

Module 1: HPSS Overview

Contents

- A. Introduction**
- B. HSM Concepts**
- C. Storage Allocation Concepts**
- D. HPSS Architecture**
 - 1. Infrastructure Components**
 - 2. HPSS Server Components**
 - 3. Storage Subsystems**
 - 4. Client Interfaces**



Module 1: HPSS Overview

Objectives:

Upon completion of Module 1, you should be able to

- Understand fundamental attributes of HPSS as a storage solution
- Demonstrate understanding of hierarchical storage management
- Demonstrate understanding of HPSS storage allocation
- Identify HPSS software components and their functions
- Identify HPSS infrastructure components and their functions
- Identify HPSS user interfaces and their characteristics
- Demonstrate understanding of HPSS storage subsystems

Introduction

- Introduction**
- HSM Concepts**
- Storage Allocation Concepts**
- HPSS Architecture**
 - Infrastructure components
 - HPSS server components
 - Storage Subsystems
 - Client Interfaces

Introduction

- **What is HPSS?**

- Hierarchical storage manager (HSM)
- Scalable
- Parallel
- Network-centric (see next slide)
- Focus on high-performance access to data
- Utilize parallel data transfer streams and/or striping across multiple servers to maximize performance



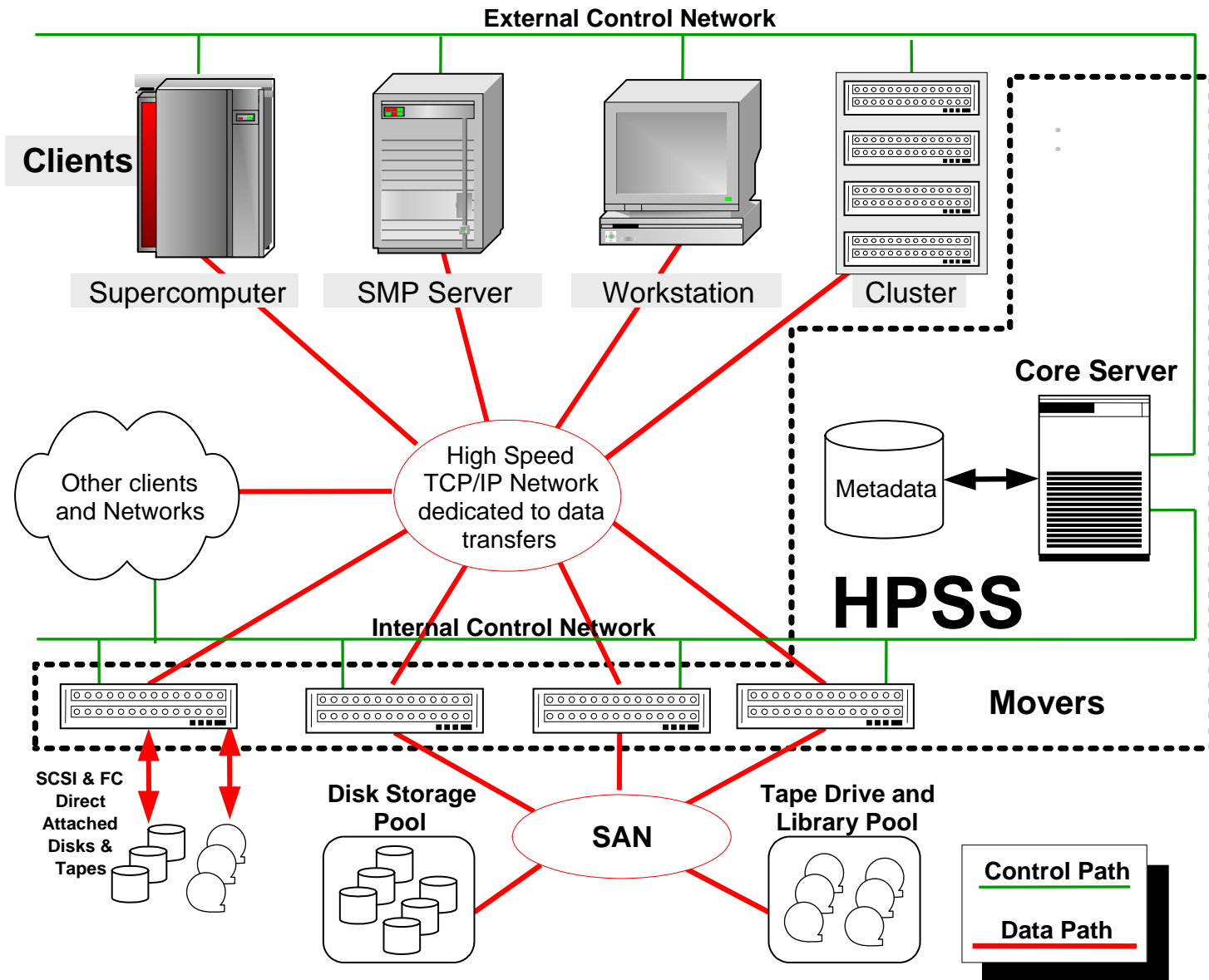
- **Why HPSS?**

- Requirements for fast per file and aggregate data transfers
- Requirements for large data stores
- Requirements for large single logical name spaces
- Requirements for distributed, secure access to data

- **Availability**

- Initially released on 7/1/96
- Available from IBM Global Services - Houston as a service offering

Introduction



Introduction

- **How do users access HPSS?**

- Standard FTP
 - **Allows convenient access from any client with a standard FTP client**
- HPSS Parallel FTP (pftp_client)
 - **An enhanced version of the standard FTP client with significantly improved performance**
- HPSS Virtual File System (VFS) Interface Client
 - **Allows user to access HPSS as a Linux file system**
- General Parallel File System (GPFS)/HPSS Interface (will be 6.2 add-on)
 - **Provides GPFS users with the view of an infinite file system by migrating older files to HPSS and staging them to GPFS on demand**
- Other HPSS Client API Applications
 - **HSI**
 - **Provides Unix-style command interface in interactive, batch, and command modes**
 - **Available from Gleicher Enterprises, Inc.**
 - **HTAR**
 - **Bundles small files into large files and store into HPSS**
 - **Available from Gleicher Enterprises, Inc.**
 - **GridFTP enabled for HPSS**
 - **An extension of the standard FTP protocol for the GRID Computing environment**
 - **Available from ANL**
 - **Network File System (NFS V4)**
 - **Allows user to access HPSS as a native file system**
 - **Available from CEA-DAM**
 - **Site-specific**

Introduction

- **HPSS Information Resources**

- **Published Documentation**

- **HPSS Installation Guide**
- **HPSS Management Guide**
- **HPSS Error Messages Reference Manual**
- **HPSS User's Guide**

- **Extensive set of web pages:**

<http://www.hpss-collaboration.org/hpss>

- **Product information**
- **Support Policy**
- **Defect / Feature tracking and submission**
- **Release information, support information, frequently asked questions, etc.**
- **Published documentation available for download in PDF and formats**



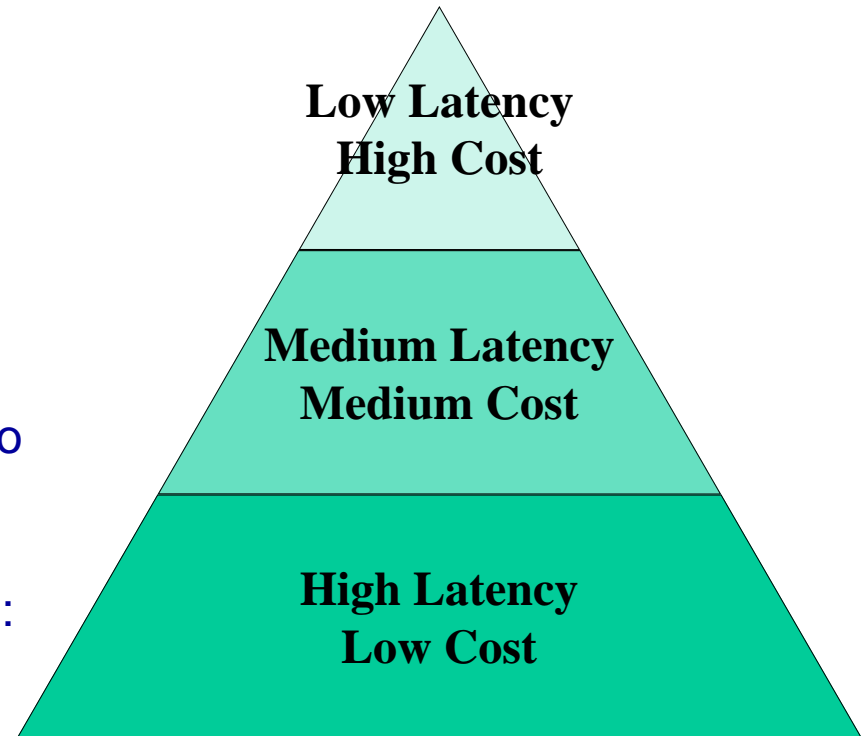
HSM Concepts

- Introduction
- HSM Concepts**
- Storage Allocation Concepts
- HPSS Architecture
 - Infrastructure components
 - HPSS server components
 - Storage Subsystems
 - Client Interfaces

