HPSS

Installation Guide

High Performance Storage System
Release 4.5

September 2002 (Revision 2)

HPSS Installation Guide

Printed in the United States of America.

Printed in the United States of America.

HPSS Release 4.5
September 2002 (Revision 2)

Table of Contents

List of Figures

List of Tables

Preface

Conventions Used in This Book

Example commands that should be typed at a command line will be proceeded by a percent sign ('**%**') and be presented in a boldface courier font:

```
% sample command
```

Names of files, variables, and variable values will appear in a boldface courier font:

Sample file, variable, or variable value

Any text preceded by a pound sign ('**#**') should be considered shell script comment lines:

```
# This is a comment
```

Chapter 1

HPSS Basics

1.1 Introduction

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

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

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

1.2 HPSS Capabilities

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

1.2.1 Network-centered Architecture

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

1.2.2 *High Data Transfer Rate*

HPSS achieves high data transfer rates by eliminating overhead normally associated with data transfer operations. In general, HPSS servers establish transfer sessions but are not involved in actual transfer of data.

1.2.3 *Parallel Operation Built In*

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

1.2.4 *A Design Based on Standard Components*

HPSS runs on UNIX with no kernel modifications and is written in ANSI C and Java. It uses the OSF Distributed Computing Environment (DCE) and Encina from Transarc Corporation as the basis for its portable, distributed, transaction-based architecture. These components are offered on many vendors' platforms. Source code is available to vendors and users for porting HPSS to new platforms. HPSS Movers and the Client API have been ported to non-DCE platforms. HPSS has been implemented on the IBM AIX and Sun Solaris platforms. In addition, selected components have been ported to other vendor platforms. The non-DCE Client API and Mover have been ported to SGI IRIX, while the Non-DCE Client API has also been ported to Linux. Parallel FTP client software has been ported to a number of vendor platforms and is also supported on Linux. Refer to Section 1.4: *HPSS Hardware Platforms* on page 37 and Section 2.3: *Prerequisite Software Considerations* on page 46 for additional information.

1.2.5 *Data Integrity Through Transaction Management*

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

1.2.6 *Multiple Hierarchies and Classes of Services*

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

1.2.7 Storage Subsystems

To increase the scalability of HPSS in handling concurrent requests, the concept of Storage Subsystem has been introduced. Each Storage Subsystem contains a single Name Server and Bitfile Server. If migration and purge are needed for the storage subsystem, then the Storage Subsystem will also contain a Migration / Purge Server. A Storage Subsystem must also contain a single Tape Storage Server and/or a single Disk Storage Server. All other servers exist outside of Storage Subsystems. Data stored within HPSS is assigned to different Storage Subsystems based on pathname resolution. A pathname consisting of **/** resolves to the root Name Server. The root Name Server is the Name Server specified in the Global Configuration file. However, if the pathname contains junction components, it may resolve to a Name Server in a different Storage Subsystem. For example, the pathname **/JunctionToSubsys2** could lead to the root fileset managed by the Name Server in Storage Subsystem 2. Sites which do not wish to partition their HPSS through the use of Storage Subsystems will effectively be running an HPSS with a single Storage Subsystem. Note that sites are not required to use multiple Storage Subsystems.

1.2.8 Federated Name Space

Federated Name Space supports data access between multiple, separate HPSS systems. With this capability, a user may access files in all or portions of a separate HPSS system using any of the configured HPSS interfaces. To create a Federated Name Space, junctions are created to point to filesets in a different HPSS system. For security purposes, access to foreign filesets is not supported for NFS, or for end-users of FTP and the Non-DCE Gateway when only the local password file is used for authentication.

1.3 HPSS Components

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

These HPSS components are discussed below in Sections 1.3.1 through 1.3.7.

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

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

- ***Files (Bitfiles).*** Files in HPSS, called bitfiles in deference to IEEE Reference Model terminology, are logical strings of bytes, even though a particular bitfile may have a structure imposed by its owner. This unstructured view decouples HPSS from any particular file management system that host clients of HPSS might have. HPSS bitfile size is limited to 2 to the power of 64 minus 1 (2^{64} - 1) bytes.

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

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

- ***Junctions.*** A junction is a Name Server object that is used to point to a fileset. This fileset may belong to the same Name Server or to a different Name Server. The ability to point junctions allows HPSS users to traverse to different Storage Subsystems and to traverse to different HPSS systems via the Federated Name Space. Junctions are components of pathnames and are the mechanism which implements this traversal.

- ***File Families***. HPSS files can be grouped into families. All files in a given family are recorded on a set of tapes assigned to the family. Only files from the given family are recorded on these tapes. HPSS supports grouping files on tape volumes only. Families can only be specified by associating the family with a fileset. All files created in the fileset belong to the family. When one of these files is migrated from disk to tape, it is recorded on a tape with other files in the same family. If no tape virtual volume is associated with the family, a blank tape is reassigned from the default family. The family affiliation is preserved when tapes are repacked.

- ***Physical Volumes.*** A physical volume is a unit of storage media on which HPSS stores data. The media can be removable (e.g., cartridge tape, optical disk) or non-removable (magnetic disk). Physical volumes may also be composite media, such as RAID disks, but must be represented by the host OS as a single device.

  Physical volumes are not visible to the end user. The end user simply stores bitfiles into a logically unlimited storage space. HPSS, however, must implement this storage on a variety of types and quantities of physical volumes.

  For a list of the tape physical volume types supported by HPSS, see Table 2-4: *Suggested Block Sizes for Tape* on page 99.

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

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

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

- ***Storage Classes.*** A storage class defines a set of characteristics and usage parameters to be associated with a particular grouping of HPSS virtual volumes. Each virtual volume and its associated physical volumes belong to a single storage class in HPSS. Storage classes in turn are grouped to form storage hierarchies (see below). An HPSS storage class is used to logically group storage media to provide storage for HPSS files with specific intended usage, similar size and usage characteristics.

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

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

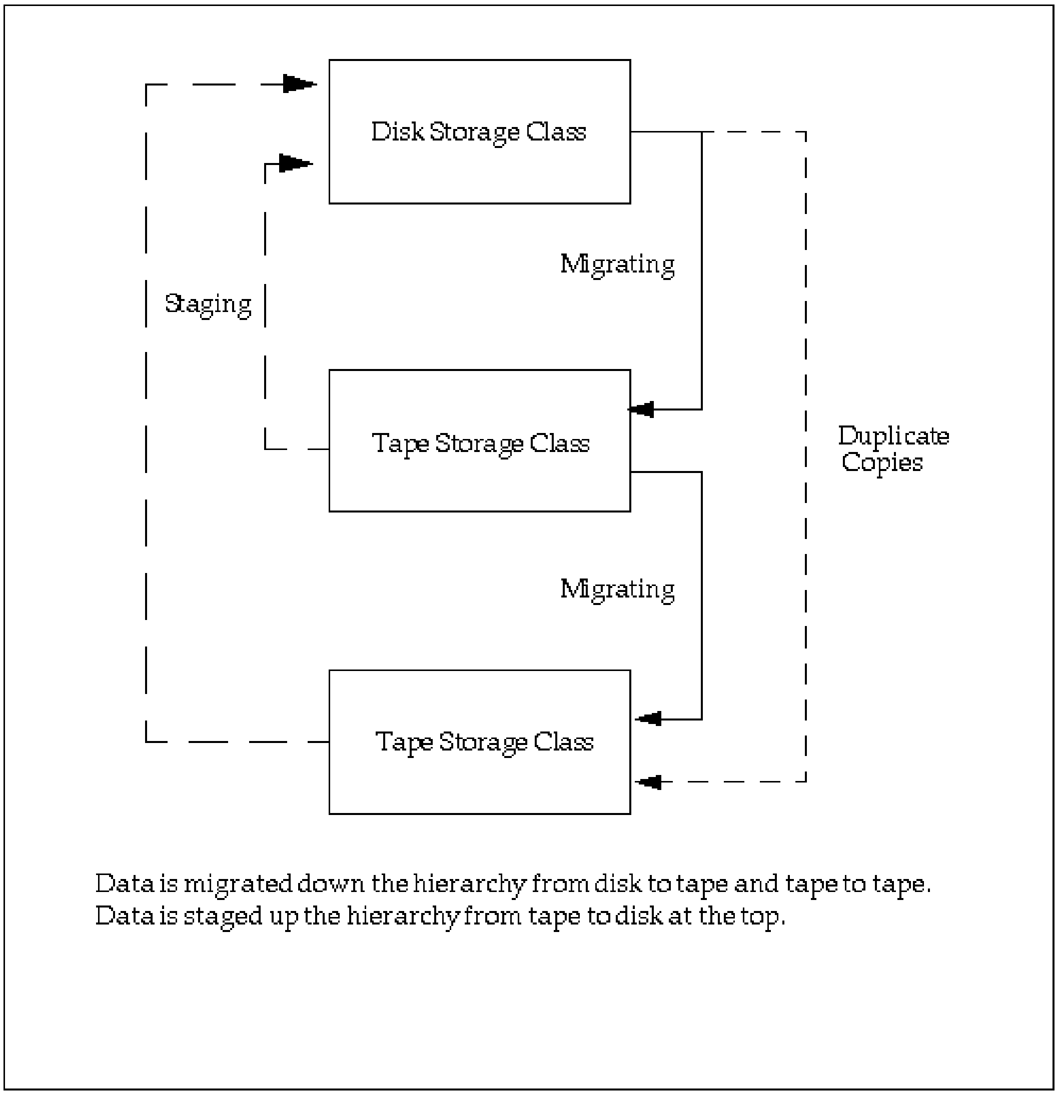

Disk Storage Class
Migrating
Staging
Tape Storage Class
Duplicate Copies
Migrating
Tape Storage Class
Data is migrated down the hierarchy from disk to tape and tape to tape.
Data is staged up the hierarchy from tape to disk at the top.

Figure 1-1 Migrate and Stage Operations

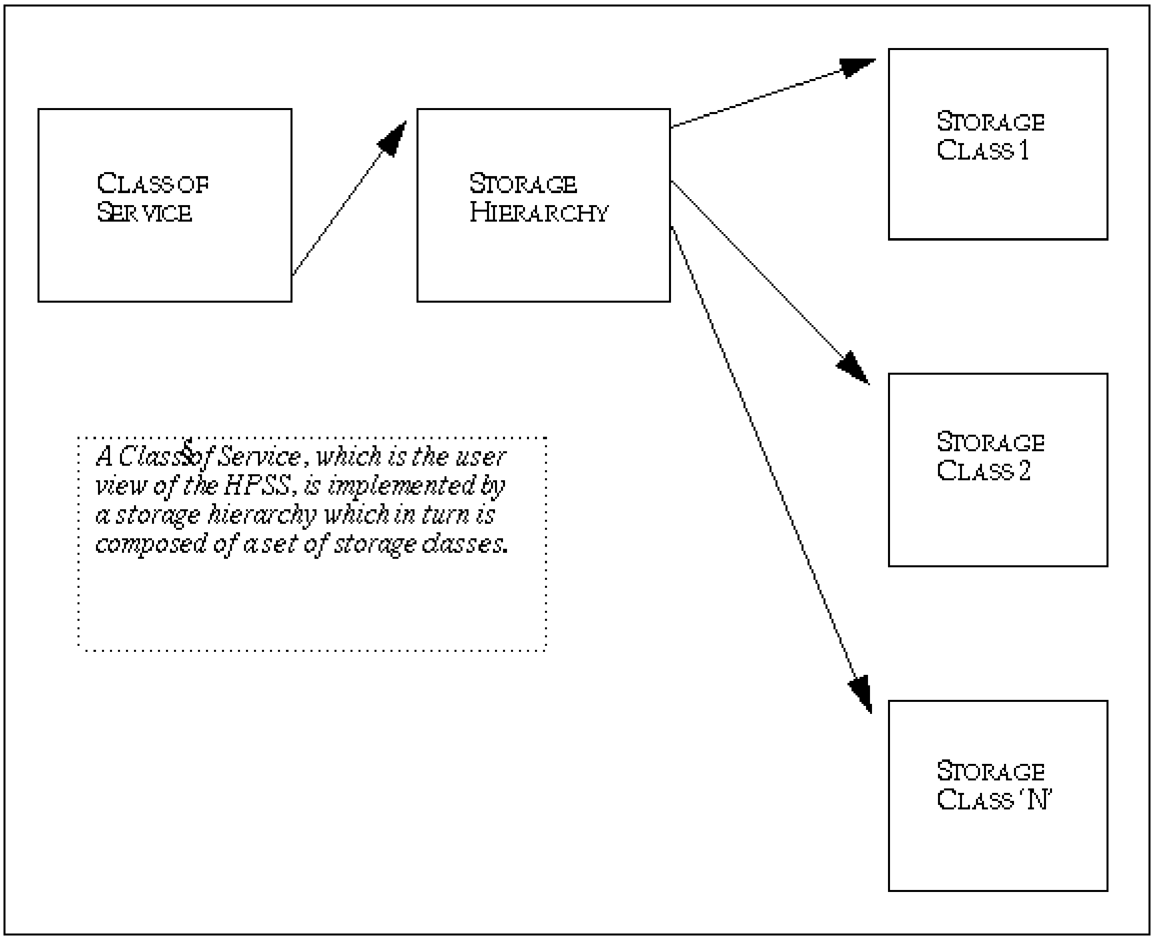

CLASS OF SERVICE
STORAGE HIERARCHY
STORAGE CLASS 1
STORAGE CLASS 2
STORAGE CLASS 'N'
A Class of Service, which is the user view of the HPSS, is implemented by a storage hierarchy which in turn is composed of a set of storage classes.

