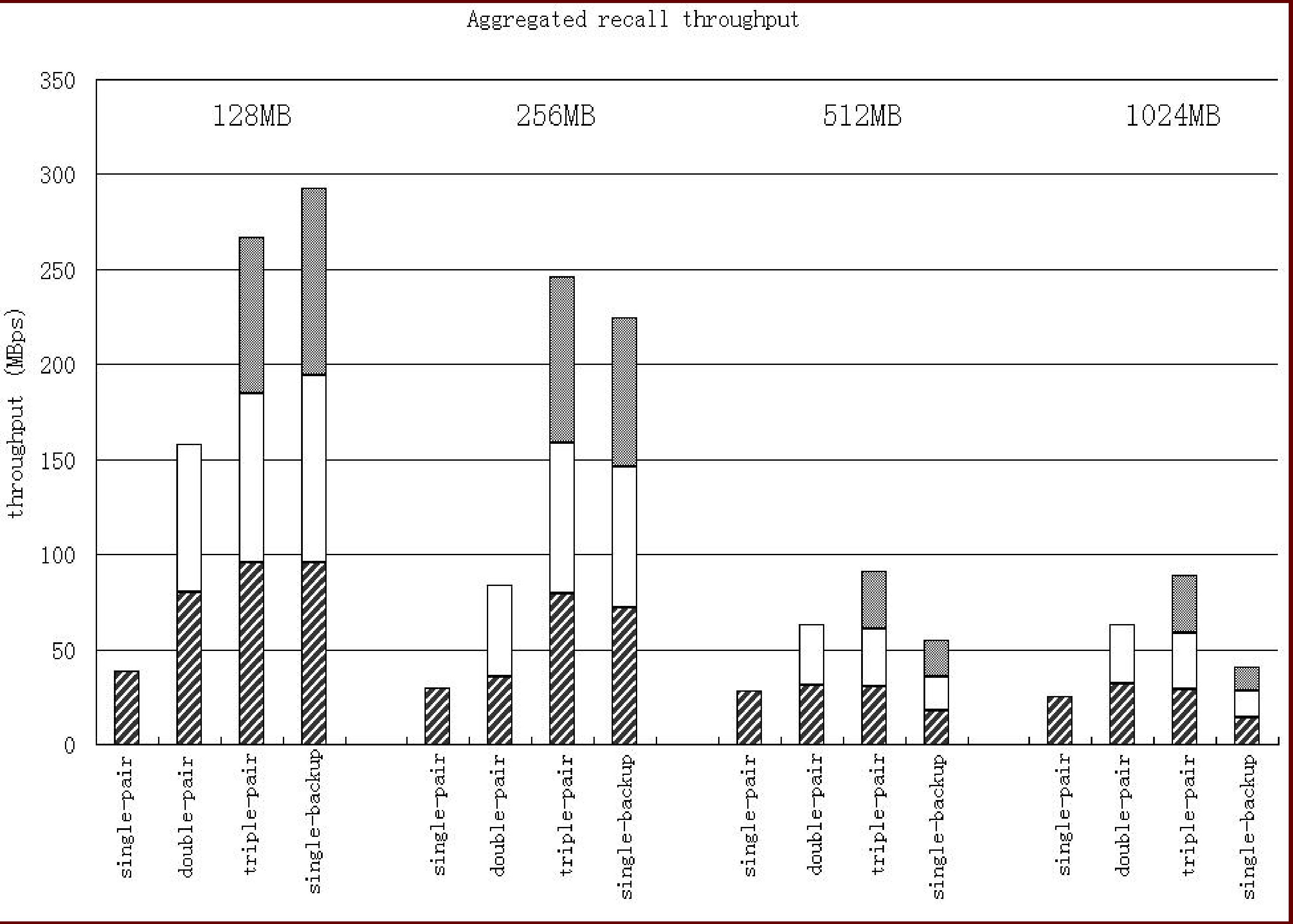
Target/MDS host configurations

CPU	4 Pentium III 500MHz
Memory	1GB EDO DIMM
SCSI interface	Ultra2/LVD SCSI (80MBps)
HDD speed	10,000RPM
Avg. seek time	5.2ms
NIC	Intel Pro/1000MF