HSM Concepts

- **Hierarchical Storage Management**

- Data management strategy that seeks to cost-effectively store large quantities of data by storing each datum on the lowest cost media available that will meet user requirements
- Implementations usually store the most recently accessed data on low latency, high cost storage while moving less recently accessed data to progressively higher latency, lower cost storage
- HPSS implements this strategy using:
 - **Storage classes**
 - **Storage hierarchies**
 - **Classes of service**
 - **Migration, purge, & stage**

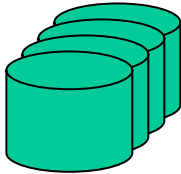


HSM Concepts

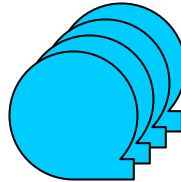
- **Storage Class**

- A logical grouping of similar storage media intended to store HPSS files with similar characteristics (i.e., file size, access frequency, tape technology, etc.)
- Examples:

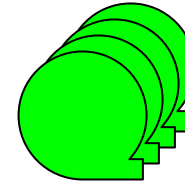
Disk – Large File



LTO – Huge File



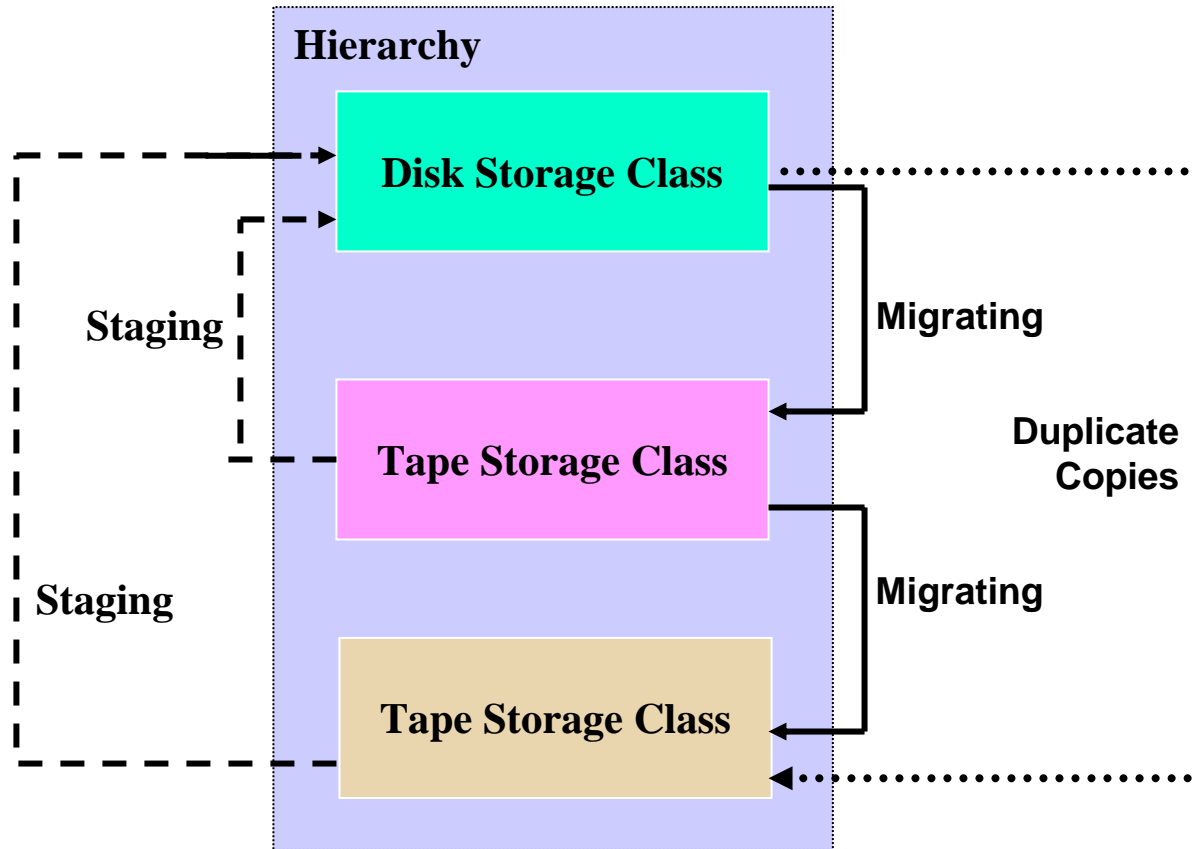
STK 9940B – Deep Archive



- **Storage Hierarchy**

- Consists of up to 5 levels, each of which is a separate Storage Class
- New files are stored at the top of the hierarchy (level 1)
- Files are copied down the hierarchy via **migrate** operations
- After migration to a lower level, file data may be deleted from a storage class via **purge** operations
- Files are copied up the hierarchy via **stage** operations, usually when accessed after being purged from the top level of the hierarchy

HSM Concepts



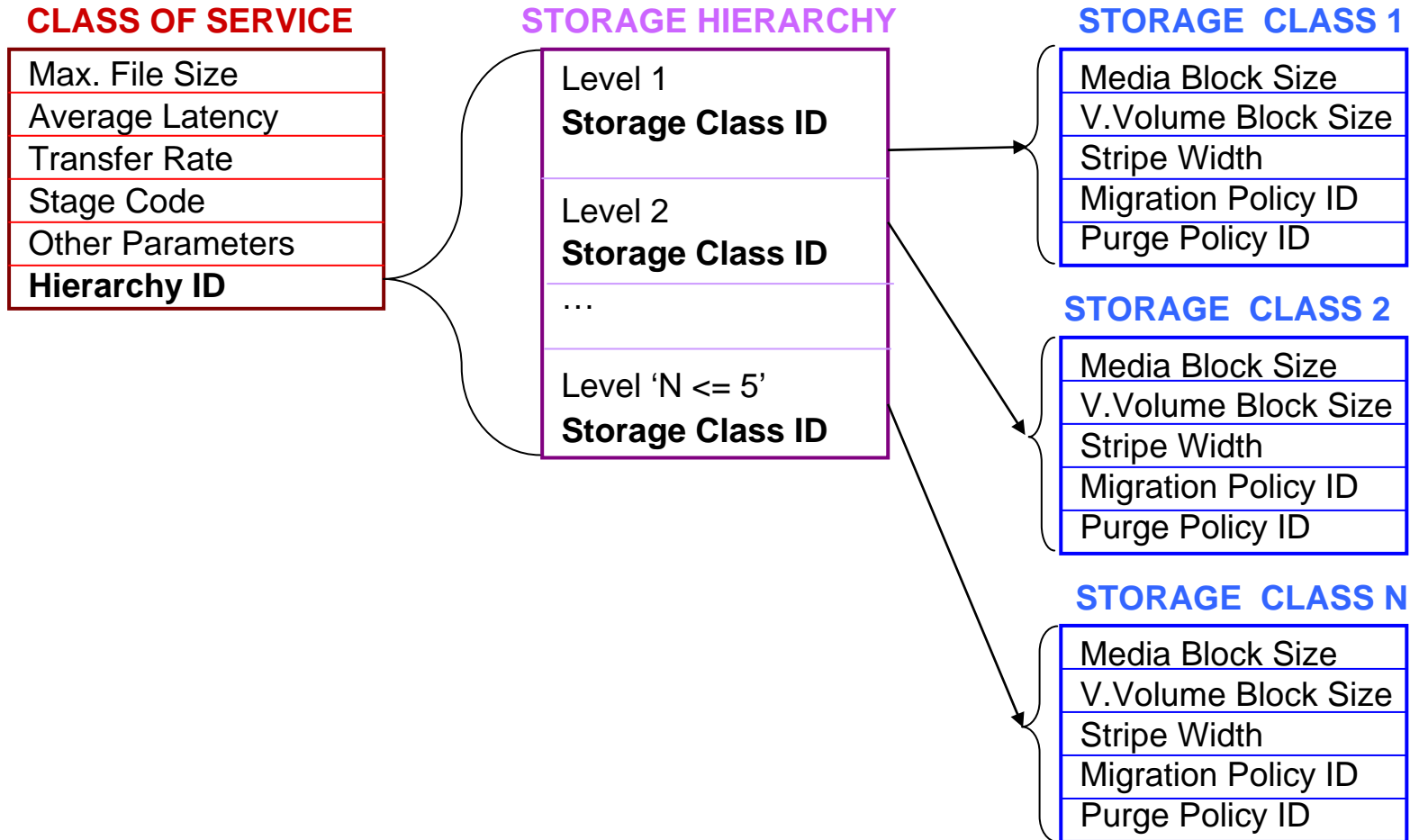
Migrate and Stage Operations

Example: Data is migrated down the hierarchy from disk to tape and tape to tape. Data is staged up the hierarchy from tape to disk at the top. Duplicate copies can be made by migrating to two different target levels from the same source level. This ensures that two extra copies exist before purging.