Figure 1-2 Relationship of HPSS Data Structures

1.3.2 HPSS Core Servers

HPSS servers include the Name Server, Bitfile Server, Migration / Purge Server, Storage Server, Gatekeeper Server, Location Server, DMAP Gateway, Physical Volume Library, Physical Volume Repository, Mover, Storage System Manager, and Non-DCE Client Gateway. Figure 1-3 provides a simplified view of the HPSS system. Each major server component is shown, along with the basic control communications paths (thin arrowed lines). The thick line reveals actual data movement. Infrastructure items (those components that "glue together" the distributed servers) are shown at the top of the cube in gray scale. These infrastructure items are discussed in Section 1.3.4. HPSS user interfaces (the clients listed in the figure) are discussed in Section 1.3.5.

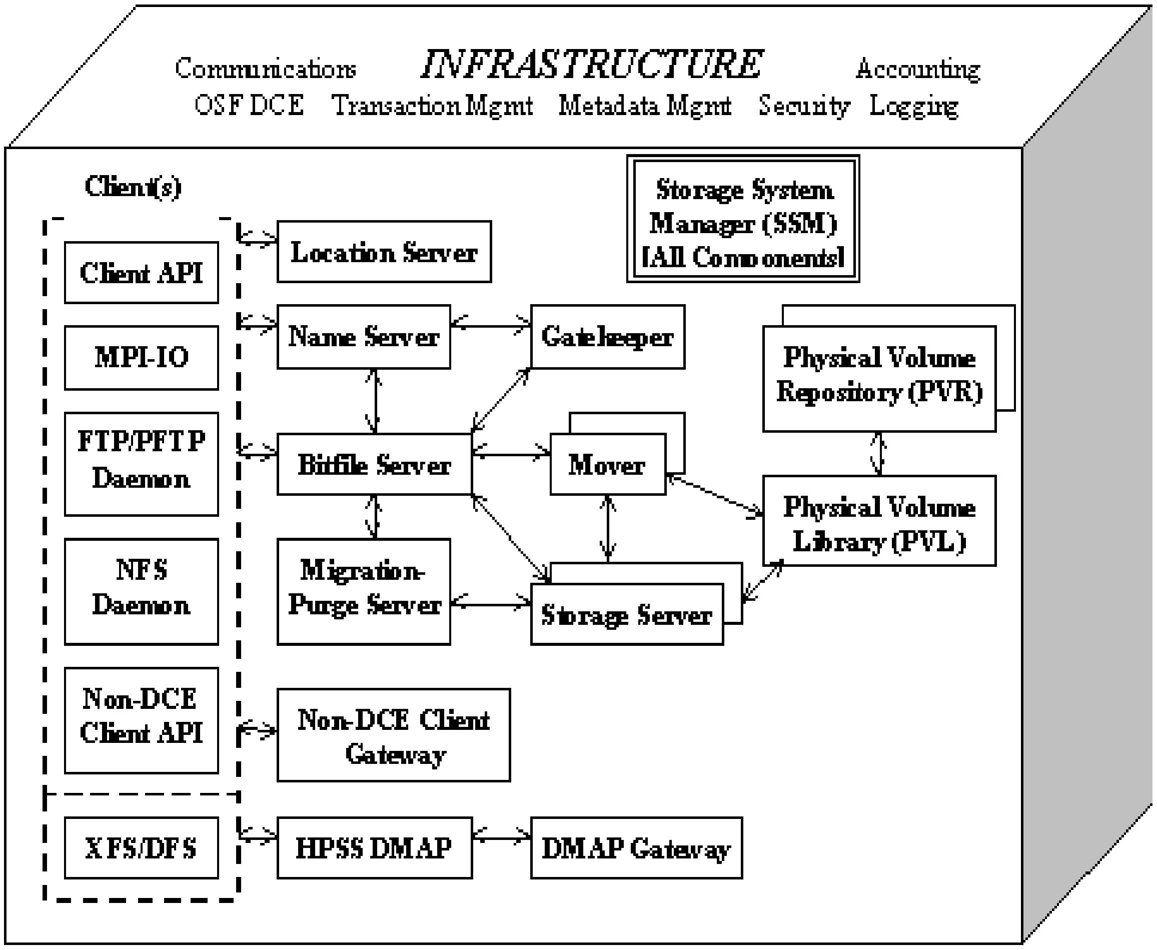


Figure 1-3 The HPSS System

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

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

- *Migration/Purge Server (MPS).* The MPS allows the local site to implement its storage management policies by managing the placement of data on HPSS storage media using site-defined migration and purge policies. By making appropriate calls to the Bitfile and Storage Servers, MPS copies data to lower levels in the hierarchy (*migration*), removes data from the current level once copies have been made (*purge*), or moves data between volumes at the same level (*lateral move*). Based on the hierarchy configuration, MPS can be directed to create duplicate copies of data when it is being migrated from disk or tape. This is done by copying the data to one or more lower levels in the storage hierarchy.

  There are three types of migration: *disk migration*, *tape file migration*, and *tape volume migration*. *Disk purge* should always be run along with disk migration. The designation *disk* or *tape* refers to the type of storage class that migration is running against. See Section 2.6.5: *Migration/Purge Server* on page 65 for a more complete discussion of the different types of migration.

  Disk Migration/Purge:

  The purpose of disk migration is to make one or more copies of disk files to lower levels in the hierarchy. The number of copies depends on the configuration of the hierarchy. For disk, migration and purge are separate operations. Any disk storage class which is configured for migration should be configured for purge as well. Once a file has been migrated (copied) downwards in the hierarchy, it becomes eligible for purge, which subsequently removes the file from the current level and allows the disk space to be reused.

  Tape File Migration:

  The purpose of tape file migration is to make a single, additional copy of files in a tape storage class to a lower level in the hierarchy. It is also possible to move files downwards instead of copying them. In this case there is no duplicate copy maintained.

  That there is no separate purge component to tape file migration, and tape volumes will require manual repack and reclaim operations to be performed by the admin.

  Tape Volume Migration:

  The purpose of tape volume migration is to free tape volumes for reuse. Tape volumes are selected based on being in the EOM map state and containing the most unused space (caused by users overwriting or deleting files). The remaining segments on these volumes are either migrated downwards to the next level in the hierarchy, or are moved laterally to another tape volume at the same level. This results in empty tape volumes which may then be reclaimed. Note that there is no purge component to tape volume migration. All of the operations use a *move* instead of a *copy* semantic.

  MPS runs migration on each storage class periodically using the time interval specified in the migration policy for that class. See Section 1.3.7: *HPSS Policy Modules* on page 35 for details on migration and purge policies. In addition, migration runs can be started automatically when the warning or critical space thresholds for the storage class are exceeded. Purge runs are started automatically on each storage class when the free space in that class falls below the percentage specified in the purge policy.

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

- *Gatekeeper Server (GK).* The Gatekeeper Server provides two main services:

  A. It provides sites with the ability to schedule the use of HPSS resources using the Gatekeeping Service.

  B. It provides sites with the ability to validate user accounts using the Account Validation Service.

  Both of these services allow sites to implement their own policy.

  The default Gatekeeping Service policy is to not do any gatekeeping. Sites may choose to implement site policy for monitoring authorized callers, creates, opens, and stages. The BFS will call the appropriate GK API depending on the requests that the site-implemented policy is monitoring.

  The Account Validation Service performs authorizations of user storage charges. A site may perform no authorization, default authorization, or site-customized authorization depending on how the Accounting Policy is set up and whether or not a site has written site-specific account validation code. Clients call this service when creating files, changing file ownership, or changing accounting information. If Account Validation is enabled, the Account Validation Service determines if the user is allowed to use a specific account or gives the user an account to use, if needed. The Name Server and Bitfile Server also call this service to perform an authorization check just before account-sensitive operations take place.

- *Location Server (LS).* The Location Server acts as an information clearinghouse to its clients through the HPSS Client API to enable them to locate servers and gather information from both local and remote HPSS systems. Its primary function is to allow a client to determine a server's location, its CDS pathname, by knowing other information about the server such as its object UUID, its server type or its subsystem id. This allows a client to contact the appropriate server. Usually this is for the Name Server, the Bitfile Server or the Gatekeeper.

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

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

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

For information on the types of tape libraries supported by HPSS PVRs, see Section 2.4.2: *Tape Robots* on page 54.

An Operator PVR is provided for cartridges not under control of a robotic library. These cartridges are mounted on a set of drives by operators.

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

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

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

- *Non-DCE Client Gateway (NDCG).* NDCG provides an interface into HPSS for client programs running on systems lacking access to DCE or Encina services. By linking the Non-DCE Client API library, instead of the usual Client API library, all API calls are routed through the NDCG. The API calls are then executed by the NDCG, and the results are returned to the client application. Note that the NDCG itself must still run on a system with DCE and Encina, while it is the client application using the Non-DCE Client API that does not suffer this restriction.

1.3.3 HPSS Storage Subsystems

Storage subsystems have been introduced starting with the 4.2 release of HPSS. The goal of this design is to increase the scalability of HPSS by allowing multiple name and bitfile servers to be used within a single HPSS system. Every HPSS system must now be partitioned into one or more storage subsystems. Each storage subsystem contains a single name server and bitfile server. If migration and purge are needed, then the storage subsystem should contain a single, optional migration/purge server. A storage subsystem must also contain one or more storage servers, but only one disk storage server and one tape storage server are allowed per subsystem. Name, bitfile, migration/

purge, and storage servers must now exist within a storage subsystem. Each storage subsystem may contain zero or one gatekeepers to perform site specific user level scheduling of HPSS storage requests or account validation. Multiple storage subsystems may share a gatekeeper. All other servers continue to exist outside of storage subsystems. Sites which do not need multiple name and bitfile servers are served by running an HPSS with a single storage subsystem.

Storage subsystems are assigned integer ids starting with one. Zero is not a valid storage subsystem id as servers which are independent of storage subsystems are assigned to storage subsystem zero. Storage subsystem ids must be unique. They do not need to be sequential and need not start with one, but they do so by default unless the administrator specifies otherwise. Each storage subsystem has a user-configurable name as well as a unique id. The name and id may be modified by the administrator at the time the subsystem is configured but may not be changed afterward. In most cases, the storage subsystem is referred to by its name, but in at least one case (suffixes on metadata file names) the storage subsystem is identified by its id. Storage subsystem names must be unique.

There are two types of configuration metadata used to support storage subsystems: a single global configuration record, and one storage subsystem configuration record per storage subsystem. The global configuration record contains a collection of those configuration metadata fields which are used by multiple servers and that are commonly modified. The storage subsystem records contain configuration metadata which is commonly used within a storage subsystem.

It is possible to use multiple SFS servers within a single HPSS system. Multiple storage subsystems are able to run from a single SFS server or using one SFS server per storage subsystem. In practice, different metadata files may be located on different SFS servers on a per file basis depending on the SFS path given for each file. For configuration and recovery purposes, however, it is desirable for all of the metadata files for a single subsystem to reside on a single SFS server. This single SFS server may either be a single server which supports the entire HPSS system, or it may support one or more subsystems. Those metadata files which belong to the servers which reside within storage subsystems are considered to belong to the storage subsystem as well. In an HPSS system with multiple storage subsystems, there are multiple copies of these files, and the name of each copy is suffixed with the integer id of the subsystem so that it may be uniquely identified (for example **`bfmigrrec.1`**, **`bfmigrrec.2`**, etc.).

Metadata files that belong to a subsystem (i.e. files with numeric suffix) should never *be shared between servers. For example, the Bitfile Server in Subsystem #1 has a metadata file called* bfmigrrec.1*. This file should only be used by the BFS in Subsystem #1, never by* any *other server.*

The definitions of classes of service, hierarchies, and storage classes apply to the entire HPSS system and are independent of storage subsystems. All classes of service, hierarchies, and storage classes are known to all storage subsystems within HPSS. The level of resources dedicated to these entities by each storage subsystem may differ. It is possible to disable selected classes of service within given storage subsystems. Although the class of service definitions are global, if a class of service is disabled within a storage subsystem then the bitfile server in that storage subsystem never selects that class of service. If a class of service is enabled for a storage subsystem, then there must be a non-zero level of storage resources supporting that class of service assigned to the storage servers in that subsystem.

Data stored within HPSS is assigned to different Storage Subsystems based on pathname resolution. A pathname consisting of **"/"** resolves to the root Name Server. The root Name Server is the Name Server specified in the Global Configuration file. However, if the pathname contains junction components, it may resolve to a Name Server in a different Storage Subsystem. For example, the pathname "**/JunctionToSubsys2**" could lead to the root fileset managed by the Name Server in Storage Subsystem 2. Sites which do not wish to partition their HPSS through the use of

Storage Subsystems will effectively be running an HPSS with a single Storage Subsystem. Note that sites are not required to use multiple Storage Subsystems.

Since the migration / purge server is contained within the storage subsystem, migration and purge operate independently in each storage subsystem. If multiple storage subsystems exist within an HPSS, then there are several migration / purge servers operating on each storage class. Each migration / purge server is responsible for migration and purge for those storage class resources contained within its particular storage subsystem. Migration and purge runs are independent and unsynchronized. This principle holds for other operations such as repack and reclaim as well. Migration and purge for a storage class may be configured differently for each storage subsystem. It is possible to set up a single migration or purge policy which applies to a storage class across all storage subsystems (to make configuration easier), but it is also possible to control migration and purge differently in each storage subsystem.

Storage class thresholds may be configured differently for each storage subsystem. It is possible to set up a single set of thresholds which apply to a storage class across all storage subsystems, but it is also possible to control the thresholds differently for each storage subsystem.