Aggregate Backup Throughputs



Aggregate Restore Throughputs



Current Work: Intelligent Migration Decision

- Files in the same fileset have different access frequency
  - Less frequently-accessed file may lead the entire fileset to be archived
  - Similar thing could happen for striped files
- How to collect global fileset access information to make informed decision instead of always approve migration requests

Current Work: Metadata Server Supports

- Adding fileset supports
- Adding efficient object id to file/fileset mapping function
  - In current Lustre prototyping, file system namespace is traversed to resolve the query
  - Create special directories or database to keep direct mapping

Future Work: OSD-based Tape Library

- Self-contained object tape cartridges
  - Object interface
  - Metadata on tape or cartridge NVRAM
  - Object attribute layout on tape media
- Tape library management
  - Add/remove self-contained cartridges
  - Maintaining mapping of object id to cartridge id
  - Scheduling of archiving/restore requests

Future Work: Extension to pNFS

- pNFS has similar structure
  - Client, metadata server and storage servers
- Heterogeneous storage servers
  - Supporting block, file (NAS) and object storage servers
  - File and object storage servers can easily use proposed solution
  - Block storage servers needs additional help
  - How to provide a generic solution for all kind of storage servers

HPTFS: High Performance Tape File System

Joint work with

Jim Hughes, Ravi Kavuri

Sun Microsystems

Background

- Huge capacity: tape capacity is doubling every two years
  - One tape reaches the capacity of 500 GB native data (15 TB just several years away)
- Relatively high streaming rate: tape drive speed is increasing
  - 120MB/s native data transfer rate from Sun Microsystems (T10000 enterprise tape drive)
- Tape storage has the advantage of low cost per storage unit, off-site portability and less power consumption