HSM Concepts

- **Migration, purge, and stage**
 - Migration, purge, and stage operations are based on policies, usage patterns, and storage availability
 - Migration and purge policies are associated with storage classes
 - Stage behavior is determined by the class of service
- **Class of Service (COS)**
 - Associates a staging code, file size range, and other performance-related characteristics with a hierarchy
 - A given hierarchy may be associated with multiple COS, but each COS is associated with exactly one hierarchy
 - All HPSS files are assigned to a COS when created
 - Some HPSS interfaces allow the end user to specify the desired COS

HSM Concepts



Relationship of Class of Service, Storage Hierarchy, and Storage Class

Storage Allocation Concepts

- Introduction
- HSM Concepts
- Storage Allocation Concepts**
- HPSS Architecture
 - Infrastructure components
 - HPSS server components
 - Storage Subsystems
 - Client Interfaces

Storage Allocation Concepts

- **Physical Volume**

- Storage media on which HPSS stores data
- For disk, this is an OS-level raw device (e.g., /dev/rhdisk5)
- For tape, this is a cartridge (e.g., A00950)

- **Data Striping**

- Distributing sequential data across multiple volumes
- Used to improve performance by taking advantage of hardware parallelism

- **Virtual Volume**

- Striped group of 1 or more Physical Volumes
- StripeWidth=1 is equivalent to no striping, but VVs exist nevertheless
- Each Virtual Volume is assigned to a Storage Class

- **Virtual Volume Block Size**

- Amount of data written to each PV in a stripe before switching to the next PV

- **Stripe Width**

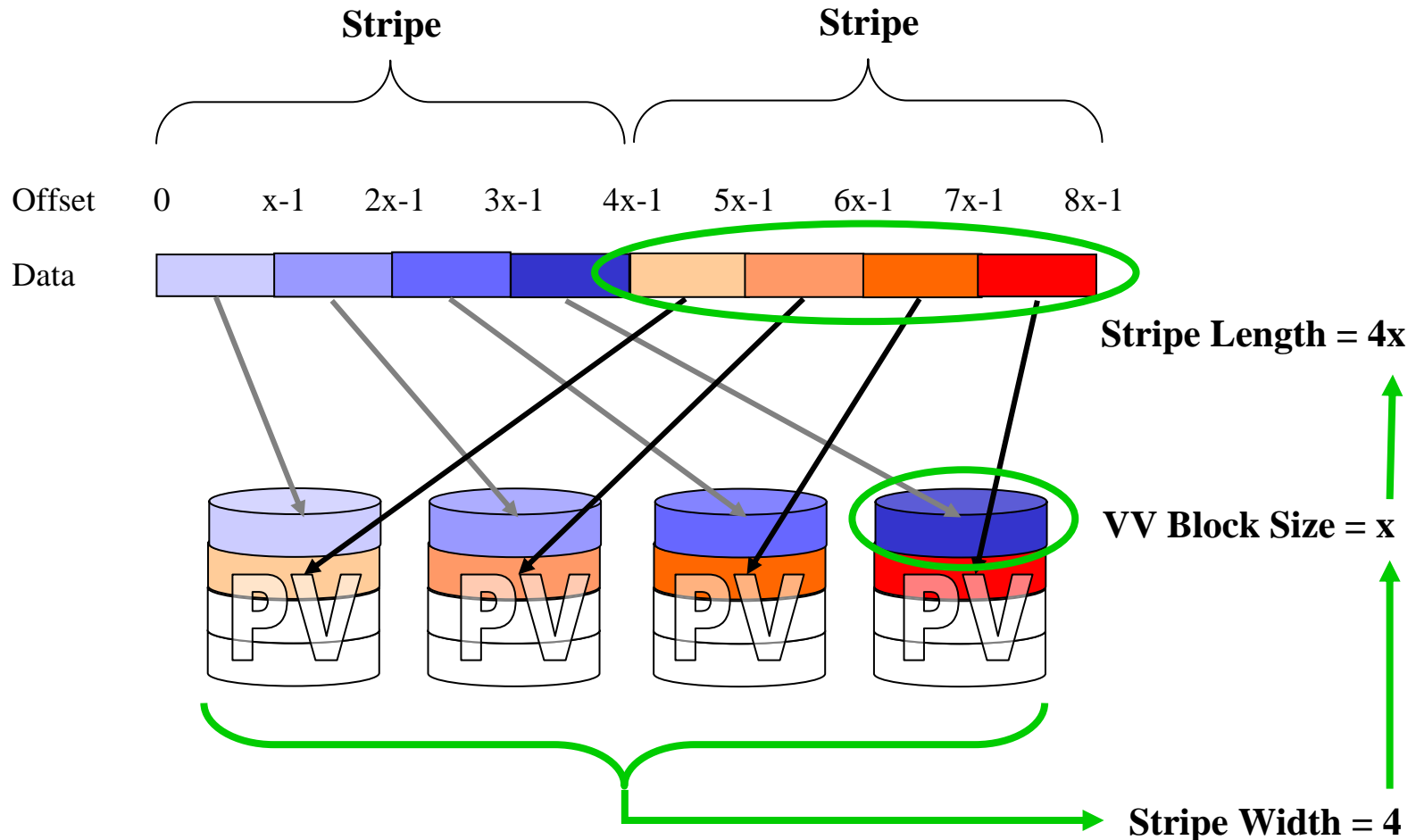
- Number of Physical Volumes grouped together to form a Virtual Volume

- **Stripe Length**

- Number of bytes written to span all Physical Volumes of a Virtual Volume
- Equals Stripe Width times the Virtual Volume Block Size

Storage Allocation Concepts

- Physical layout of a VV in a 4-way disk storage class:



Storage Allocation Concepts

- **Bitfile**
 - Logical sequence of bits that constitutes the data for a single file
- **Storage Map**
 - Describes usage of storage space for a VV
 - **Disk Storage Map** contains configuration information including the **Cluster Length, Number of Clusters, Usable Length, Free Space and Number of Segments**
 - **Tape Storage Map** contains the **Estimated Size and Space Left on the tape, and the Number of Segments**
- **Cluster**
 - The smallest unit of allocation for a disk storage map
 - The size of the cluster is determined when the disk VV is created
 - Size is equivalent to the minimum storage segment size