1.3.4 HPSS Infrastructure

The HPSS infrastructure items (see Figure 1-3) are those components that “glue together” the distributed servers. While each HPSS server component provides some explicit functionality, they must all work together to provide users with a stable, reliable, and portable storage system. The HPSS infrastructure components common among servers that tie servers together are discussed below.

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

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

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

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

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

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

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

  - ◆ *Authentication* — is responsible for guaranteeing that a principal (a customer identity) is the entity that is claimed, and that information received from an entity is from that entity. An additional mechanism is provided to allow authentication of File Transfer Protocol (FTP) users through a site-supplied policy module.

  - ◆ *Authorization* — is responsible for enabling an authenticated entity access to an allowed set of resources and objects. Authorization enables end user access to HPSS directories and bitfiles.

  - ◆ *Enforcement* — is responsible for guaranteeing that operations are restricted to the authorized set of operations.

  - ◆ *Audit* — is responsible for generating a log of security-relevant activity. HPSS audit capabilities allow sites to monitor HPSS authentication, authorization, and file security events. File security events include file creation, deletion, opening for I/O, and attribute modification operations.

  - ◆ *Security management* — allows the HPSS administrative component to monitor the underlying DCE security services used by the HPSS security subsystem.

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

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

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

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

  For every account index, the storage usage information is written out to an ASCII text file. It is the responsibility of the individual site to sort and use this information for subsequent billing based on site-specific charging policies. For more information on the HPSS accounting policy, refer to Section 1.3.7.

1.3.5 HPSS User Interfaces

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

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

- *Network File System (NFS).* The NFS server interface for HPSS provides transparent access to HPSS name space objects and bitfile data for client systems through the NFS service. The NFS implementation consists of an NFS Daemon and a Mount Daemon that provide access to HPSS, plus server support functions that are not accessible to NFS clients. The HPSS NFS service will work with any industry-standard NFS client that supports either (or both) NFS V2 and V3 protocols.

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

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

- *Non-DCE Client Application Program Interface (Non-DCE Client API).* The Non-DCE Client API is a programming interface that allows the client program the option of running on a platform that does not support DCE and Encina. This API does not call HPSS directly. Instead, it sends network messages to the Non-DCE Client Gateway, which runs on the target HPSS system. The NDCG then performs the requested Client API calls for the client and returns the results. The Non-DCE Client API has the same functionality as the standard Client API with the following exceptions: it does not support ACL functions, it does not support transactional processing, and it implements its own security interface. Client authentication is performed in one of three ways: remote DCE login, kerberos authentication or no authentication (in case of trusted clients).

- *MPI-IO Application Programming Interface (MPI-IO API).* The MPI-IO API is a subset of the MPI-2 standard. It gives applications written for a distributed memory programming model an interface that offers coordinated access to HPSS files from multiple processes. The interface also lets applications specify discontiguous patterns of access to files and memory buffers using the same "datatype" constructs that the Message-Passing Interface (MPI) offers. For the specific details of this interface, see the *HPSS Programmer's Reference Guide*, Volume 1, Release 4.5.

- *Distributed File System (DFS).* Distributed file system services optionally allow HPSS to interface with Transarc's DFS^{TM}. DFS is a scalable distributed file system that provides a uniform view of file data to all users through a global name space. DFS uses the DCE concept of cells, and allows data access and authorization between clients and servers in different cells. DFS uses the Episode physical file system which communicates with HPSS via an XDSM-compliant interface. For the specific details of this interface, refer to Section 7.6: *HDM Configuration* on page 435.

- *XFS.* HPSS is capable of interfacing with SGI's XFS for Linux. XFS is a scalable open-source journaling file system for Linux that provides an implementation of the Open Group's XDSM interface. This essentially allows the XFS file system to become part of an HPSS storage hierarchy, with file data migrated into HPSS and purged from XFS disk. For the specific details of this interface, refer to Section 7.6: *HDM Configuration* on page 435.

1.3.6 HPSS Management Interface

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

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

HPSS also provides a command line SSM interface, **hpssadm**. This tool does not provide all the functionality of the GUI, but does implement a subset of its frequently used features, such as some monitoring and some control of servers, devices, storage classes, volumes, and alarms. It is useful for performing HPSS administration from remote locations where X traffic is slow, difficult, or impossible, such as from home or from scripts. Additionally, **hpssadm** provides some rudimentary mass configuration support by means of the ability to issue configuration commands from a batch script.

1.3.7 HPSS Policy Modules

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

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

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

  Note that the number of copies which migration makes and the location of these copies is determined by the definition of the storage hierarchy and not by the migration policy.

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

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

- *Security Policy*. Site security policy defines the authorization and access controls to be used for client access to HPSS. Site policy managers were developed for controlling access from FTP and/or Parallel FTP using either **Ident** or **Kerberos** credentials. These access methods are supported by request using the **hpss_pftpd_amgr** and an appropriate authentication manager. The Policy Manager is no longer supported. The Non-DCE Client Gateway provides three Security Policies: none, Kerberos, and DCE.

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

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

- *Accounting Policy*. The accounting policy provides runtime information to the accounting report utility and to the Account Validation service of the Gatekeeper. It helps determine what style of accounting should be used and what level of validation should be enforced.

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

  Account Validation allows a site to perform usage authorization of an account for a user. It is turned on by enabling the Account Validation field. If Account Validation is enabled, the accounting style in use at the site is determined by the Accounting Style field. A site policy module may be implemented by the local site to perform customized account validation operations. The default Account Validation behavior is performed for any Account Validation operation that is not overridden by the site policy module.

  If Account Validation is not enabled, as in previous versions of HPSS, the accounting style to use is determined by the GECOS field on the user's DCE account in the DCE registry or by the HPSS.gecos Extended Registry Attribute (ERA) on the DCE principal in the DCE registry.

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

- *Gatekeeping Policy*. The Gatekeeper Server provides a Gatekeeping Service along with an Account Validation Service. These services provide the mechanism for HPSS to communicate information though a well-defined interface to a policy software module that can be completely written by a site. The site policy code is placed in well-defined shared libraries for the gatekeeping policy and the accounting policy (**/opt/hpss/lib/libgksite.[a|so]** and **/opt/hpss/lib/libacctsite.[a|so]** respectively) which are linked to the Gatekeeper Server. The Gatekeeping policy shared library contains a default policy which does NO gatekeeping. Sites will need to enhance this library to implement local policy rules if they wish to monitor and load-balance requests.

1.4 HPSS Hardware Platforms

1.4.1 Client Platforms

The following matrix illustrates which platforms support HPSS interfaces.

Table 1-1 HPSS Client Interface Platforms

Platform	PFTP Client	Client API	Non-DCE Client API	MPI-IO	DFS HDM	XFS HDM	NFS, DFS, and FTP Clients
IBM AIX	X	X	X	X	X		Any platform running standard NFS, DFS, or FTP clients, respectively
Sun Solaris	X	X	X	X	X		
Cray UniCOS	X						
Intel Teraflop	X						
Digital Unix	X						
Hewlett-Packard HPUX	X						
Silicon Graphics IRIX (32-bit)	X		X				
Compaq Tru64	X						
Linux (Intel)	X		X			X	
Linux (OpenNAS)						X	
Linux (Alpha)	X						

The full-function Client API can be ported to any platform that supports Unix, DCE, and Encina.

The PFTP client code and Client API source code for the platforms other than AIX, Linux, and Solaris listed above is on the HPSS distribution tape. Maintenance of the PFTP and Client API software on these platforms is the responsibility of the customer, unless a support agreement is negotiated with IBM. Contact IBM for information on how to obtain the software mentioned above.

The MPI-IO API can be ported to any platform that supports a compatible host MPI and the HPSS Client API (DCE or Non-DCE version). See Section 2.5.6: *MPI-IO API* on page 60 for determining a compatible host MPI.

The XFS HDM is supported on standard Intel Linux platforms as well as the OpenNAS network appliance from Consensys Corp.

1.4.2 Server and Mover Platforms

HPSS currently requires at least one AIX or Solaris machine for the core server components. The core server machine must include DCE and Encina as prerequisites (see Section 2.3: *Prerequisite Software Considerations* on page 46) and must have sufficient processing power and memory to handle the work load needed by HPSS.

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

Chapter 2

HPSS Planning

2.1 Overview

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

- Requirements and Intended Usages for HPSS on page 43
- Prerequisite Software Considerations on page 46
- Hardware Considerations on page 53
- HPSS Interface Considerations on page 58
- HPSS Server Considerations on page 62
- Storage Subsystem Considerations on page 81
- Storage Policy Considerations on page 81
- Storage Characteristics Considerations on page 91
- HPSS Sizing Considerations on page 105
- HPSS Performance Considerations on page 135
- HPSS Metadata Backup Considerations on page 142

The planning process for HPSS must be done carefully to ensure that the resulting system satisfies the site's requirements and operates in an efficient manner. We recommend that the administrator read the entire document before planning the system.

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

2.1.1 HPSS Configuration Planning

Before beginning the planning process, there is an important issue to consider. HPSS handles large files much better than it does small files. If at all possible, try to reduce the number of small files that are introduced into your HPSS system. For example, if you plan to use HPSS to backup all of the PCs in your organization, it would be best to aggregate the individual files into large individual files before moving them into the HPSS name space.

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

1. Identify the site's storage requirements and policies, such as the initial storage system size, anticipated growth, usage trends, average file size, expected throughput, backup policy. and availability. For more information, see Section 2.2: *Requirements and Intended Usages for HPSS* on page 43.

2. Define the architecture of the entire HPSS system to satisfy the above requirements.The planning should:

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

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

   ◆ Identify the HPSS subsystems to be configured and how resources will be allocated among them. Refer to Section 2.7: *Storage Subsystem Considerations* on page 81 for more discussion on subsystems.

   ◆ Identify the HPSS servers to be configured and the node where each of the servers will run. Refer to Section 2.6: *HPSS Server Considerations* on page 62 for more discussions on the HPSS server configuration.

   ◆ Identify the HPSS user interfaces (e.g. FTP, PFTP, NFS,XFS, DFS) to be configured and the nodes where the components of each user interface will run. Refer to Section 2.5: *HPSS Interface Considerations* on page 58 for more discussion on the user interfaces supported by HPSS.

3. Ensure that the prerequisite software has been purchased, installed, and configured properly in order to satisfy the identified HPSS architecture. Refer to Section 2.3: *Prerequisite Software Considerations* on page 46 for more information on the HPSS prerequisite software requirements.

4. Determine the SFS space needed to satisfy the requirements of the HPSS system. In addition, verify that there is sufficient disk space to configure the space needed by SFS. Refer to Section 2.10.2: *HPSS Metadata Space* on page 106 for more discussions of SFS sizing required by HPSS.

5. Verify that each of the identified nodes has sufficient resources to handle the work loads and resources to be imposed on the node. Refer to Section 2.10.3: *System Memory and Disk Space* on page 132 for more discussions on the system resource requirements.

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

   - Define the HPSS file families. Refer to Section 2.9.4: *File Families* on page 105 for more information about configuring families.

   - Define filesets and junctions. Refer to Section 8.7: *Creating Filesets and Junctions* on page 465 for more information.

   - Define the HPSS storage classes. Refer to Section 2.9.1: *Storage Class* on page 93 for more information on the storage class configuration.

   - Define the HPSS storage hierarchies. Refer to Section 2.9.2: *Storage Hierarchy* on page 101 for more information on the storage hierarchy configuration.

   - Define the HPSS classes of service. Refer to Section 2.9.3: *Class of Service* on page 102 for more information on the class of service configuration.

   - Define the migration and purge policy for each storage class. Refer to Section 2.8.1: *Migration Policy* on page 82 and Section 2.8.2: *Purge Policy* on page 84 for more information on the Migration Policy and Purge Policy configuration, respectively.

   - Determine the HPSS storage space needed for each storage class. Refer to Section 2.10.1: *HPSS Storage Space* on page 106 for more information on the HPSS storage space considerations.

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

7. Define the location policy to be used. Refer to Section 2.8.6: *Location Policy* on page 89 for more information.

8. Define the accounting policy to be used. Refer to Section 2.8.3: *Accounting Policy and Validation* on page 85 for more information on the Accounting Policy configuration.

9. Define the logging policy for the each of the HPSS servers. Refer to Section 2.8.5: *Logging Policy* on page 89 for more information on the Logging Policy configuration.

10. Define the security policy for the HPSS system. Refer to Section 2.8.4: *Security Policy* on page 87 for more information on the Security Policy for HPSS.

11. Identify whether or not a Gatekeeper Server will be required. It is required if a site wants to do Account Validation or Gatekeeping. Refer to Section 2.8.3: *Accounting Policy and Validation* on page 85 for more information on Account Validation and Section Section 2.8.7: *Gatekeeping* on page 90 for more information on Gatekeeping.

2.1.2 Purchasing Hardware and Software

It is recommended that you not purchase hardware until you have planned your HPSS configuration. Purchasing the hardware prior to the planning process may result in performance issues and/or utilization issues that could easily be avoided by simple advance planning.

If deciding to purchase Sun or SGI servers for storage purposes, note that OS limitations will only allow a static number of raw devices to be configured per logical unit (disk drive or disk array). Solaris currently allows only eight partitions per logical unit (one of which is used by the OS). Irix currently allows only sixteen partitions per logical unit. These numbers can potentially impact the utilization of a disk drive or disk array.

Refer to Section 2.10: *HPSS Sizing Considerations* on page 105 for more information on calculating the number and size of raw devices that will be needed to meet your requirements.

Refer to Section 2.3: *Prerequisite Software Considerations* on page 46 for more information on the required software that will be needed to run HPSS.

When you finally have an HPSS configuration that meets your requirements, it is then time to purchase the necessary hardware and software.