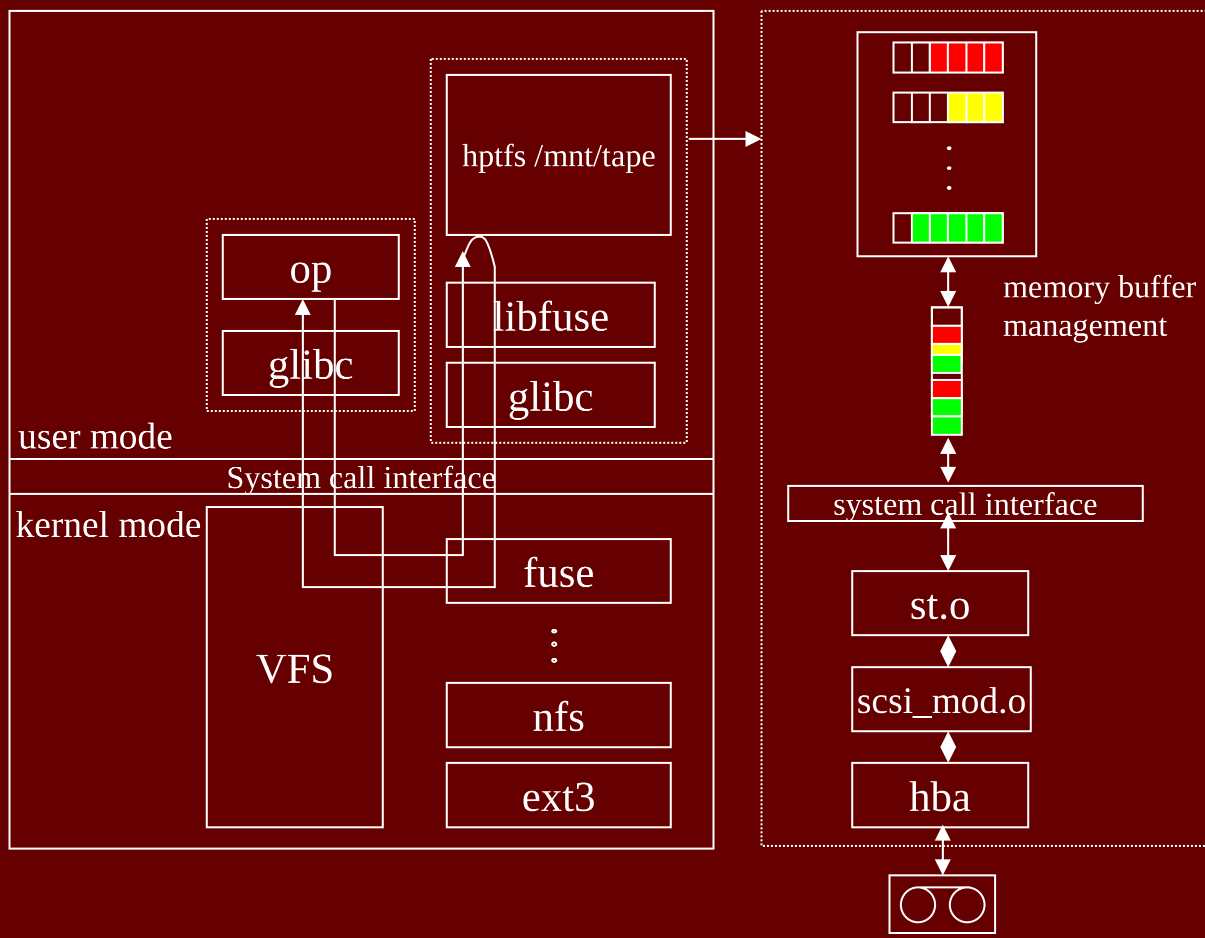
Motivation

- To avoid disasters and terror attacks, critical data needs to be backed up to tapes and sent to off-site
- Reducing the time to move massive data from disk to tape is critical for the data safety and the overall performance of HPC
- Easy to use I/O interface is key to the success and survival of tape storage

System Design Goals

- Tape is provided as a normal storage device with generic file system interfaces
- Direct backup/archive to the final destination – tapes – with streaming speed and without involving disk caching
- Provide “infinite” storage and support tape drive sharing without expensive backup software