Storage Allocation Concepts

- **Storage Segment**

- Space allocated from a storage class to a file using a storage map

- Disk

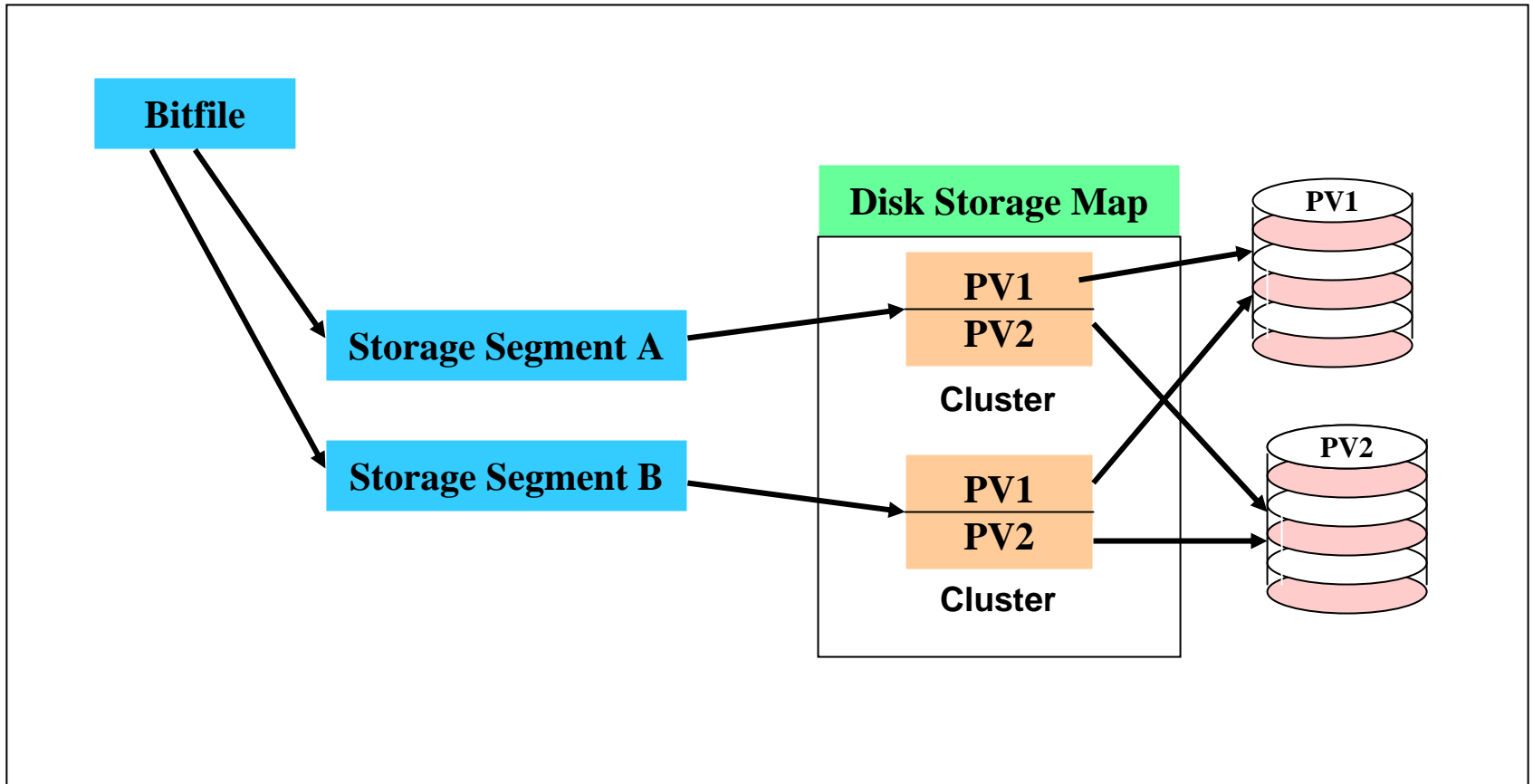
- **A disk storage segment is a contiguous block of disk space occupying one or more clusters. Segments don't come from maps, they are on disk. The maps are only an aid to figuring out where to put the segments**
- **The disk storage segment size for a file in a given SC is fixed**

- Tape

- **Tape storage segments are created with an indefinite length, but a fixed starting location on the tape**

Storage Allocation Concepts

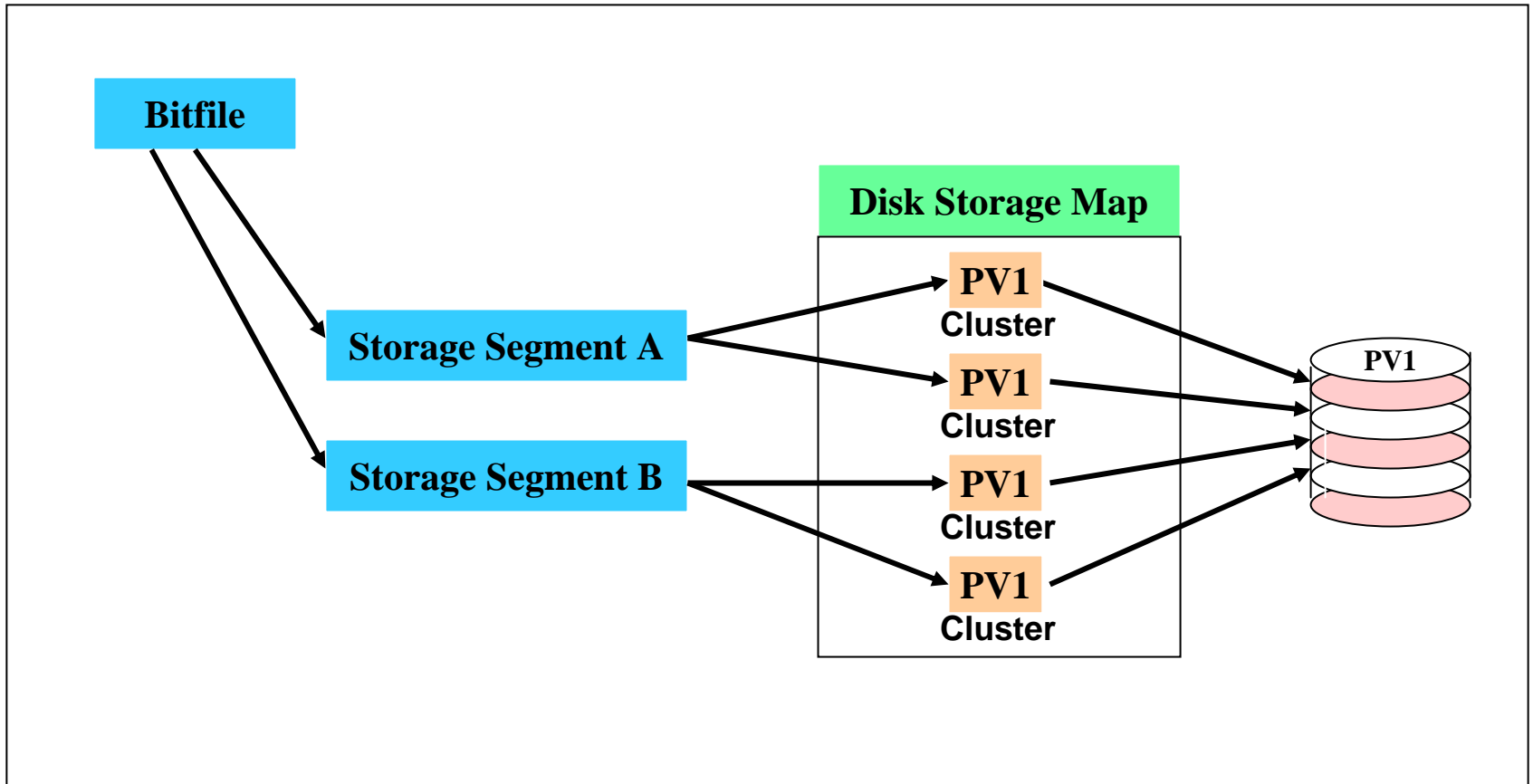
Two Way Stripe
Virtual Volume



Relationship between Bitfiles, Segments, Maps, VVs, and PVs

Storage Allocation Concepts

One Way Stripe
Virtual Volume



Relationship between Bitfiles, Segments, Maps, VVs, and PVs

Storage Allocation Concepts

- **Sparse Tapes**

- Over time, some tapes become sparse (the tape holds far less active data than its capacity; there is a lot of inactive data)
- Tape becomes sparse because we can only append to tapes
- When tape segments are “deleted” or become inactive, this segment space cannot be reused
- To address this issue, HPSS uses repack to move segments off sparse tapes and then reclaim the newly empty tapes for reuse.