2.1.3 HPSS Operational Planning

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

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

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

   - Each SSM user (usually an HPSS administrator or an operator) should be assigned an appropriate SSM security level. The SSM security level defines what functions each SSM user can perform on HPSS through SSM. Refer to Section 11.1: *SSM Security* on page 275 of the *HPSS Management Guide* for more information on setting up the security level for an SSM user.

2. Define the site policy and procedure for repacking and reclaiming HPSS tape volumes. This policy should consider how often to repack and reclaim HPSS volumes in the configured tape storage classes. In addition, the policy must consider when the repack and reclaim should be performed to minimize the impact on the HPSS normal operations. Refer to Section 3.7: *Repacking HPSS Volumes* on page 74 and Section 3.8: *Reclaiming HPSS Tape Virtual Volumes* on page 76 (both in the *HPSS Management Guide*) for more information.

3. Define the site policy and procedure for generating the accounting reports. This policy should consider how often an accounting report needs to be generated, how to use the accounting information from the report to produce the desired cost accounting, and whether the accounting reports need to be archived. Refer to Section 2.8.3: *Accounting Policy and Validation* on page 85 and Section 6.6.3: *Configure the Accounting Policy* on page 289 for more information on defining an Accounting Policy and generating accounting reports.

4. Determine whether or not gatekeeping (monitoring or load-balancing) will be required. If so, define and write the site policy code for gatekeeping. Refer to Section 2.8.7: *Gatekeeping* on page 90 for more information on Gatekeeping and *HPSS Programmers Reference, Volume 1* for guidelines on implementing the Site Interfaces for the Gatekeeping Service.

2.1.4 HPSS Deployment Planning

The successful deployment of an HPSS installation is a complicated task which requires reviewing customer/system requirements, integration of numerous products and resources, proper training of users/administrators, and extensive integration testing in the customer environment. Early on, a set of meetings and documents are required to ensure the resources and intended configuration of those resources at the customer location can adequately meet the expectations required of the system. The next step in this process is to help the customer coordinate the availability and readiness of the resources before the actual installation of HPSS begins. Each one of the products/resources that HPSS uses must be installed, configured and tuned correctly for the final system to function and perform as expected. Once installed, a series of tests must be planned and performed to verify that the system can meet the demands of the final production environment. And finally, proper training of those administrating the system, as well as those who will use it, is necessary to make a smooth transition to production.

To help the HPSS system administrators in all of these tasks, a set of procedures in addition to this document have been developed to assist those involved in this process. The HPSS Deployment Process contains a detailed outline of what is required to bring an HPSS system online from an initial introduction and review of the customer environment to the time the system is ready for production use. The deployment procedures include a time line plus checklist that the HPSS customer installation/system administration team should use to keep the deployment of an HPSS system on track. This is the same guide that the HPSS support/deployment team uses to monitor and check the progress of an installation.

2.2 Requirements and Intended Usages for HPSS

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

2.2.1 Storage System Capacity

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

- The amount of storage data from anticipated new user accounts.
- The amount of new storage data resulting from normal growth of current accounts.
- The amount of storage space needed to support normal storage management such as migration and repack.
- The amount of storage data to be duplicated.

Another component of data storage is the amount of metadata space needed for user directories and other HPSS metadata. Much of this data must be duplicated (called "mirroring") in real time on separate storage devices. Refer to Section 2.10.1 and Section 2.10.2 for more information on determining the needed storage space and metadata space.

2.2.2 Required Throughputs

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

2.2.3 Load Characterization

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

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

2.2.4 Usage Trends

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

2.2.5 Duplicate File Policy

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

2.2.6 Charging Policy

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

2.2.7 Security

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

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

Each HPSS server authorizes and enforces access to its interfaces through access control lists attached to an object (named Security) that is contained in its CDS directory. To be able to modify server state, control access is required. Generally, this is only given to the DCE principal associated with the HPSS system administrative component. Additional DCE principals can be allowed or denied access by setting permissions appropriately. See Section 6.5.1.2: *Server CDS Security ACLs* on page 278 for more information.

Security auditing in each server may be configured to record all, none, or some security event. Some sites may choose to log every client connection; every bitfile creation, deletion, and open; and every file management operation. Other sites may choose to log only errors. See the security information fields in the general server configuration (Section 6.5.1: *Configure the Basic Server Information* (page 263)) for more details.

User access to HPSS interfaces depends on the interface being used. Access through DFS and the native Client API uses the DCE authentication and authorization services described above. Access through the Non-DCE Client API is configurable as described in Section 6.8.11: *Non-DCE Client Gateway Specific Configuration* on page 367. Access through NFS is determined based on how the HPSS directories are exported. Refer to Section 12.2: *HPSS Utility Manual Pages* on page 293 of the *HPSS Management Guide* for more information on NFS exports and the **nfsmap** utility (Section 12.2.42: *nfsmap — Manipulate the HPSS NFS Daemon's Credentials Map* (page 418) in the *HPSS Management Guide*). FTP or Parallel FTP access may utilize an FTP password file or may utilize the DCE Registry. Additional FTP access is available using **Ident**, Kerberos GSS credentials, or DCE GSS credentials. The **Ident** and GSS authentication methods require running the **hpss_pftpd_amgr** server and an associated authentication manager in place of the standard **hpss_pftpd**. Refer to the FTP section of the *HPSS User’s Guide* for additional details.

2.2.7.1 Cross Cell Access

DCE provides facilities for secure communication between multiple DCE cells (realms/domains) referred to as Trusted “Cross Cell”. These features use the DCE facilities to provide a trusted environment between cooperative DCE locations. HPSS uses the DCE Cross Cell features for authentication and to provide HPSS scalability opportunities. The procedures for inter-connecting DCE cells are outlined in Section Chapter 11: *Managing HPSS Security and Remote System Access* on page 275 of the *HPSS Management Guide.* The HPSS DFS facilities, Federated Name Space, and HPSS Parallel FTP can utilize the DCE and HPSS Cross Cell features.

The Generic Security Service (GSS) FTP (available from the Massachusetts Institute of Technology, MIT) and Parallel FTP applications may also take advantage of the Cross Cell features for authentication and authorization. Use of Kerberos / DCE Credentials with the HPSS Parallel FTP Daemon requires using the **hpss_pftpd_amgr** server and an associated authentication manager in place of the standard **hpss_pftpd**. Both the GSSFTP from MIT and the **krb5_gss_pftp_client** applications may use the Cross Cell features (subject to certain caveats! – See FTP documentation for details.) The **krb5_gss_pftp_client** is *not distributed* by the HPSS project; but may be built from distributed source by the HPSS sites. It is the site's responsibility to obtain the necessary Kerberos components (header files, library files, etc.)

Local Location Servers exchange server information with remote Location Servers for both inter-subsystem and HPSS Cross Cell communications. It is necessary to add a record to the Remote Sites metadata file for each remote HPSS system you are connecting to in order to tell each local Location Server how to contact its remote counterpart. To do this, the Local HPSS will need to obtain all of the information contained in the Remote HPSS Site Identification fields from the remote site's Location Policy screen. Similarly, information from the Local site's Location Policy screen will be required by the Remote HPSS site in the Remote Site's policy file.

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

2.2.8 Availability

The High Availability component allows HPSS to inter operate with IBM’s HACMP software. When configured with the appropriate redundant hardware, this allows failures of individual system components (network adapters, core server nodes, power supplies, etc.) to be overcome, returning the system to resume servicing requests with minimal downtime. For more information on the High Availability component, see Appendix G: *High Availability* (page 535).

2.3 Prerequisite Software Considerations

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

2.3.1 Overview

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

2.3.1.1 DCE

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

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

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

If the DCE cell used for HPSS is expected to communicate with other DCE cells the *DCE Global Directory Service* and *DCE Global Directory Client* will also be required.

The following nodes must have DCE installed:

- Nodes that run DCE servers
- Nodes that run Encina SFS
- Nodes that run HPSS servers, including DCE Movers
- Nodes that run site-developed client applications that link with the HPSS Client API library
- Nodes that run NFS clients with DCE Kerberos authentication

The following nodes do not require DCE:

- Nodes that only run Non-DCE Mover
- Nodes that only run FTP, PFTP, and NFS clients not using DCE authentication
- Nodes that run the Non-DCE Client API.

Specific DCE product versions relative to specific versions of operating systems can be found in Sections 2.3.2.1 through 2.3.4.2.

When laying out a DCE cell, it is best to get the CDS daemon, SEC daemon, and the HPSS core servers as “close” to each other as possible due to the large amount of communication between them. If it is feasible, put them all on the same node. Otherwise, use a fast network interconnect, and shut down any slower interfaces using the **RPC_UNSUPPORTED_NETIFS** environment variable in the **/etc/environments** file (AIX) or the **/etc/default/init** file (Sun). If the DCE daemons are on different nodes, at least try to put them on the same subnet.

2.3.1.2 DFS

HPSS uses the Distributed File System (DFS) from the Open Group to provide distributed file system services.

The following nodes must have DFS installed:

- Nodes that run DFS servers
- Nodes that run DFS clients
- Nodes that run the HPSS/DFS HDM servers

2.3.1.3 XFS

HPSS uses the open source Linux version of SGI's XFS filesystem as a front-end to an HPSS archive.

The following nodes must have XFS installed:

- Nodes that run the HPSS/XFS HDM servers

2.3.1.4 Encina

HPSS uses the Encina distributed transaction processing software developed by Transarc Corporation, including the Encina Structured File Server (SFS) to manage all HPSS metadata.

The Encina software consists of the following three components:

- Encina Server
- Encina Client
- Encina Structured File Server

The following nodes must have all Encina components installed:

- Nodes that run Encina SFS

The following nodes must have the Encina Client component installed:

- Nodes that run one or more HPSS servers (with the exception of nodes that run only the non-DCE Mover)
- Nodes that run an end-user client application that links with the HPSS Client API library

The following nodes do not require Encina:

- Nodes that only run Non-DCE Mover
- Nodes that only run FTP, PFTP, and NFS clients
- Nodes that only run Non-DCE Clients

Specific Encina product versions relative to specific versions of operating systems can be found in Sections 2.3.2.1 through 2.3.4.2.

2.3.1.5 Sammi

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

The HPSS Server Sammi License and, optionally, the HPSS Client Sammi License available from the Kinesix Corporation are required. The Sammi software must be installed separately prior to the HPSS installation. In addition, the Sammi license(s) for the above components must be obtained from Kinesix and set up as described in Section 4.5.3: *Set Up Sammi License Key* (page 213) before running Sammi.

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

1. Operating System and System Software Requirements:
    - ◆ AIX 5.1 or Solaris 5.8
    - ◆ TCP/IP and Sun NFS/RPC
    - ◆ X Window System (X11 Release 5) and Motif
2. System Space Requirements:
    - ◆ 32 MB of memory
    - ◆ 40 MB of swap space
3. Hardware Requirements:
    - ◆ A graphic adapter with at least 256 colors
    - ◆ A monitor with a minimal resolution of 1280 x 1024 pixels
    - ◆ A mouse (two or three buttons)

Specific Sammi product versions relative to specific versions of operating systems can be found in Sections 2.3.2.1 through 2.3.4.2.

2.3.1.6 Miscellaneous

- Sites needing to compile the HPSS FTP source code must have the **yacc** compiler.
- Sites needing to compile the HPSS NFS Daemon source code must have the NFS client software.
- Sites needing to compile the HPSS Generic Security Service (GSS) security code must have the X.500 component of DCE.
- Sites needing to compile the HDM code, build the **cs/xdr/dmapi** libraries, or run the SFS Backup scripts must have the **perl** compiler installed (**perl 5.6.0.0** or later).
- Sites needing to compile the MPI-IO source code must have a compatible host MPI installed, as described in Section 7.5: *MPI-IO API Configuration* on page 434. Sites needing to compile the Fortran interfaces for MPI-IO must have a Fortran77 standard compiler that accepts C preprocessor directives. Sites needing to compile the C++ interfaces for MPI-IO must have a C++ standard compiler that accepts **namespace**. Note that the Fortran and

C++ interfaces may be selectively disabled in **Makefile.macros** if these components of MPI-IO cannot be compiled.

- Sites using the Command Line SSM utility, **hpssadm**, will require Java 1.3.0 and JSSE (the Java Secure Sockets Extension) 1.0.2. These are required not only for **hpssadm** itself but also for building the SSM Data Server to support **hpssadm**.