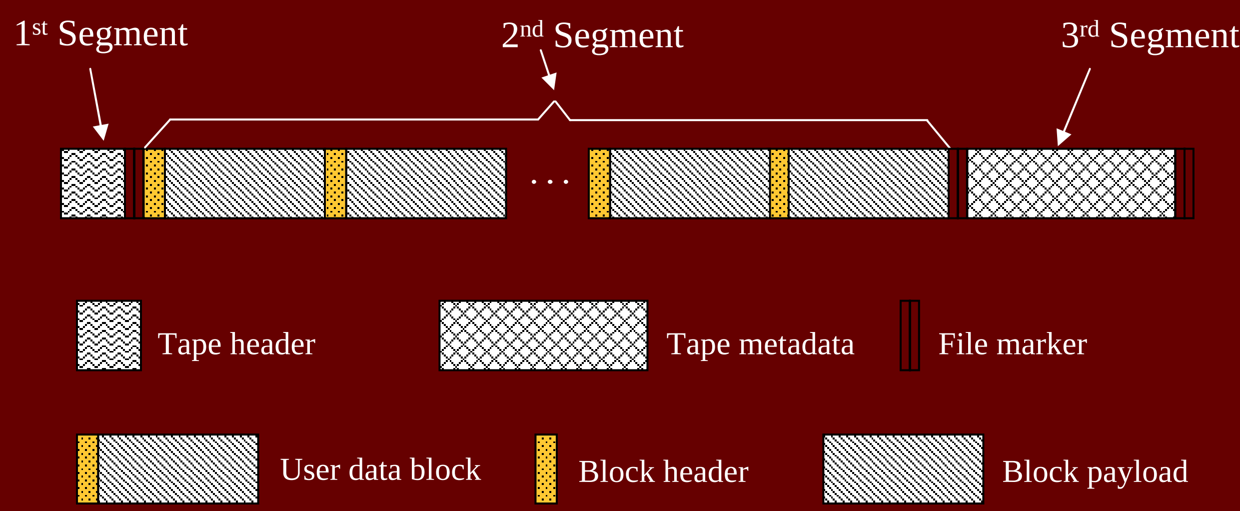
System Architecture



Objects Residing on Tape

- Tape data is self-contained and light-weighted
- User data and metadata
  - Each tape maintains three data segments: tape header, user data and metadata
  - Metadata contains object id, start position and end position
  - Metadata can be stored at the end of a tape or in tape cartridge embedded memory chip

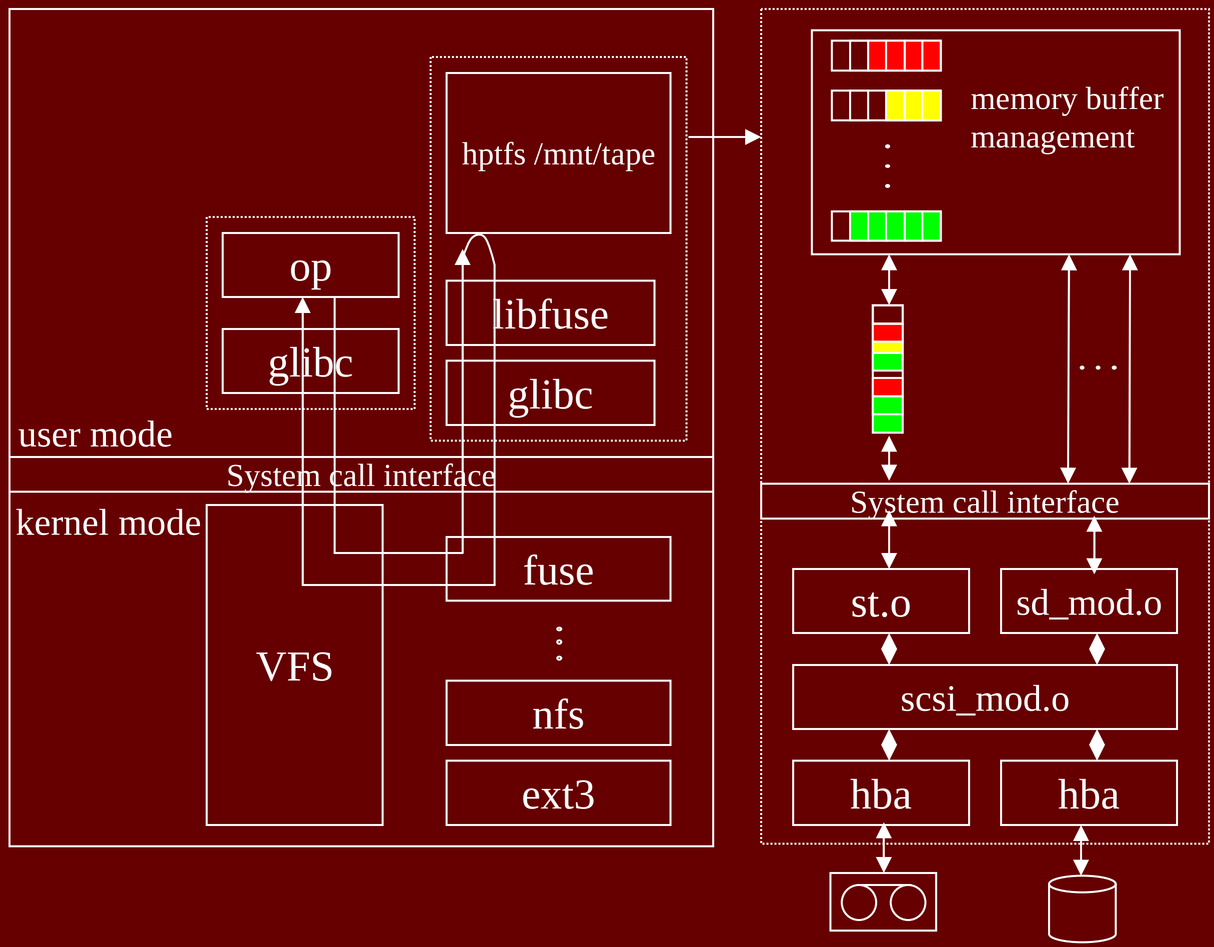
Tape Data Layout & Structure




```
struct objid
{
    int vol;
    int f_no;
    int b_sp;
    int seq;
};

struct tapemeta
{
    char name[1024];
    int f_no;
    int b_sp;
    int b_ep;
    struct objid id;
    struct stat stbuf;
    struct tapemeta *next;
};
```