- **Repack**

- Moves segments from EOMed VVs to free VVs in the same storage class
- Accessible via SSM GUI and command line utility
- Can automatically select candidate VVs for a specified storage class

- **Reclaim**

- Resets the state of tape VVs so they can be reused
- Accessible via SSM GUI and command line utility
- Can automatically identify candidate VVs

- **Retire**

- When media needs to be removed from HPSS, it can be ‘retired’
- This prevents new storage segments from being allocated
- Volume can be removed after it becomes empty via repack or attrition

Storage Allocation Concepts

- **Shelf Tape**

- HPSS tapes with data that is unlikely to be needed soon can be removed from robotic management using the 'shelf_tape' utility
- If shelved tapes are later required to satisfy a request, a notification will be sent to SSM to notify an operator or admin that the tapes are needed
- When selecting tapes to be shelved, the utility can select tapes based on input parameters or simply shelve a specified set of tapes
- In either case, the cartridges must be dismounted and either EOMed or retired
- When using automatic selection, the user supplies:
 - **Minimum days since the last tape access**
 - **Maximum number of tapes to shelve**
 - **Storage class (optional)**
 - **An option indicating the primary selection criteria, one of:**
 - **Age of data**
 - **Last access time of data**
 - **Total number of data accesses**

Storage Allocation Concepts

- **Fileset**

- All name space objects are in filesets
- Every Core Server has a default root fileset
- All other filesets are disjoint directory subtrees, administered as a unit, that can be mounted in the global name space
- Similar to a UNIX file system

- **Junction**

- Name space object used to attach a fileset to a specified point in the HPSS name space
- Similar to a UNIX mount point

- **File Families**

- A Family Id may be associated with a fileset when the fileset is created
- Data for files in such a fileset will be isolated to tapes designated with the corresponding Family Id
- A segment's Family Id is preserved on all segment moves and copies, and across COS changes
- The SSM GUI can be used to manage the list of available file families
- How are tapes assigned to a family?
 - **The Core Server allocates free space on tapes already associated with the requested family if possible. Otherwise, it dynamically assigns tapes from the set of free tapes with no assigned family (family 0)**
 - **Reclaim resets the Family Id of tapes back to the default (family 0)**

Concepts Covered

- **HSM Storage Concepts**

- hierarchical storage management (HSM)
- storage class (SC)
- storage hierarchy (“hierarchy”)
- migration
- purge
- stage
- class of service

- **Storage Allocation Concepts**

- Physical Volume (PV)
- Virtual Volume (VV)
- VV Block Size
- Data striping
- Stripe Width
- Stripe Length

- Bitfile
- Storage Map
- Cluster
- Storage Segment

- Sparse Tape
- Repack
- Reclaim
- Shelf Tape

- Fileset
- Junction
- File Family

Exercise

- **Exercise (fill in the blank)**

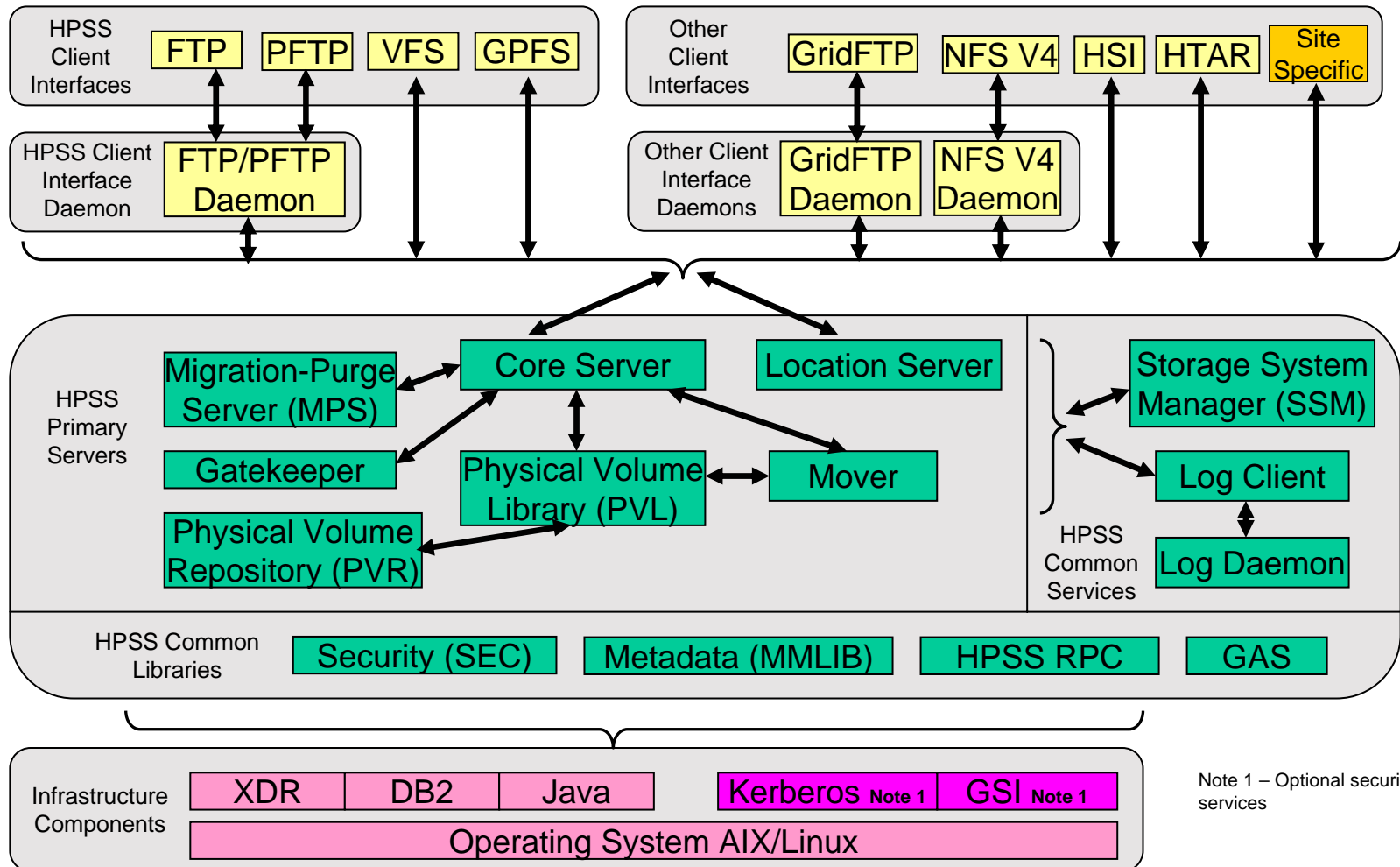
- A _____ is a logical ordering of storage classes, with the faster, more expensive storage at the top and the slower, cheaper storage at the bottom.
- A _____ tape has a large percentage of wasted space.
- _____ is used to copy data to a lower level in a hierarchy, but _____ is used to copy data to the highest level.
- A logical group of similar storage media is called a _____.
- Some interfaces allow a user to choose the _____ for their files, which determines the media used to store the file's data.
- Filesets can force files stored in them to use a particular class of service and/or be a member of a particular _____.
- After a file has migrated from disk, it will later be _____ to free up space in that disk storage class.
- A disk storage segment consists of one or more contiguous _____s allocated from a disk storage _____.
- After a tape VV has been repacked, it must be _____ed so that new data may be written to it.

HPSS Architecture

Introduction
HSM Concepts
Storage Allocation Concepts
HPSS Architecture
Infrastructure components
HPSS server components
Storage Subsystems
Client Interfaces

HPSS Architecture

- Shows interaction between HPSS components
- Communication occurs via remote procedure calls (RPCs)

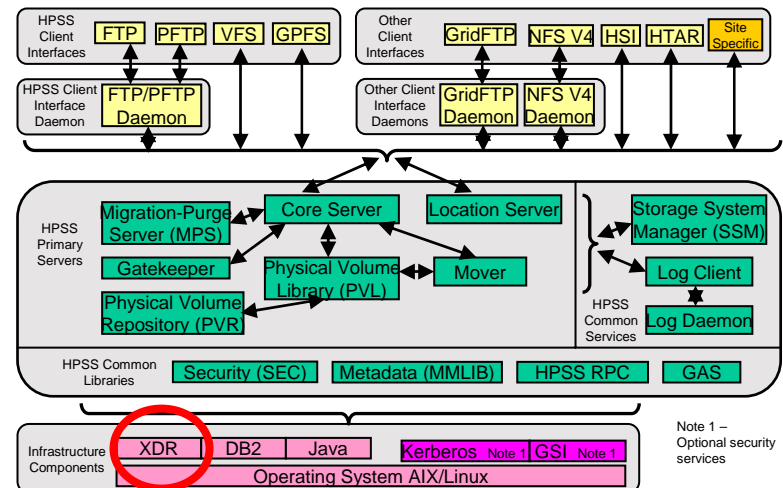


HPSS Architecture

Infrastructure Components

- **eXternal Data Representation (XDR)**
 - Creates portable data by translating its local data representation using the XDR standard representation such that any program from any machines can read it
 - HPSS RPC is built on XDR over TCP/IP
 - HPSS uses XDR to marshal HPSS IOD and IOR