2.3.2 Prerequisite Summary for AIX

2.3.2.1 HPSS Server/Mover Machine - AIX

1. AIX 5.1 (For High Availability core servers, patch level 2 or later is required)
2. DCE for AIX Version 3.2 (patch level 1 or later)
3. DFS for AIX Version 3.1 (patch level 4 or later) if HPSS HDM is to be run on the machine
4. TXSeries 5.0 for AIX (no patch available at release time)
5. HPSS Server Sammi License (Part Number 01-0002100-A, version 4.6.3.5.5 for AIX 4.3.3) from Kinesix Corporation for each HPSS license which usually consists of a production and a test system. In addition, an HPSS Client Sammi License (Part Number 01-0002200-B, version 4.6.3.5.5 for AIX 4.3.3) is required for each additional, concurrent SSM user. Refer to Section 2.3.1.5 for more information on Sammi prerequisite and installation requirements. This is only needed if Sammi is to be run on the machine.
6. High Performance Parallel Interface Drivers Group (HiPPI/6000), if HiPPI is required
7. C compiler for AIX, version 5.0
8. Data Encryption Standard Library, version 4.3.0.1

2.3.2.2 HPSS Non-DCE Mover/Client Machine

1. AIX 5.1
2. C compiler for AIX, version 5.0

2.3.3 Prerequisite Summary for IRIX

2.3.3.1 HPSS Non-DCE Mover/Client Machine

1. IRIX 6.5 (with latest/recommended patch set)
2. HiPPI drivers, if HiPPI network support is required.
3. C compiler

2.3.4 Prerequisite Summary for Solaris

2.3.4.1 HPSS Server/Mover Machine

1. Solaris 5.8
2. DCE for Solaris Version 3.2 (patch level 1 or later)
3. DFS for Solaris Version 3.1 (patch level 4 or later) if HPSS HDM is to be run on the machine
4. TXSeries 4.3 for Solaris from WebSphere 3.5 (patch level 4 or later)
5. HPSS Server Sammi License (Part Number 01-0002100-A, version 4.7) from Kinesix Corporation for each HPSS license which usually consists of a production and a test system. In addition, an HPSS Client Sammi License (Part Number 01-0002200-B, version 4.7) is required for each additional, concurrent SSM user. Refer to Section 2.3.1.5 for more information on Sammi prerequisite and installation requirements. This is only needed if Sammi is to be run on the machine.
6. High Performance Parallel Interface Drivers Group (HiPPI/6000), if HiPPI is required
7. C compiler
8. The following Solaris 5.8 Supplemental Encryption packages are required:
   - SUNWcrman
   - SUNWcry
   - SUNWcry64
   - SUNWcryr
   - SUNWcryrx

2.3.4.2 HPSS Non-DCE Mover/Client Machine

1. Solaris 5.8
2. C compiler

2.3.5 *Prerequisite Summary for Linux and Intel*

2.3.5.1 *HPSS/XFS HDM Machine*

1. Linux kernel 2.4.18 or later (Available via FTP from `ftp://www.kernel.org/pub/linux/kernel/v2.4`)

2. Linux XFS 1.1 (Available via FTP as a 2.4.18 kernel patch at `ftp://oss.sgi.com/projects/xfs/download/Release-1.1/kernel_patches`)

3. Userspace packages (Available via FTP as RPMs or tars from `ftp://oss.sgi.com/projects/xfs/download/Release-1.1/cmd_rpms` and `ftp://oss.sgi.com/projects/xfs/download/Release-1.1/cmd_tars`, respectively):

   - acl-2.0.9-0
   - acl-devel-2.0.9-0
   - attr-2.0.7-0
   - attr-devel-2.0.7-0
   - dmapi-2.0.2-0
   - dmapi-devel-2.0.2-0
   - xfsdump-2.0.1-0
   - xfsprogs-2.0.3-0
   - xfsprogs-devel-2.0.3-0

4. HPSS Kernel Patch

   It will also be necessary to apply patch **xfs-2.4.18-1** to the kernel after applying the XFS 1.1 patch. This addresses a problem found after XFS 1.1 was released. The procedure for applying this patch is outlined in Section 3.9.1: *Apply the HPSS Linux XFS Patch* on page 194.

5. C compiler

   For XFS HDM machines, it will be necessary to update the main kernel makefile (usually found in **/usr/src/linux/Makefile**). The default compiler that the makefile uses, **gcc**, needs to be replaced with **kgcc**. Simply comment-out the line that reads:

   ```
   CC                    = $(CROSS_COMPILE)gcc
   ```

   And uncomment the line that reads:

   ```
   CC                    = $(CROSS_COMPILE)kgcc
   ```

2.3.5.2 HPSS Non-DCE Mover Machine

1. Linux kernel 2.4.18

2. HPSS KAIO Patch

   It will be necessary to apply the HPSS KAIO kernel patch (**kaio-2.4.18-1**). This patch adds asynchronous I/O support to the kernel which is required for the Mover. The procedure for applying this patch is outlined in Section 3.10: *Setup Linux Environment for Non-DCE Mover* on page 195

2.3.5.3 HPSS Non-DCE Client API Machine

1. Redhat Linux, version 7.1 or later

2. C compiler

2.3.5.4 *HPSS* pftp *Client Machine*

1. Redhat, version 7.1 or later

2. C compiler

2.4 Hardware Considerations

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

2.4.1 Network Considerations

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

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

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

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

Along with shared memory, HPSS also supports a Local File Transfer data path, for client transfers that involve HPSS Movers that have access to the client's file system. In this case, the HPSS Mover can be configured to transfer the data directly to or from the client's file.

The DCE RPC mechanism used for HPSS control communications can be configured to utilize a combination of TCP/IP and User Datagram Protocol (UDP)/IP (i.e., one of the two protocols or both of the protocols). However, during the development and testing of HPSS, it was discovered that using TCP/IP would result in increasing and unbounded memory utilization in the servers over time (which would eventually cause servers to terminate when system memory and paging space resources were exhausted). Because of this behavior when using TCP/IP for the DCE RPC mechanism, the HPSS servers should only utilize UDP/IP for control communications. The default HPSS installation/configuration process will enable only UDP/IP for DCE RPC communications with the HPSS servers. See Section 5.3: *Define the HPSS Environment Variables* (page 216) for further details (specifically the environment variable **RPC_SUPPORTED_PROTSEQS**).

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

2.4.2 Tape Robots

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

When using RAIT PVRs, an extra level of complexity is added by the virtualization of the physical drives into logical drives. Locking a given logical drive in HPSS would not necessarily guarantee that HPSS will not access it via another logical drive. It is therefore not recommended to shared drives managed by a RAIT PVR.

The STK RAIT PVR cannot be supported at this time since STK has not yet made RAIT generally available.

Generally, only one HPSS PVR is required per robot. However, it is possible for multiple PVRs to manage a single robot in order to provide drive and tape pools within a robot. The drives in the robot must be partitioned among the PVRs and no drive should be configured in more than one

PVR. Each tape is assigned to exactly one PVR when it is imported into the HPSS system and will only be mounted in drives managed by that PVR.

The tape libraries supported by HPSS are:

- IBM 3494 / 3495
- IBM 3584
- STK Tape Libraries that support ACSLS
- ADIC AML

2.4.2.1 IBM 3494/3495

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

2.4.2.2 IBM 3584 (LTO)

The IBM 3584 Tape Library and Robot must be attached to an AIX workstation either through an LVD Ultra2 SCSI or HVD Ultra SCSI interface. The library shares the same SCSI channel as the first drive, so in actuality the first drive in the 3584 must be connected to the AIX workstation. This workstation must be an HPSS node running the PVR. The latest level of the AIX tape driver must be installed on this machine.

2.4.2.3 STK

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

The STK RAIT PVR cannot be supported at this time since STK has not yet made RAIT generally available.

2.4.2.4 ADIC AML

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

2.4.2.5 *Operator Mounted Drives*

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

2.4.3 *Tape Devices*

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

- IBM 3480, 3490, 3490E, 3590, 3590E and 3590H are supported via SCSI attachment.
- IBM 3580 devices are supported via SCSI attachment.
- StorageTek 9840, 9940, 9940B, RedWood (SD-3), and TimberLine (9490) are supported via s SCSI attachment. Support for STK RAIT drives is included.

The STK RAIT PVR cannot be supported at this time since STK has not yet made RAIT generally available.

- Ampex DST-312 and DST-314 devices are supported via SCSI attachment.
- Sony GY-8240 devices are supported via Ultra-Wide Differential SCSI attachment.

For platform and driver information for these drives, see Section 6.9.2: *Supported Platform/Driver/ Tape Drive Combinations* on page 411

2.4.3.1 *Multiple Media Support*

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

Table 2-1 Cartridge/Drive Affinity Table

Cartridge Type	Drive Preference List
AMPEX DST-312	AMPEX DST-312 AMPEX DST-314
AMPEX DST-314	AMPEX DST-314

Table 2-1 Cartridge/Drive Affinity Table

Cartridge Type	Drive Preference List
Single-Length 3590	Single-Length 3590 Double-Length 3590 Single-Length 3590E Double-Length 3590E Single-Length 3590H Double-Length 3590H
Double-Length 3590	Double-Length 3590 Double-Length 3590E Double-Length 3590H
Single-Length 3590E	Single-Length 3590E Double-Length 3590E Double-Length 3590H
Double-Length 3590E	Double-Length 3590E Double-Length 3590H
Single-Length 3590H	Single-Length 3590H Double-Length 3590H
Double-Length 3590H	Double-Length 3590H

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

2.4.4 Disk Devices

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

Locally attached disk devices that are supported include those devices attached via either SCSI, SSA or Fibre Channel. For these devices, operating system disk partitions of the desired size must be created(e.g.,AIX logical volume or Solaris disk partition), and the raw device name must be used when creating the Mover Device configuration (see Chapter 5: *Managing HPSS Devices and Drives* (page 95) in the *HPSS Management Guide* for details on configuring storage devices).

2.4.5 Special Bid Considerations

Some hardware and hardware configurations are supported only by special bid. These items are listed below:

- IBM 3580 Drive
- IBM 3584 Tape Libraries
- ADIC AML Tape Libraries

- High Availability HPSS configuration

2.5 HPSS Interface Considerations

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

2.5.1 Client API

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

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

The Client API allows clients to specify the amount of data to be transferred with each request. The amount requested can have a considerable impact on system performance and the amount of metadata generated when writing directly to a tape storage class. See Sections 2.8.6 and 2.11 for further information.

The details of the Application Programming Interface are described in the *HPSS Programmer's Reference Guide, Volume 1.*

2.5.2 Non-DCE Client API

The Non-DCE Client API provides the same user function calls as the Client API for client applications who are running on platforms without DCE or Encina with the following exceptions:

- ACL calls are not supported.
- Calls are not transactional.
- It implements its own security interface.

Client authentication is performed in one of three ways: remote DCE login, Kerberos authentication or no authentication (in case of trusted clients). In order to use the NDAPI, the client application must link the Non-DCE Client API library, and the target HPSS system must have a Non-DCE Client Gateway server configured and running. The application calls to the library are then sent over the network to the NDCG who executes the appropriate Client API calls, returning the results to the client application. Kerberos must be installed on the client and the gateway in order to use the Kerberos security feature.

2.5.3 FTP

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

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

The configuration of the FTP server allows the size of the buffer to be used for reading and writing HPSS files to be specified. The buffer size selected can have a considerable impact on both system performance and the amount of metadata generated when writing directly to a tape Storage Class. See Sections 2.8.6 and 2.11 for further information.

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

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

2.5.4 Parallel FTP

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

The PFTP protocol is supported by the HPSS FTP Daemon. Refer to Section 7.3: *FTP Daemon Configuration* (page 422) for configuration information. No additional configuration of the FTP Daemon is required to support PFTP clients.

The client side executable for PFTP is **pftp_client**. **pftp_client** supports TCP based transfers. Because the client executable is a superset of standard FTP, standard FTP requests can be issued as well as the PFTP extensions.

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

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

2.5.5 NFS

The HPSS NFS interface implements versions 2 and 3 of the Network File System (NFS) protocol for access to HPSS name space objects and bitfile data. The NFS protocol was developed by Sun Microsystems to provide transparent remote access to shared file systems over local area networks. Because the NFS designers wanted a robust protocol that was easy to port, NFS is implemented as a stateless protocol. This allows use of a connectionless networking transport protocol (UDP) that requires much less overhead than the more robust TCP. As a result, client systems must time out requests to servers and retry requests that have timed out before a response is received. Client time-out values and retransmission limits are specified when a remote file system is mounted on the client system.

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

The HPSS NFS interface does not support Access Control Lists (ACLs), so don't attempt to use them with NFS-exported portions of the HPSS name space.

Because of the distributed nature of HPSS and the potential for data being stored on tertiary storage, the time required to complete an NFS request may be greater than the time required for non-HPSS NFS servers. The HPSS NFS server implements caching mechanisms to minimize these delays, but time-out values (**timeo** option) and retransmission limits (**retrans** option) should be adjusted accordingly. A time-out value of no less than 10 and a transmission limit of no less than 3 (the default) are recommended. The values of **timeo** and **retrans** should be coordinated carefully with the daemon's disk and memory cache configuration parameters, in particular, the thread interval and touch interval. The larger these values are, the larger the **timeo** and **retrans** times will need to be to avoid timeouts.

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

2.5.6 MPI-IO API

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

The MPI-IO API is layered on top of a host MPI library. The characteristics of a specific host MPI are designated through the **include/mpio_MPI_config.h**, which is generated at HPSS creation time from the **MPIO_MPI** setting in the **Makefile.macros**. The configuration for MPI-IO is described in Section 7.5: *MPI-IO API Configuration* on page 434

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

The threads used by MPI-IO must be compatible with the HPSS CLAPI or NDAPI threads use. Threaded applications must be loaded with the appropriate threads libraries.

This raises some thread-safety issues with the Sun and MPICH hosts. Neither of these host MPIs support multithreading, per se. They are in conformance with the MPI-1 standard which prescribes that an implementation is thread-safe provided only one thread makes MPI calls. With HPSS MPI-IO, multiple threads will make MPI calls. HPSS MPI-IO attempts to impose thread-safety on these hosts by utilizing a global lock that must be acquired in order to make an MPI call. However, there are known problems with this approach, and the bottom line is that until these hosts provide true thread-safety, the potential for deadlock within an MPI application will exist when using HPSS MPI-IO in conjunction with other MPI operations. See the *HPSS Programmers Reference Guide, Volume 1,* Release 4.5 for more details.

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

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

2.5.7 DFS

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

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

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

HPSS and DFS define disk partitions differently from one another. In HPSS, the option for how files are mirrored or archived is associated with a fileset. Recall that in DFS, multiple filesets may reside on a single aggregate. However, the XDSM implementation provided in DFS generates events on a per-aggregate basis. Therefore, in DFS this option applies to all filesets on a given aggregate.