Write to Tape & Disk Simultaneously



Usage of HPTFS

Commands and outputs	Notes
[root@oak lib]#./HPTFS /mnt/tape /home/xzhang/tape w	Mount tape in write mode at /mnt/tape
[root@oak lib]# ls -lt *.c -rw-r–r– 1 root root 61725 Jun 2 04:50 fuse.c -rw-r–r– 1 root root 12461 Jun 2 04:50 helper.c -rw-r–r– 1 root root 5064 Mar 21 05:37 fuse_mt.c -rw-r–r– 1 root root 3045 Feb 2 2005 mount.c	List all C files under current folder (on disk)
[root@oak lib]# cp *.c /mnt/tape	Copy all C files from disk to tape
[root@oak lib]#fusermount -u /mnt/tape	Write out metadata to tape and umount tape
[root@oak lib]#./HPTFS /mnt/tape /home/xzhang/tape r	Mount tape in read mode at /mnt/tape
[root@oak lib]#ls -lt /mnt/tape -rw-r–r– 1 root root 61725 Aug 15 23:55 fuse.c -rw-r–r– 1 root root 5064 Aug 15 23:55 fuse_mt.c -rw-r–r– 1 root root 12461 Aug 15 23:55 helper.c -rw-r–r– 1 root root 3045 Aug 15 23:55 mount.c	List all C files on tape media

Performance Evaluation

Main observations:

- User applications directly write/read data to/from tape without the knowledge of tape storage
- Support concurrent writes nicely
- Stream tape drive if enough data are provided