Replaced DCE



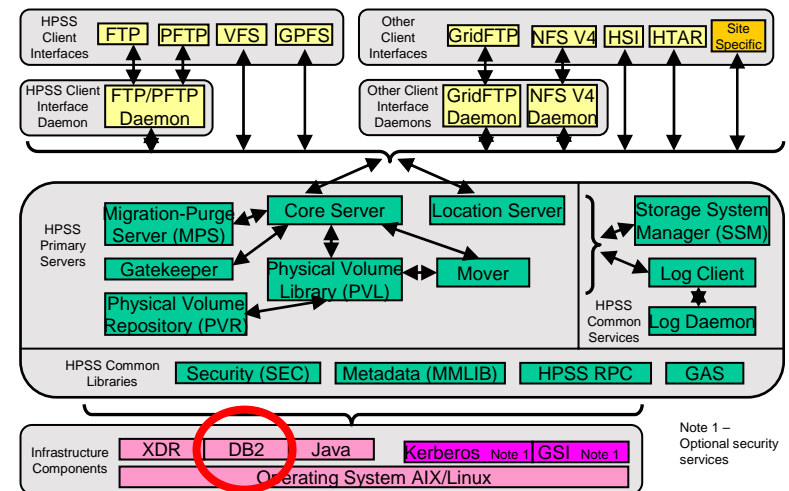
HPSS Architecture

- DB2

- Manages HPSS metadata
- Provides transactional integrity
 - required to guarantee consistency of metadata in case of component failures
- Allows use of standard DB2 backup/restore features to backup/restore HPSS metadata

Infrastructure Components

Replaced Encina/SFS

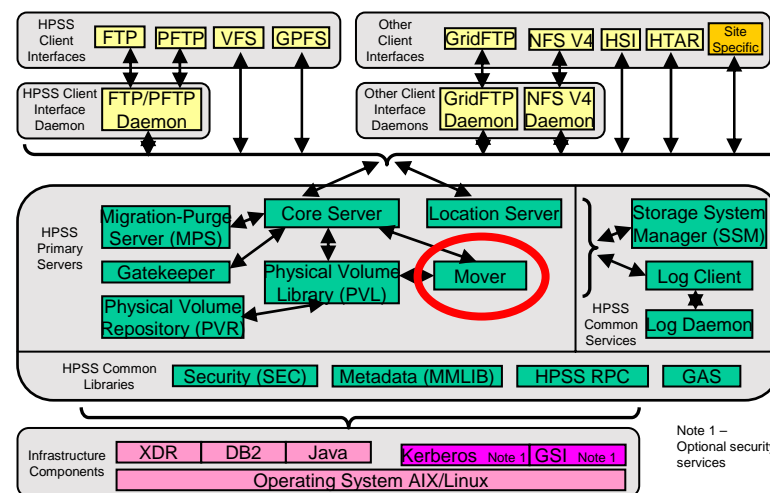


HPSS Architecture

HPSS Server Components

- **Mover (MVR)**

- Transfers data from one location to another.
 - A location may be a standard I/O device (e.g., disk or tape), a network address, or a shared memory location
- A single mover can handle I/O for up to 64 disk and/or tape devices on a single node
- Writes and verifies HPSS media labels



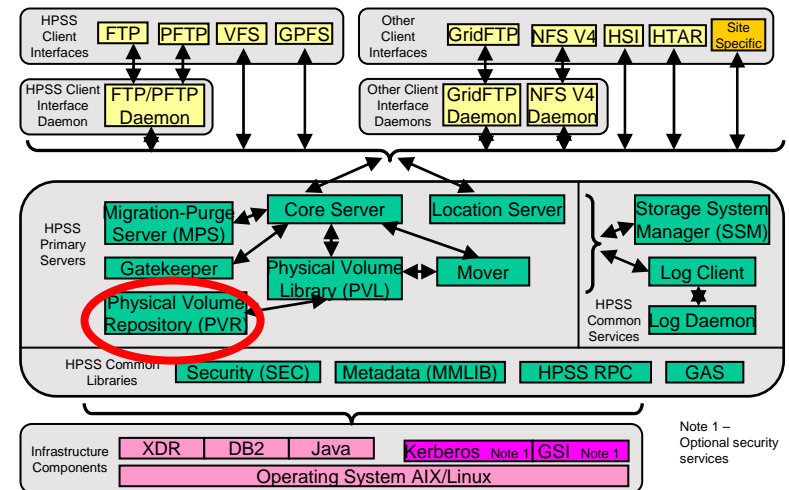
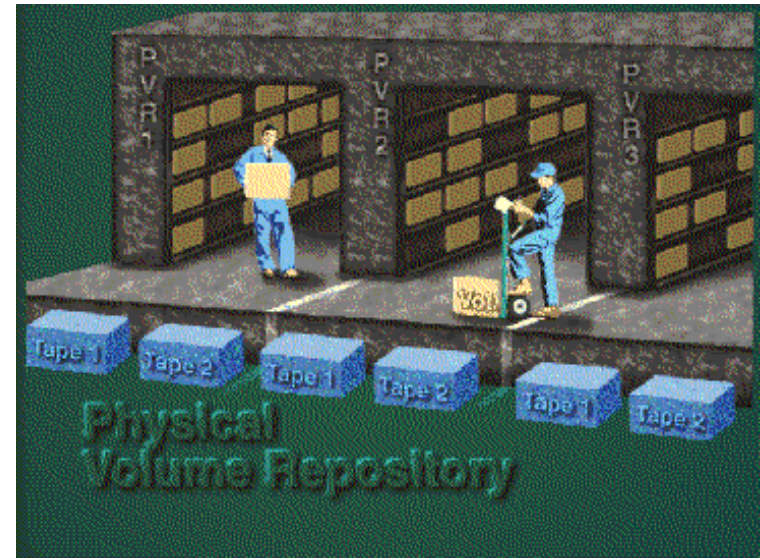
HPSS Architecture

- **Physical Volume Repository (PVR)**

- Manages tape cartridges within a particular robotic library
- Controls all robotics within the library (mount, dismount, etc...)
- Each library class requires a unique PVR. Currently, the following PVR are supported:

- **3494 PVR (support 3494 library)**
- **LTO PVR (support 3584 and IBM TS libraries)**
- **STK PVR (support ACSLS libraries)**
- **ADIC AML (support AML library - by special bid)**
- **SCSI PVR (support IBM (3584/3582), STK (L40/SL500/SL8500), ADIC i500, and Spectra Logic libraries)**
- **Operator PVR (support operator mounted drives not under the control of a robotics device)**

HPSS Server Components

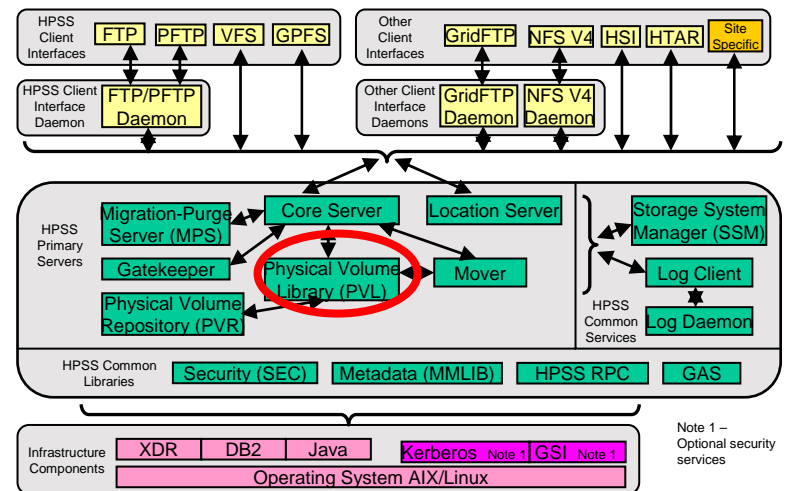
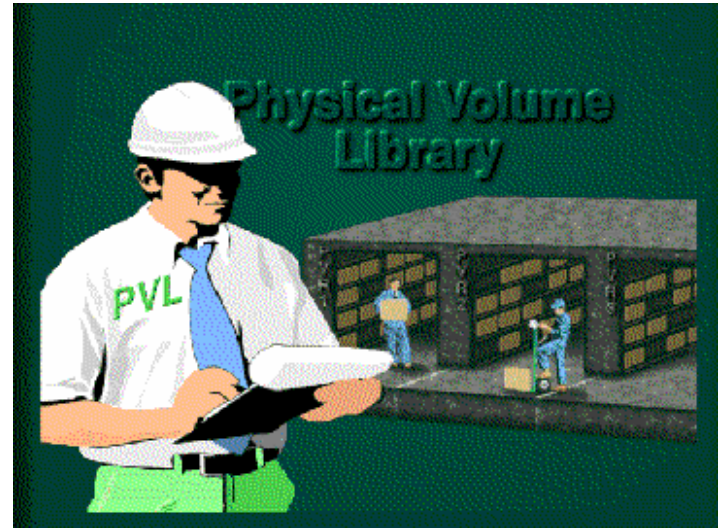


Note 1 – Optional security services

HPSS Architecture

- **Physical Volume Library (PVL)**
 - Manages all HPSS Physical Volumes of tape and disk
 - Coordinates all configured PVR to mount/dismount/eject tape cartridges as directed by Core Server
 - Directs Movers to verify HPSS media labels prior to allowing I/O operations