To use the DFS / HPSS interface on an aggregate, the aggregate must be on a processor that has Transarc's DFS SMT kernel extensions installed. These extensions are available for Sun Solaris and IBM AIX platforms. Once an aggregate has been set up, end users can access filesets on the aggregate from any machine that supports DFS client software, including PCs. The wait / retry logic in DFS client software was modified to account for potential delays caused by staging data from HPSS. Using a DFS client without this change may result in long delays for some IO requests.

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

2.5.8 XFS

XFS for Linux is an open source filesystem from SGI based on SGI's XFS filesystem for IRIX.

HPSS has the capability to backend XFS and transparently archive inactive data. This frees XFS disk to handle data that is being actively utilized, giving users the impression of an infinitely large XFS filesystem that performs at near-native XFS speeds.

It is well suited to sites with large numbers of small files or clients who wish to use NFS to access HPSS data. However, the files can only be accessed through XFS (or NFS via XFS) interfaces and cannot be accessed with HPSS utilities such as parallel FTP. Therefore, data transfer performance is limited to XFS speeds.

2.6 HPSS Server Considerations

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

2.6.1 Name Server

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

The total number of objects permitted in the name space is limited by the number of SFS records allocated to the NS. Refer to Section 2.10.2.4 for details on selecting the size of the name space. With this release of HPSS, provisions have been made for increasing the size of the name space by adding additional Name Servers, storage subsystems, or by junctioning to a Name Server in a different HPSS system (see Section 11.2.3: *Federated Name Space* on page 279 of the *HPSS Management Guide*). Refer to Section 10.7.3: *Name Server Space Shortage* (page 272) in the *HPSS Management Guide* for information on handling an NS space shortage.

The NS uses DCE threads to service concurrent requests. Refer to Section 6.5.1: *Configure the Basic Server Information* (page 263) for more information on selecting an appropriate number of DCE threads. The NS accepts requests from any client that is authenticated through DCE; however, certain NS functions can be performed only if the request is from a trusted client. Trusted clients are those clients for whom control permission has been set in their CDS ACL entry for the NS. Higher levels of trust are granted to clients who have both control and write permission set in their CDS ACL entry. Refer to Table 6-3: *Basic Server Configuration Variables* on page 266 for information concerning the CDS ACL for the Name Server.

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

Multiple Name Servers are supported, and each storage subsystem contains exactly one Name Server. Though the servers are separate, each Name Server in a given DCE cell must share the same metadata global file for filesets.

2.6.2 *Bitfile Server*

The Bitfile Server (BFS) provides a view of HPSS as a collection of files. It provides access to these files and maps the logical file storage into underlying storage objects in the Storage Servers. When a BFS is configured, it is assigned a server ID. This value should never be changed. It is embedded in the identifier that is used to name bitfiles in the BFS. This value can be used to link the bitfile to the Bitfile Server that manages the bitfile.

The BFS maps bitfiles to their underlying physical storage by maintaining mapping information that ties a bitfile to the storage server storage segments that contain its data. These storage segments are referenced using an identifier that contains the server ID of the Storage Server that manages the storage segment. For this reason, once a Storage Server has been assigned a server ID, this ID must never change. For additional information on this point, see Section 2.6.3: *Disk Storage Server* on page 64 and Section 2.6.4: *Tape Storage Server* on page 64. The relationship of BFS bitfiles to SS storage segments and other structures is shown in Figure 2-1 on page 63.

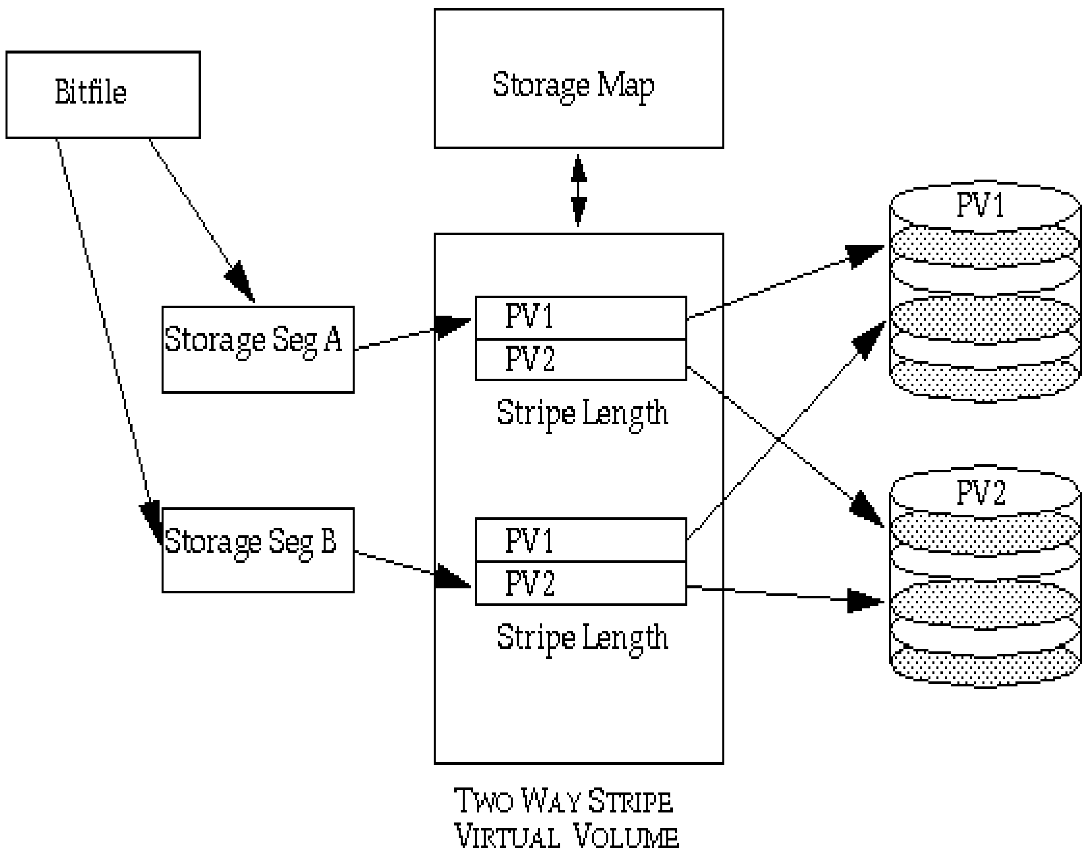


Figure 2-1 The Relationship of Various Server Data Structures

2.6.3 Disk Storage Server

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

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

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

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

The Disk Storage Server is designed to scale up its ability to manage disks as the number of disks increases. As long as sufficient memory and CPU capacity exist, threads can be added to the server to increase its throughput. Additional Storage Subsystems can also be added to a system, increasing concurrency even further.

2.6.4 Tape Storage Server

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

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

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

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

The Tape Storage Server is designed to scale up its ability to manage tapes as the number of tapes increases. As long as sufficient memory and CPU capacity exist, threads can be added to the server to increase its throughput. Additional Storage Subsystems can also be added to a system, increasing concurrency even further.

Note that the number of tape units the server manages has much more to do with the throughput of the server than the number of tapes the server manages. If the number of tape units in the system increases, adding a new Tape Storage Server to the system may be the best way to deal with the increased load.

2.6.5 Migration/Purge Server

The Migration / Purge Server (MPS) can only exist within a storage subsystem. Any storage subsystem which is configured to make use of a storage hierarchy which requires the *migration* and *purge* operations must be configured with one and only one MPS within that subsystem. The definition of storage hierarchies is global across all storage subsystems within an HPSS system, but a given hierarchy may or may not be enabled within a given subsystem. A hierarchy is enabled within a subsystem by using the storage subsystem configuration to enable one or more classes of service which reference that hierarchy. If a hierarchy is enabled within a subsystem, storage resources must be assigned to the storage classes in that hierarchy for that subsystem. This is done by creating resources for the Storage Servers in the given subsystem. If the hierarchy contains storage classes which require migration and purge, then an MPS must be configured in the subsystem. This MPS will manage migration and purge operations on only those storage resources within its assigned subsystem. Hence, in an HPSS system with multiple storage subsystems, there may be multiple MPSs, each operating on the resources within a particular subsystem.

MPS manages the amount of free space available in a storage class within its assigned storage subsystem by performing periodic migration and purge *runs* on that storage class. Migration copies data from the storage class on which it runs to one or more lower levels in the storage hierarchy. Once data has been migrated, a subsequent purge run will delete the data from the migrated storage class. Migration is a prerequisite for purge, and MPS will never purge data which has not previously been migrated. It is important to recognize that migration and purge policies determine *when* data is copied from a storage class and then *when* the data is deleted from that storage class; however, the *number* of copies and the *location* of those copies is determined solely by the storage hierarchy definition. Note that this is a major difference between release 4.2+ versions of the HPSS system and all previous releases.

Migration and purge must be configured for each storage class on which they are desired to run. Since the storage class definition is global across all storage subsystems, a storage class may not be selectively migrated and purged in different subsystems. Additionally, migration and purge operate differently on disk and tape storage classes. *Disk migration* and *disk purge* are configured on a disk storage class by associating a *migration policy* and a *purge policy* with that storage class. It is possible, but not desirable, to assign only a migration policy and no purge policy to a disk storage class; however, this will result in data being copied but never deleted. For tape storage classes, the migration and purge operations are combined, and are collectively referred to as *tape migration*. Tape migration is enabled by associating a migration policy with a tape storage class. Purge policies are not needed or supported on tape storage classes.

Once migration and purge are configured for a storage class (and MPS is restarted), MPS will begin scheduling migration and purge runs for that storage class. Migration on both disk and tape is run periodically according to the runtime interval configured in the migration policy. Disk purge runs are not scheduled periodically, but rather are started when the percentage of space used in the storage class reaches the threshold configured in the purge policy for that storage class. Remember that simply adding migration and purge policies to a storage class will cause MPS to begin running against the storage class, but it is also critical that the hierarchies to which that storage class belongs be configured with proper migration targets in order for migration and purge to perform as expected.

The purpose of disk migration is to make one or more copies of data stored in a disk storage class to lower levels in the storage hierarchy. BFS uses a metadata queue to pass *migration records* to MPS. When a disk file needs to be migrated (because it has been created, modified, or undergone a class of service change), BFS places a migration record on this queue. During a disk migration run on a given storage class, MPS uses the records on this queue to identify files which are *migration candidates*. Migration records on this queue are ordered by storage hierarchy, file family, and record create time, in that order. This ordering determines the order in which files are migrated.

MPS allows disk storage classes to be used atop multiple hierarchies (to avoid fragmenting disk resources). To avoid unnecessary tape mounts, it is desirable to migrate all of the files in one hierarchy before moving on to the next. At the beginning of each run MPS selects a starting hierarchy. This is stored in the MPS checkpoint metadata between runs. The starting hierarchy alternates to ensure that, when errors are encountered or the migration target is not 100 percent, all hierarchies are served equally. For example, if a disk storage class is being used in three hierarchies, 1, 2, and 3, successive runs will migrate the hierarchies in the following order: 1-2-3, 3-1-2, 2-3-1, 1-2-3, etc. A migration run ends when either the migration target is reached or all of the eligible files in every hierarchy are migrated. Files are ordered by file family for the same reason, although families are not checkpoints as hierarchies are. Finally, the record create time is simply the time at which BFS adds the migration record to the queue, and so files in the same storage class, hierarchy, and family tend to migrate in the order which they are written (actually the order in which the write completes).

When a migration run for a given storage class starts work on a hierarchy, it sets a pointer in the migration record queue to the first migration record for the given hierarchy and file family. Following this, migration attempts to build lists of 256 migration candidates. Each migration record read is evaluated against the values in the migration policy. If the file in question is eligible for migration its migration record is added to the list. If the file is not eligible, it is skipped *and it will not be considered again until the next migration run*. When 256 eligible files are found, MPS stops reading migration records and does the actual work to migrate these files. This cycle continues until either the migration target is reached or all of the migration records for the hierarchy in question are exhausted.

The purpose of disk purge is to maintain a given amount of free space in a disk storage class by removing data of which copies exist at lower levels in the hierarchy. BFS uses another metadata queue to pass *purge records* to MPS. A purge record is created for any disk file which may be removed from a given level in the hierarchy (because it has been migrated or staged). During a disk purge run on a given storage class, MPS uses the records on this queue to identify files which are *purge candidates*. The order in which purge records are sorted may be configured on the purge policy, and this determines the order in which files are purged. It should be noted that all of the options except *purge record create time* require additional metadata updates and can impose extra overhead on SFS. Also, unpredictable purge behavior may be observed if the purge record ordering is changed with existing purge records in the system until these existing records are cleared. Purge operates strictly on a storage class basis, and makes no consideration of hierarchies or file families. MPS builds lists of 32 purge records, and each file is evaluated for purge at the point when its purge record is read. If a file is deemed to be ineligible, *it will not be considered again until the next purge run*. A purge run ends when either the supply of purge records is exhausted or the purge target is reached.

There are two different tape migration algorithms, *tape volume migration* and *tape file migration*. The algorithm which is applied to a tape storage class is selected in the migration policy for that class.