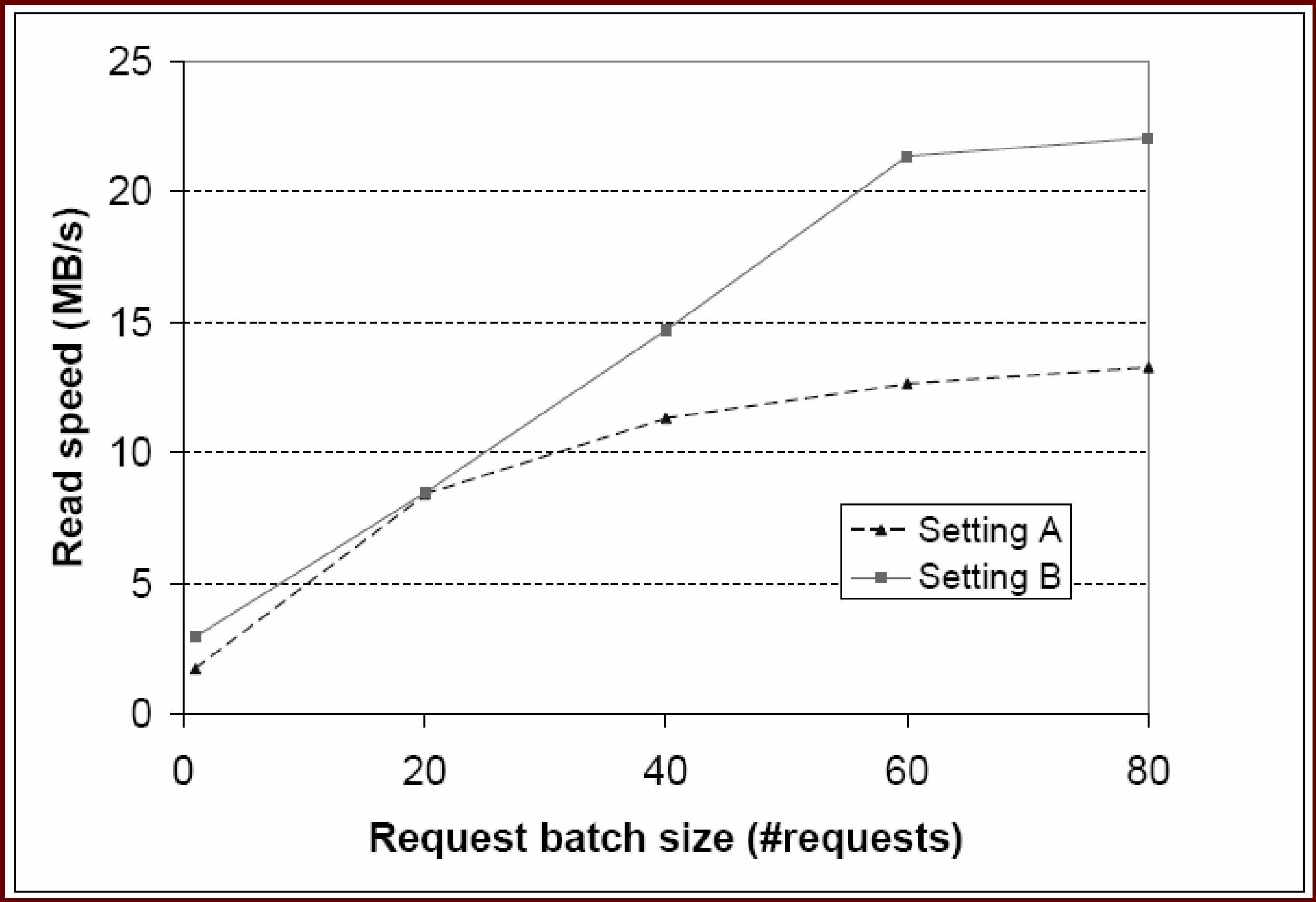
Part of the Performance Results

Table 2. Tape write performance (MB/s, tape block size=256KB)

Degree of concurrency	Setting A rate		Setting B rate	
	Mean	Stdv	Mean	Stdv
2	24.148	0.433	37.709	0.004
3	24.222	0.392	37.713	0.005
4	24.169	0.373	37.719	0.005

Note: write speeds of Setting A and B are rated as 29.759 MB/s and 37.604MB/s respectively

Tape Random Read Performance with PostMark (1,000 files and 100 read operations)



HPTF-Summary

- HPTFS provides generic file system interface for tape data access: writing to tape is as easy as writing to disk
- Provides tape drive sharing with high performance
- Built over HPTFS, software for backup and HSM can be made simpler
- Potential to embed HPTFS functionality into tape drive totally changing tape access paradigm