HPSS Server Components



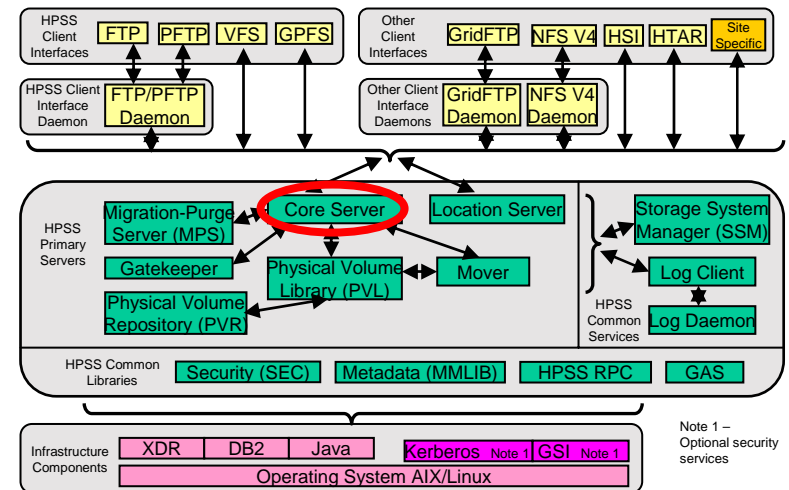
HPSS Architecture

- Core Server

- Manages the HPSS name space
- Enforces security (e.g., file access permissions)
- Handles I/O and name space requests from HPSS clients
- Performs file staging as requested by HPSS clients
- Manages HPSS storage and storage allocation
- Performs file migration and purge as requested by MPS
- Directs the PVL and Movers to fulfill client requests

HPSS Server Components

Combined NS, BFS, SS

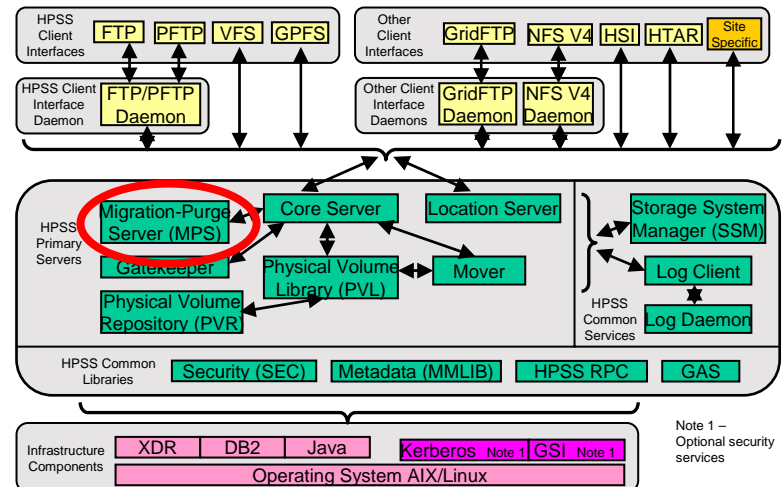


HPSS Architecture

- Migration / Purge Server (MPS)

- Executes defined migration and purge policies to manage HPSS storage space
- Submits migration and purge requests to the Core Server when policies dictate

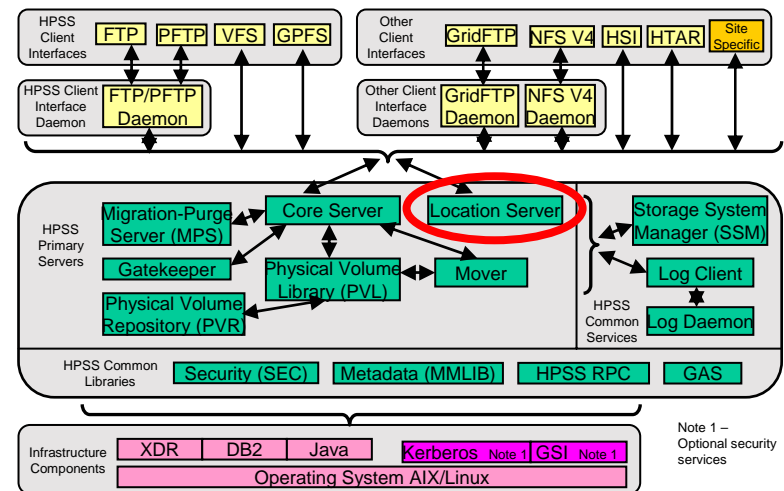
HPSS Server Components



- Location Server (LS)

- Serves as an information clearinghouse:

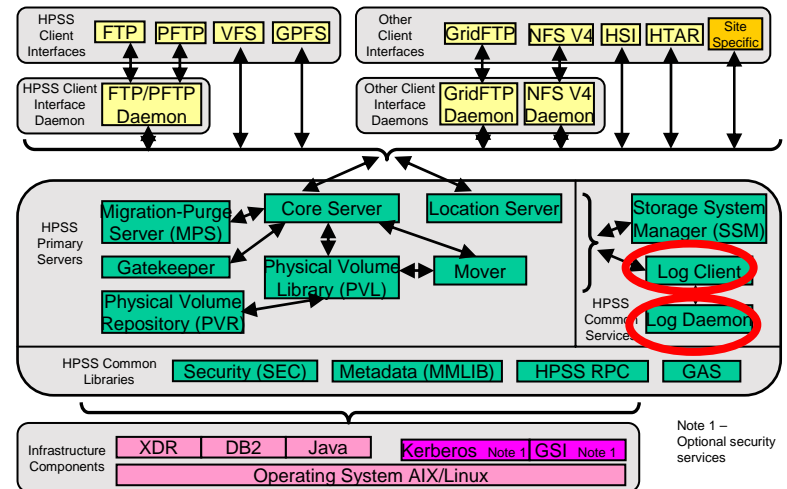
- Allows user interfaces to contact the Core Server
- Allows the Core Server to find the appropriate Gatekeeper



HPSS Architecture

- **Log Daemon (LOGD)**
 - Maintains the central HPSS log (binary format)
 - Optionally archives the central HPSS log into HPSS namespace
- **Log Client (LOGC)**
 - Receives log messages from all HPSS servers on a particular node
 - Applies each server's log policy
 - **Decides which message types are logged for any server**
 - **Decides which message types are sent to SSM to be displayed**
 - Writes these same log messages to a local ASCII log file
 - Forwards appropriate messages to the LOGD for central logging
 - Forwards appropriate log messages to SSM for display

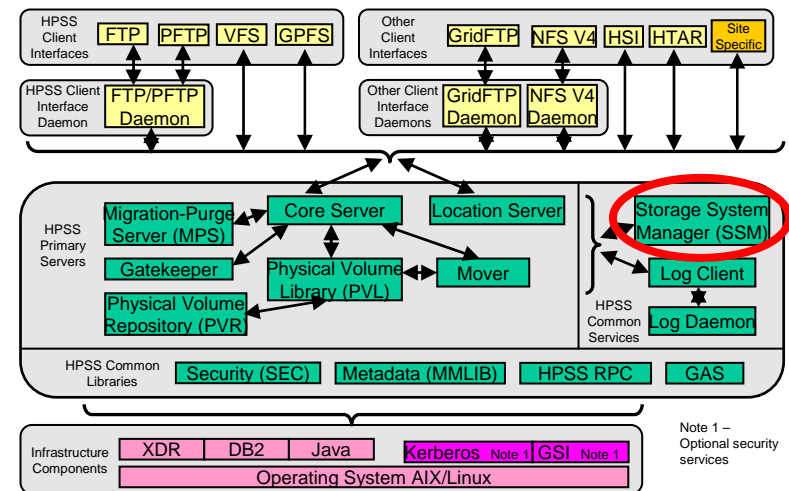
HPSS Server Components



Java Based – Replaced SAMMI

- **Storage System Manager (SSM)**

- Allows administrators to interactively configure, monitor and control HPSS
- Can be administered via either of two interfaces
 - **Graphical User Interface (GUI) - hpssgui**
 - **Command line interface - hpssadm**
 - **Limited configuration capability**