The purpose of tape volume migration is to move data stored in a tape storage class either downward to the next level of the storage hierarchy (*migration*) or to another tape volume within the same storage class (*lateral move*) in order to empty tape volumes and allow them to be reclaimed. Unlike disk migration, the data is purged from the source volume as soon as it is copied. Tape volume migration operates on storage segments rather than files. A file may contain one or more segments. In order for a segment to be a candidate for tape volume migration it must reside on a virtual volume whose storage map is in the **EOM** state. Tape volume migration functions by selecting EOM volumes and moving or migrating the segments off of these volumes. When a volume is selected for tape volume migration, MPS repeatedly processes lists of up to 3200 segments on that volume until the volume is empty. Once all of the segments have been removed from a volume, that volume automatically moves into the **EMPTY** state and may be reclaimed for reuse. MPS continues this process until either the percentage of volumes specified in the migration policy are emptied or no more EOM volumes can be found. Segments on an EOM volume are evaluated for tape volume migration based on the values in the migration policy for that storage class. If a segment has been inactive for a sufficient length of time it will be migrated. If a segment has been active within the configured amount of time, or if any other segment in the selected segment's file has been active, the selected segment will be moved laterally. The **Migrate Volumes and Whole Files** option in the migration policy allows all of the segments belonging to a file to be migrated together, including those segments which reside on other, potentially non-EOM volumes than the EOM volume which is being processed. This option tends to keep all of the segments belonging to a given file at the same level in the hierarchy. If a segment is selected for migration by MPS, then all other segments belonging to the same file, regardless of their location, will be migrated during the same migration run. If any of the segments in the file are active then none of them, including the segment on the selected EOM volume, will be allowed to migrate. Rather, the selected segment will be moved laterally and none of the additional segments will be moved at all.

Tape file migration can be thought of as a hybrid between the disk and tape volume migration algorithms. Disk migration is a file based algorithm which is strictly concerned with making one or more copies of disk files. Tape volume migration is only concerned with freeing tape volumes by moving data segments from sparsely filled volumes either laterally or vertically. Tape file migration is a file-based tape algorithm which is able to make a single copy of tape files to the immediately lower level in the hierarchy. Similarly to disk migration, tape copies are made roughly in file creation order, but the order is optimized to limit the number of tape mounts.

As with disk files, BFS creates migration records for tape files in storage classes which are using tape file migration. MPS reads these migration records in the same manner as it does for disk. Within a given storage class, files are always migrated in hierarchy and family order. Hierarchies are checkpointed in the same way as disk. Files are roughly migrated by creation time (the time at which the first file write completed), but priority is given to migrating all of the files off of the current source volume over migrating files in time order.

When a tape file migration run begins, all of the eligible migration records for the storage class are read. For each migration record, the tape volume containing the corresponding file is identified. A list of all of the source tape volumes which are to be migrated in the current run is created. MPS then begins creating threads to perform the actual file migrations. A thread is created for each source volume up to the limit specified by **Request Count** in the migration policy. These threads then read the migration records corresponding to their assigned volumes and migrate each file. The migration threads end when the supply of migration records is exhausted. As each thread ends, MPS starts another thread for the next source tape volume to be migrated. The migration run ends when all volumes in all hierarchies have been migrated.

MPS provides the capability of generating migration / purge report files that document the activities of the server. The specification of the UNIX report file name prefix in the MPS server specific configuration enables the server to create these report files. It is suggested that a complete path be provided as part of this file name prefix. Once reporting is enabled, a new report file is started every 24 hours. The names of the report files are made up of the UNIX file name prefix from the server specific configuration, plus a year-month-day suffix. With reporting enabled, MPS will generate file-level migration and purge report entries in real time. These report files can be interpreted and viewed using the **mps_reporter** utility. Since the number and size of the report files grow rapidly, each site should develop a cron job that will periodically remove the reports that are no longer needed.

MPS uses threads to perform the migration and purge operations and to gather the storage class statistics from the Storage Servers. In particular, MPS spawns one thread for each disk or tape storage class on which migration is enabled and one thread for each disk storage class on which purge is enabled. These threads are created at startup time and exist for the life of the MPS. During disk migration runs, MPS spawns an additional number of temporary threads equal to the product of the number of copies being made (determined by the storage hierarchy configuration) and the number of concurrent threads requested for migration (configured in the Request Count field in the migration policy). During tape migration runs, MPS spawns one temporary thread for each Tape Storage Server within its configured subsystem. These threads exist only for the duration of a disk or tape migration run. Purge does not use any temporary threads. MPS uses a single thread to monitor the usage statistics of all of the storage classes. This thread also exists for the life of the MPS.

MPS provides the information displayed in the **HPSS Active Storage Classes** window in SSM. Each MPS contributes storage class usage information for the resources within its storage subsystem. MPS accomplishes this by polling the Storage Servers within its subsystem at the interval specified in the MPS server specific configuration. The resulting output is one line for each storage class for each storage subsystem in which that class is enabled. The MPS for a subsystem does not report on classes which are not enabled within that subsystem. MPS also activates and deactivates the warning and critical storage class thresholds.

Because the MPS uses the BFS and any Storage Servers within its assigned storage subsystem to perform data movement between hierarchy levels, the BFS and the Storage Servers must be running in order for the MPS to perform its functions. In addition, the MPS requires that the Storage Servers within its subsystem be running in order to report storage class usage statistics.

2.6.6 Gatekeeper

Each Gatekeeper may provide two main services:

1. Providing sites with the ability to schedule the use of HPSS resources using Gatekeeping Services.

2. Providing sites with the ability to validate user accounts using the Account Validation Service.

If the site doesn't want either service, then it is not necessary to configure a Gatekeeper into the HPSS system.

Sites can choose to configure zero (0) or more Gatekeepers per HPSS system. Gatekeepers are associated with storage subsystems. Each storage subsystem can have zero or one Gatekeeper associated with it and each Gatekeeper can support one or more storage subsystems. Gatekeepers are associated with storage subsystems using the **Storage Subsystem Configuration** screen (see Section 6.4: *Storage Subsystems Configuration* on page 259). If a storage subsystem has no Gatekeeper, then the Gatekeeper field will be blank. A single Gatekeeper can be associated with every storage subsystem, a group of storage subsystems, or one storage subsystem. A storage subsystem can NOT use more than one Gatekeeper.

Every Gatekeeper Server has the ability to supply the Account Validation Services. A bypass flag in the Accounting Policy metadata indicates whether or not Account Validation for an HPSS system is on or off. Each Gatekeeper Server will read the Accounting Policy metadata file, so if multiple Gatekeeper Servers are configured and Account Validation has been turned on, then any Gatekeeper Server can be chosen by the Location Server to fulfill Account Validation requests.

Every Gatekeeper Server has the ability to supply the Gatekeeping Service. The Gatekeeping Service provides a mechanism for HPSS to communicate information through a well-defined interface to a policy software module to be completely written by the site. The site policy code is placed in a well-defined site shared library for the gatekeeping policy (**/opt/hpss/lib/ libgksite.[a|so]**) which is linked to the Gatekeeper Server. The gatekeeping policy shared library contains a default policy which does NO gatekeeping. Sites will need to enhance this library to implement local policy rules if they wish to monitor and/or load balance requests.

The gatekeeping site policy code will determine which types of requests it wants to monitor (authorized caller, create, open, and stage). Upon initialization, each BFS will look for a Gatekeeper Server in the storage subsystem metadata. If no Gatekeeper Server is configured for a particular storage subsystem, then the BFS in that storage subsystem will not attempt to connect to any Gatekeeper Server. If a Gatekeeper Server is configured for the storage subsystem that the BFS is configured for, then the BFS will query the Gatekeeper Server asking for the monitor types by calling a particular Gatekeeping Service API which will in turn call the appropriate Site Interface which each site will write the code to determine which types of requests it wishes to monitor. This query by the BFS will occur each time the BFS (re)connects to the Gatekeeper Server. The BFS will need to (re)connect to the Gatekeeper whenever the BFS or Gatekeeper Server is restarted. Thus if a site wants to change the types of requests it is monitoring, then it will need to restart the Gatekeeper Server and BFS.

If multiple Gatekeeper Servers are configured for gatekeeping, then the BFS that controls the file being monitored will contact the Gatekeeper Server that is located in the same storage subsystem. Conversely if one Gatekeeper Server is configured for gatekeeping for all storage subsystems, then each BFS will contact the same Gatekeeper Server.

A Gatekeeper Server registers five different interfaces: Gatekeeper Services, Account Validation Services, Administrative Services, Connection Manager Services, and Real Time Monitoring Services. When the Gatekeeper Server initializes, it registers each separate interface. The Gatekeeper Server specific configuration SFS file will contain any pertinent data about each interface.

The Gatekeeper Service interface provides the Gatekeeping APIs which calls the site implemented Site Interfaces. The Account Validation Service interface provides the Account Validation APIs. The Administrative Service provides the server APIs used by SSM for viewing, monitoring, and setting server attributes. The Connection Manager Service provides the HPSS DCE connection management interfaces. The Real Time Monitoring Service interface provides the Real Time Monitoring APIs.

The Gatekeeper Service Site Interfaces provide a site the mechanism to create local policy on how to throttle or deny create, open and stage requests and which of these request types to monitor. For example, it might limit the number of files a user has opened at one time; or it might deny all create requests from a particular host or user. The Site Interfaces will be located in a shared library that is linked into the Gatekeeper Server.

It is important that the Site Interfaces return a status in a timely fashion. Create, open, and stage requests from DFS, NFS, and MPS are timing sensitive, thus the Site Interfaces won't be permitted to delay or deny these requests, however the Site Interfaces may choose to be involved in keeping statistics on these requests by monitoring requests from Authorized Callers.

If a Gatekeeper Server should become heavily loaded, additional Gatekeeper Servers can be configured (maximum of one Gatekeeper Server per storage subsystem). In order to keep the Gatekeepers simple and fast, Gatekeeper Servers do not share state information. Thus if a site wrote a policy to allow each host a maximum of 20 creates, then that host would be allowed to create 20 files on each storage subsystem that has a separate Gatekeeper Server.

The Gatekeeper Server Real Time Monitoring Interface supports clients such as a Real Time Monitoring utility which requests information about particular user files or HPSS Request Ids.

2.6.7 Location Server

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

If any server is down for an extended length of time it is important to mark the server as non-executable within SSM. As long as a server is marked executable the Location Server continues to advertise its location to clients which may try to contact it.

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

When multiple HPSS systems are connected to each other, the Location Servers share server information. If you are connecting multiple HPSS systems together you will need to tell the Location Server how to locate the Location Servers at the remote HPSS systems. You will need to add a record for each site to the Remote Sites metadata file. See Section 2.2.7.1: *Cross Cell Access* on page 45 for more details.

2.6.8 PVL

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

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

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

2.6.9 PVR

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

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

2.6.9.1 STK PVR and STK RAIT PVR

The STK RAIT PVR cannot be supported at this time since STK has not yet made RAIT generally available.

The STK PVR and STK RAIT PVR communicate to the ACSLS server via STK's SSI software.

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

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

HPSS supports the device virtualization feature of the StorageTek StorageNet 6000 series of domain managers. This feature allows for the ability to configure a Redundant Arrays of Independent Tapes (RAIT) as an HPSS device. This capability is enabled by configuring an STK RAIT PVR for each RAIT physical drive pool. The number of physical drives to be used by HPSS is set in the RAIT PVR specific configuration, and used by the PVL when scheduling mounts of RAIT virtual volumes. HPSS supports the following striping / parity combinations for both the 9840 and 9940 virtual drives: 1+0, 2+1, 4+1, 4+2, 6+1, 6+2, 8+1, 8+2, and 8+4.

2.6.9.2 LTO PVR

The LTO PVR manages the IBM 3584 Tape Library and Robot, which mounts, dismounts and manges LTO tape cartridges and IBM 3580 tape drives. The PVR uses the Atape driver interface to issue SCSI commands to the library.

The SCSI control path to the library controller device (**/dev/smc***) is shared with the first drive in the library (typically **/dev/rmt0**). Since the PVR communicates directly to the library via the Atape interface, the PVR must be installed on the same node that is attached to the library.

The LTO PVR operates synchronously, that is, once a request is made to the 3584 library, the request thread does not regain control until the operation has completed or terminated. This means that other requests must wait on an operation to complete before the PVR can issue them to the 3584.

2.6.9.3 3494/3495 PVR

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

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

2.6.9.4 AML PVR

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

2.6.9.5 Operator PVR

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

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

2.6.10 Mover

The Mover configuration will be largely dictated by the hardware configuration of the HPSS system. Each Mover can handle both disk and tape devices and must run on the node to which the storage devices are attached. The Mover is also capable of supporting multiple data transfer mechanisms for sending data to or receiving data from HPSS clients (e.g., TCP/IP and shared memory).

2.6.10.1 Asynchronous I/O

Asynchronous I/O must be enabled manually on AIX and Linux platforms. There should be no asynchronous I/O setup required for Solaris or IRIX platforms.

2.6.10.1.1 AIX

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

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

or **smitty**:

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

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

2.6.10.1.2 Linux

For Linux platforms asynchronous I/O is enabled at the time the kernel is built. To enable asynchronous I/O on the Mover machine, follow the steps below:

1. Update the kernel to level 2.4.18.
2. Download the HPSS kaio-2.4.18 patch from the HPSS support Web site..?????

   Note: Before rebuilding your Linux kernel please read all of section 2.6.10. This might prevent you from doing multiple builds.

   This package contains a kernel source patch for a modified version of SGI's open source implementation of the asynchronous I/O facility (defined by the POSIX standard).
3. Untar the contents of the package. For example:

   ```
   % tar xvf kaio-2.4.18-1.tar
   ```
4. There should be a README file and the source patch. As root, copy the **kaio-2.4.18-1** patch file to the /usr/src directory, and change directory to the base of the Linux source tree. For example:

   ```
   % cp kaio-2.4.18-1 /usr/src
   % cd /usr/src/linux-2.4.18
   ```
5. Apply the source patch using the Lunix "patch" utility. For example:

   ```
   % patch -p1 < ../kaio-2.4.18-1
   ```

6. Now, rebuild the kernel configuration by running the "**make config**" command and answering "yes" when questioned about AIO support. The default value of 4096 should be sufficient for the number of system-wide AIO requests.