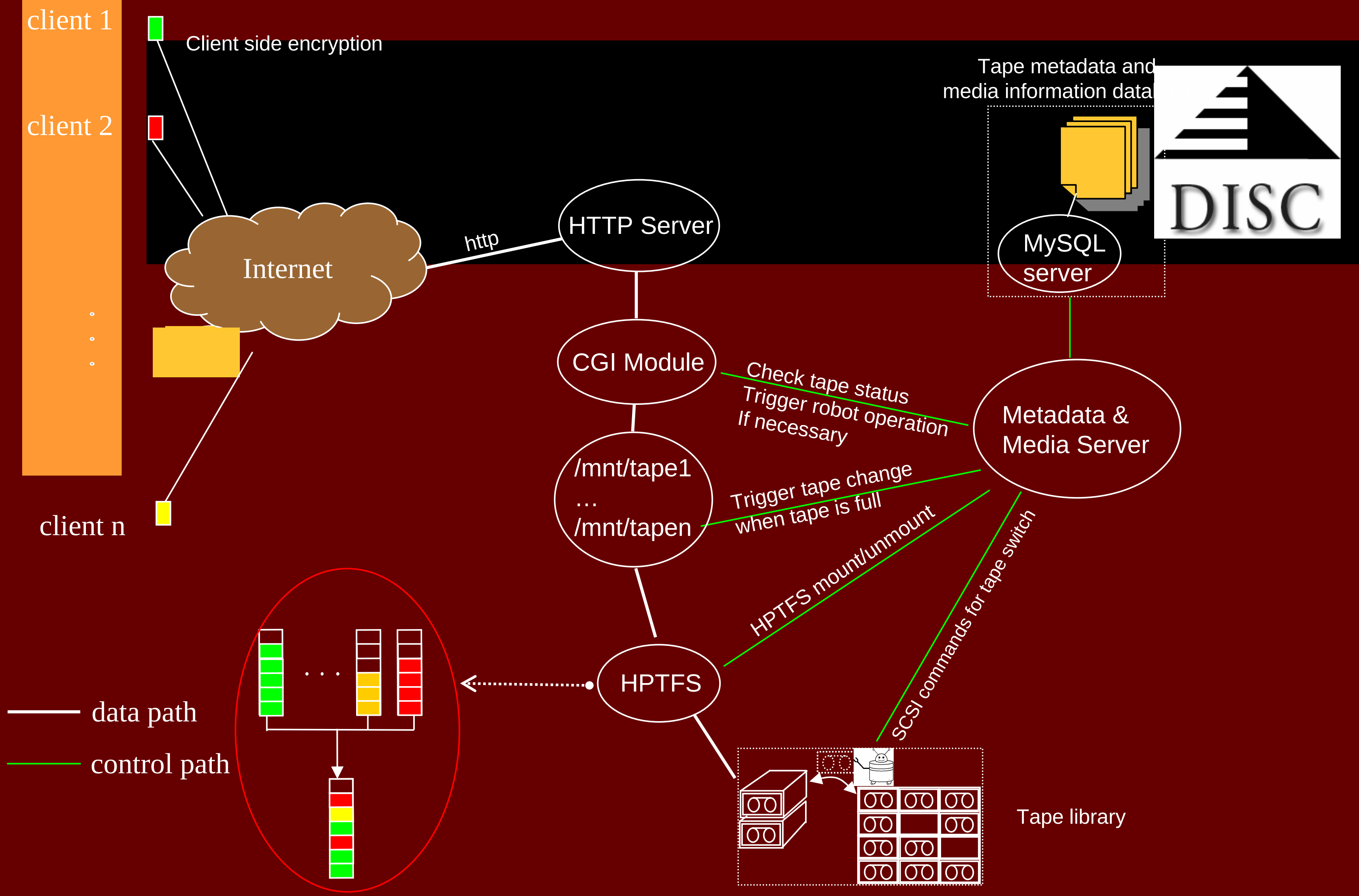
Tape Storage based High Performance Internet Backup/Archive System