HPSS Architecture

HPSS Server Components

- **Gatekeeper (GK) - OPTIONAL**

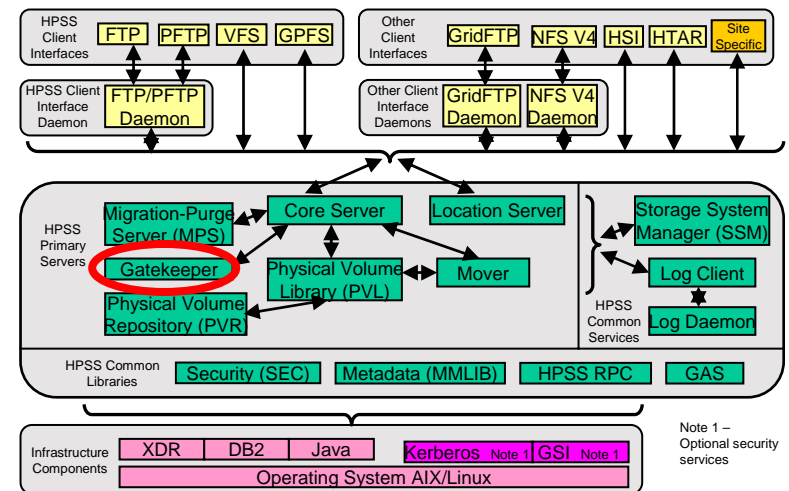
- Allows a site to monitor and restrict certain HPSS file activities

- File creation
- File open
- File stage
- Authorized caller requests (monitor only)

- Allows a site to monitor and restrict certain accounting changes

- File creation
- Directory creation
- File ownership changes
- File account code changes

- All behavior is determined by site-written policy modules (templates provided)



- **Storage Subsystems**

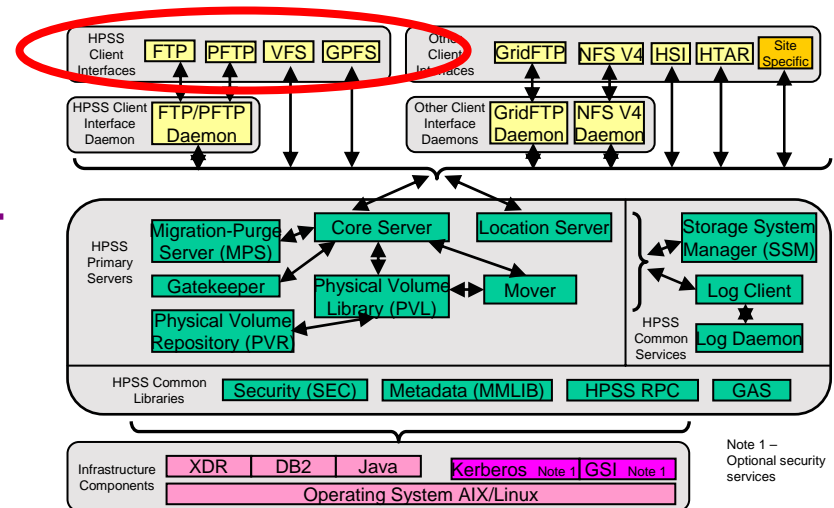
- A storage subsystem (or just 'subsystem') is all the HPSS data, metadata, and storage media managed by a Core Server and a Migration/Purge Server
- Files are assigned to a particular subsystem by their location in the HPSS name space
- Each subsystem has its own pool of VVs for each storage class
- Subsystems are not required to support all of the configured COS
- Migration and purge policies can be modified to act differently depending on the subsystem
- All HPSS systems have at least one subsystem

- **Why use multiple subsystems?**
 - To distribute the database processing overhead between multiple nodes
 - **For WAN-based HPSS systems**
 - **For systems where DB2 would otherwise become a bottleneck**
 - To segregate HPSS resources (nodes, media, networks, etc.) among multiple user groups
 - **When users supply their own nodes, networks, media, etc.**
 - **When users do not want to “step on” one another when putting the system under heavy load**

HPSS Architecture

Client Interfaces

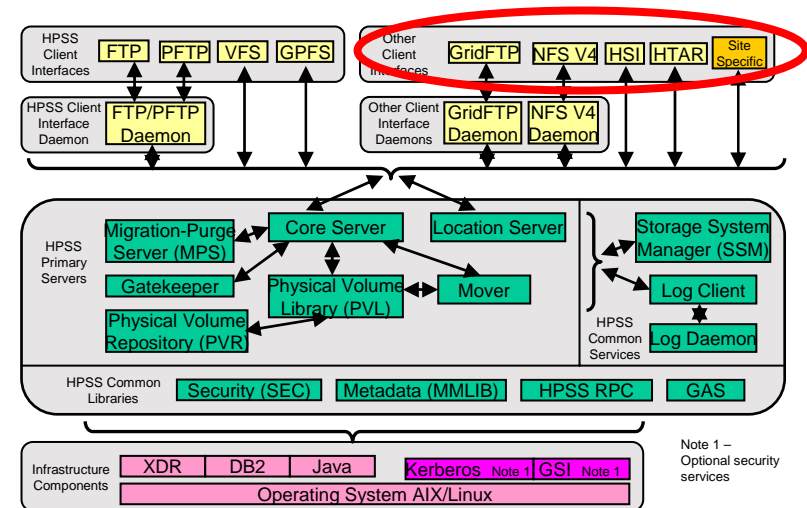
- **FTP**
 - Implements standard FTP command set
 - No special client code required
- **Parallel FTP (pftp_client)**
 - Augments command set to support parallel data paths for HPSS transfers
 - Transfers data directly from Mover to client without routing through the FTP Daemon
- **HPSS VFS Interface Client**
 - Enables HPSS to appear as a local file system on Linux machines
- **GPFS/HPSS Interface (will be 6.2 add-on)**
 - Presents GPFS users with an infinite file system
 - HPSS will archive GPFS data and stage it on demand



HPSS Architecture

Client Interfaces

- **HSI**
 - Provides Unix-style command interface in interactive, batch, and command modes
 - Available from Gleicher Enterprises, Inc.
- **HTAR**
 - Bundles small files into large files and store into HPSS
 - Available from Gleicher Enterprises, Inc.
- **GridFTP enabled for HPSS**
 - An extension of the standard FTP protocol for the GRID Computing environment
 - Available from ANL
- **Network File System (NFS V4)**
 - Allows user to access HPSS as a native file system
 - Available from CEA-DAM
- **Site-specific**
 - Developed and owned by HPSS sites



Configuration Limits

- **Servers**
 - unlimited
- **Storage Policy**
 - Total Accounting Policies: 1
 - Total Migration Policies: 64
 - Total Purge Policies: 64
- **Storage Characteristic**
 - Total Storage Classes: 384
 - Total Storage Hierarchies: 64
 - Total Classes of Service: 64
 - Total File Families: 1024
- **Mover Device / PVL Drive**
 - Total Devices/Drives: unlimited
 - Total Devices per Mover: 64
- **Storage Space**
 - Import: 10,000 cartridges per import request
 - Export: 10,000 cartridges per export request
 - Create: 10,000 virtual volumes per create request
 - Delete: 10,000 physical volumes per delete request



Concepts Covered

- **HPSS Server Components**

- Core Server (Core)
- Migration/Purge Server (MPS)
- Location Server (LS)
- Gatekeeper (GK)
- Physical Volume Library (PVL)
- Physical Volume Repository (PVR)
- Mover (MVR)
- Log Daemon (LOGD)
- Log Client (LOGC)
- Storage System Manager (SSM)

- **Infrastructure Components**

- XDR/RPC
- DB2

- **Client Interfaces**

- File Transfer Protocol (FTP)
- Parallel FTP (PFTP)
- GridFTP
- GPFS/HPSS Interface
- HPSS VFS Client
- HSI
- HTAR
- NFS V4

Exercise

Match the server:

1. Mover (MVR)
2. Physical Volume Repository (PVR)
3. Physical Volume Library (PVL)
4. Core Server (Core)
5. Migration/Purge Server (MPS)
6. Location Server (LS)
7. Log Daemon (LOGD)
8. Log Client (LOGC)
9. Storage System Manager (SSM)
10. Gatekeeper (GK)

with its function:

- A. Manages central HPSS log
- B. Provides administrative interface
- C. Performs all read/writes to/from HPSS media
- D. Directs clients to the appropriate Core Server
- E. Controls tape library robotics
- F. Manages all log messages for an HPSS node
- G. Executes site-written policy modules
- H. Initiates movement of data down a hierarchy
- I. Manages all tape mounts in an HPSS system
- J. Manages namespace and storage allocation

Exercise

- **Determine the most appropriate client interface for each user request:**
 - Wants to port an application that uses POSIX filesystem calls to use HPSS
 - Wants the ability to create and retrieve HPSS files with good performance without the need for writing custom code
 - Wants easy access to the HPSS namespace and doesn't plan on doing much I/O (perhaps just one or two small files from time to time)
 - Others

THE END of the Overview