Another Application of HPTF

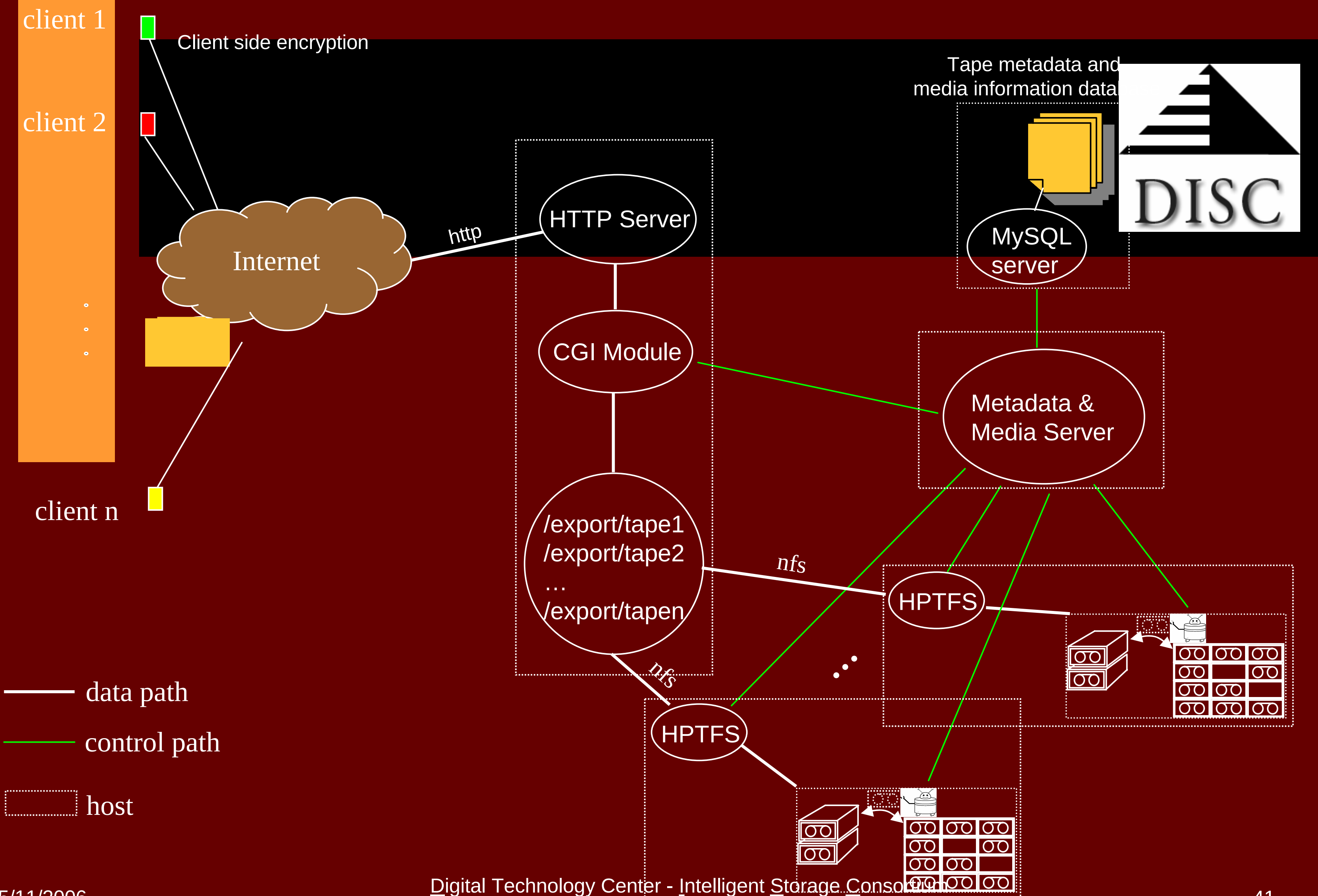
Motivation

- Personal user data is not well protected as business user data
- Personal user needs a low-cost data protection solution with a predictable lifetime and a proven success history
  - P2P storage solution does not provide these required characteristics
- Data recovery happen much less frequently than data backup/archive
- High speed internet access is ubiquitous

Internet Backup/Archive



How It Scales



Conclusions

- It is still very challenging to provide high-performance archive/backup in HEC environment.
- User convenience and transparency are paramount.
- Data managing and preserving over long-term are difficult.

Contact Information: du@cs.umn.edu or ddu@nsf.gov