   At this time, you should also configure the kernel to support your disk or tape devices. If tape device access is required, be sure to also enable the kernel for SCSI tape support. See the following section for information on device support on the Linux platform.

   Note that the Linux kernel configuration varibles that control the KAIO facility are CONFIG_AIO and CONFIG_AIO_MAX.

7. Follow your procedure for rebuilding your Linux kernel. For example:

   ```
   % make dep
   % make bzImage
   ```

8. Copy the new kernel image to the boot directory, update the lilo configuration, and recycle the system. For example:

   ```
   % cd /boot
   % cp /usr/src/linux-2.4.18/arch/i386/boot/bzImage vmlinux-2.4.18
   % vi /etc/lilo.conf
   % /sbin/lilo
   % shutdown -Fr 0
   ```

9. If you need to rebuild the HPSS Mover, make a link from **/usr/src/linux/include/linux/aio.h** to **/usr/include/linux/aio.h**. For example:

   ```
   % cd /usr/include/linux
   % ln -s /usr/src/linux-2.4.18/include/linux/aio.h
   ```

2.6.10.2 Tape Devices

2.6.10.2.1 AIX

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

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

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

or **smitty**:

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

2.6.10.2.2 Solaris

For Solaris, the method used to enable variable block sizes for a tape device is dependent on the type of driver used. Supported devices include Solaris SCSI Tape Driver and IBM SCSI Tape Driver.

For the IBM SCSI Tape Driver, set the **block_size** parameter in the **/opt/IBMtape/IBMtape.conf** configuration file to **0** and perform a reboot with the reconfiguration option. The Solaris SCSI Tape Driver has a built-in configuration table for all HPSS supported tape drives. This configuration provides variable block size for most HPSS supported drives. In order to override the built-in configuration, device information can be supplied in the **/dev/kernel/st.conf** as global properties that apply to each node.

Consult the tape device driver documentation for instructions on installation and configuration.

2.6.10.2.3 IRIX

Variable block sizes can be enabled for the IRIX native tape device driver by configuring the Mover to use the tape device special file with a “**v**” in the name (e.g. **/dev/rmt/tps5d5nsvc**).

2.6.10.2.4 Linux

HPSS supports tape devices on Linux with the use of the native SCSI tape device driver (**st**). To enable the loading of the Linux native tape device, uncomment the following lines in the "**.config**" file and follow the procedure for rebuilding your Linux kernel.

```
CONFIG_SCSI=y
CONFIG_CHR_DEV_ST=y
```

In Linux, tape device files are dynamically mapped to SCSI IDs/LUNs on your SCSI bus. The mapping allocates devices consecutively for each LUN of each device on each SCSI bus found at the time of the SCSI scan, beginning at the lower LUNs/IDs/buses. The tape device file will be in this format: **`/dev/st[0-31]`**. This will be the device name to use when configuring your HPSS device.

2.6.10.3 Disk Devices

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

For Linux systems, this may involve special consideration.

HPSS supports disk device on Linux with the use of the native SCSI disk device driver (**sd**) and the raw device driver (**raw**).

The Linux SCSI Disk Driver presents disk devices to the user as device files with the following naming convention: **`/dev/sd[a-h][0-8]`**. The first variable is a letter denoting the physical drive, and the second is a number denoting the partition on that physical drive. Often, the partition number, will be left off when the device corresponds to the whole drive. Drives can be partitioned using the Linux **fdisk** utility.

The Linux raw device driver is used to bind a Linux raw character device to a block device. Any block device may be used.

See the Linux manual page for more information on the SCSI Disk Driver, the Raw Device Driver and the **fdisk** utility.

To enable the loading of the Linux native SCSI disk device, uncomment the following lines in the **.config** file and follow the procedure for rebuilding your Linux kernel.

```
CONFIG_SCSI=y
CONFIG_BLK_DEV_SD=y
```

Also, depending on the type of SCSI host bus adapter (HBA) that will be used, you will need to enable one or more of the lower level SCSI drivers. For example, if you are using one of the Adaptec HBAs with a 7000 series chip set, uncomment the following lines in the "**.config**" file and follow the procedure for rebuilding your Linux kernel.

```
CONFIG_SCSI_AIC7XXX=y
CONFIG_AIC7XXX_CMDS_PER_DEVICE=253
CONFIG_AIC7XXX_RESET_DELAY_MS=15000
```

2.6.10.4 Performance

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

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

- Each Mover executable is built to handle a single particular device interface (e.g., IBM SCSI-attached 3490E/3590 drives, IBM BMUX-attached 3480/3490/3490E drives). If multiple types of device specific interfaces are to be supported, multiple Movers must be configured.
- Each Mover currently limits the number of concurrently outstanding connections. If a large number of concurrent requests are anticipated on the drives planned for a single Mover, the device work load should be split across multiple Movers (this is primarily an issue for Movers that will support disk devices).
- The planned device allocation should be examined to verify that the device allocated to a single node will not overload that node's resource to the point that the full transfer rates of the device cannot be achieved (based on the anticipated storage system usage). To off-load a single node, some number of the devices can be allocated to other nodes, and corresponding Movers defined on those same nodes.
- In general, the connectivity between the nodes on which the Movers will run and the nodes on which the clients will run should have an impact on the planned Mover configuration. For TCP/IP data transfers, the only functional requirement is that routes exist between the clients and Movers; however, the existing routes and network types will be important to the performance of client I/O operations.

- Mover to Mover data transfers (accomplished for migration, staging, and repack operations) also will impact the planned Mover configuration. For devices that support storage classes for which there will be internal HPSS data transfers, the Movers controlling those devices should be configured such that there is an efficient data path among them. If Movers involved in a data transfer are configured on the same node, the transfer will occur via a shared memory segment (involving no explicit data movement from one Mover to the other).

2.6.11 Logging Service

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

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

The Delog process is executed as an on-demand process by the Storage System Manger, or can be executed as a command line utility. If Delog is to be initiated from the Storage System Manager, the Delog process will execute on the same node as the Storage System Manager. If Delog is initiated from the command line utility, the central log file must be accessible from the node on which the command is being executed (e.g., NFS mounted). Refer to Section 1.16.2: *Viewing the HPSS Log Messages and Notifications* (page 37) in the *HPSS Management Guide* for detailed information on Delog.

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

2.6.12 Metadata Monitor

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

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

2.6.13 NFS Daemons

By default files and directories created with NFS will have their HPSS account index set to the user's default account index. Account validation is recommended with NFS if consistency of accounting information is desired. If account validation is not enabled, it may not be possible to keep accounting information consistent. For example, if users have no DCE account, their UID will be used as the account. This may conflict with users set up to use site style accounting in the same cell.

Even if no client NFS access is required, the NFS interface may provide a useful mechanism for HPSS name space object administration.

The HPSS NFS Daemon cannot be run on a processor that also runs the native operating system's NFS daemon. Therefore it will not be possible to export both HPSS and native Unix file systems from the same processor. In addition the NFS daemon will require memory and local disk storage to maintain caches for HPSS file data and attributes. NFS memory and disk requirements are discussed in Section 2.10.3.2 and 2.10.3.2.8. Since the NFS Daemon communicates with the HPSS Name Server frequently, running the NFS Daemon on the same platform as the HPSS Name Server is recommended.

NFS access to an exported subtree or fileset is controlled through the use of an exports file, which is a Unix text file located on the machine where the HPSS NFS daemon is running. Entries in the exports file tell which subtrees and filesets are exported and what client systems can access them. Additional options are available to specify the type of access allowed and additional security related features. Export entry options are described in more detail in Sections 2.8.4 and 7.4: *NFS Daemon Configuration* (page 431).

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

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

By default, files created with NFS will have the HPSS accounting index set to -1, which means that HPSS will choose the account code for the user. Standard HPSS accounting mechanisms are supported only through the export file’s **UIDMAP** option, described in Section 7.4.1: *The HPSS Exports File* on page 432. If the **UIDMAP** option is specified, the user’s default account index will be used for file creation. The **nfsmap** utility provides a capability for specifying an account index other than the user’s default.

2.6.14 Startup Daemon

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

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

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

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

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

2.6.15 Storage System Management

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

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

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

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

2.6.15.1 hpssadm Considerations

The Command Line SSM utility, **hpssadm**, may be executed on any AIX, Solaris, Linux, or Windows system. It has been tested on AIX and Solaris.

Great care must be taken to secure the machine on which the Data Server executes. Only trusted, administrative users should have accounts on this machine, and its file systems should not be shared across the network.

2.6.16 HDM Considerations

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

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

When the HDM fetches an event from DFS or XFS, it is put on a queue and assigned to an appropriate event handler when one becomes free. The total number of entries allowed in the queue is determined by a configuration parameter. If this value is not large enough to handle a reasonable number of requests, some of the event handlers may be starved. For example, if the queue fills up with data events, the name space handlers will be starved. Section 7.6.3.3.1: *config.dat File* on page 449 discusses the criteria for selecting the size of an event queue.

HDM logs outstanding name space events. If the HDM is interrupted, the log is replayed when the HDM restarts to ensure that the events have been processed to completion and the DFS/XFS and HPSS name spaces are synchronized. The size of the log is determined by a configuration parameter, as discussed in Section 7.6.3.3.1: *config.dat File* on page 449.

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

Since the HDM may be running on a machine where it cannot write error messages to the HPSS message log, it uses its own log. This HDM log consists of a configurable number of files (usually 2) that are written in round-robin fashion. The sizes of these files are determined by a configuration parameter.

HDM logging policy allows the system administrator to determine the type of messages written to the log file: alarm, event, debug, and/or trace messages. Typically, only alarms should be enabled, although event messages can be useful, and do not add significant overhead. If a problem occurs, activating debug and trace messages may provide additional information to help identify the problem. However, these messages add overhead, and the system will perform best if messages are kept to a minimum. The type of messages logged is controlled by a parameter in the configuration file and can be dynamically changed using the **hdm_admin** utility.

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

2.6.17 Non-DCE Client Gateway

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

An HPSS installation can have multiple NDCG servers. A client can utilize a particular NDCG server by setting its **HPSS_NDCG_SERVERS** environment variable with the hostname and port of the target NDCG server.

The NDCG can be configured to support client authentication. A single NDCG can be configured to support Kerberos and/or DCE authentication. The client requests one of the supported authentication methods during the initial connection. Client authentication can also be completely disabled on a NDCG server basis. In this case, the NDCG server believes the identity of the client sent during the initial connection. When using DCE authentication, the DCE identity and password are passed in an encrypted format from client to server during the initial connection. The NDCG can be configured to support either DES or simple hashing function for encryption of the DCE identity and password that is passed to the NDCG.

See Section 2.3.4.2: *HPSS Non-DCE Mover/Client Machine* on page 51 for more information on prerequisites for a Non-DCE configuration.

2.7 Storage Subsystem Considerations

Storage subsystems have been introduced into HPSS for the purpose of increasing the scalability of the system - particularly with respect to the name and bitfile servers. In releases prior to 4.2, an HPSS system could only contain a single name and bitfile server. With the addition of storage subsystems, an HPSS system must now contain one or more storage subsystems, and each storage subsystem contains its own name and bitfile servers. If multiple name and bitfile servers are desired, this is now possible by configuring an HPSS system with multiple storage subsystems.

A basic HPSS system contains a single storage subsystem. Additional storage subsystems allow the use of multiple name and bitfile servers, but also introduce additional complexity in the form of extra subsystems and servers being defined. Storage subsystems can also be used to increase scalability with respect to SFS, but the price of this is that each storage subsystem requires its own copies of several metadata files to support the servers in that subsystem. Finally, storage subsystems provide a useful way to partition an HPSS system, though they also require that storage resources be fragmented in order to support the multiple subsystems.

2.8 Storage Policy Considerations

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

2.8.1 Migration Policy

The migration policy provides the capability for HPSS to copy (migrate) data from one level in a hierarchy to one or more lower levels. The migration policy defines the amount of data and the conditions under which it is migrated, but the number of copies and the location of those copies is determined by the storage hierarchy definition. The site administrator will need to monitor the usage of the storage classes being migrated and adjust both the migration and purge policies to obtain the desired results.

2.8.1.1 Migration Policy for Disk

Disk migration in HPSS copies (migrates) files from a disk storage class to one or more lower levels in the storage hierarchy. Removing or purging of the files from disk storage class is controlled by the purge policy. The migration and purge policies work in conjunction to maintain sufficient storage space in the disk storage class.

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

Disk migration is controlled by several parameters:

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

- The **Free Space Target** controls the number of bytes to be copied by a migration run. The value of this parameter is important in association with the purge policy. The amount of data that is copied is potentially purgeable when the next purge on this storage class is run. This value must be set at a sufficient level so that enough purgeable space is created for the purge to meet the free space demands for users of this storage class. If this value is set to other than 100%, the time at which copies are made of disk files will be unpredictable.

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

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

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

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

2.8.1.2 Migration Policy for Tape

There are two different tape migration algorithms: tape file migration and tape volume migration. The algorithm which MPS applies to a given tape storage class is selected in the migration policy for that storage class.

The purpose of tape file migration is to make a second copy of files written on tape. This algorithm is similar to disk migration, but only a single additional copy is possible. It is also possible to configure tape file migration such that files are moved downwards in the hierarchy without keeping a second copy.

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

Tape volume migration attempts to empty tapes by moving data off of the tapes to other volumes. When a tape becomes empty, it is a candidate for reuse. A special utility called **reclaim** resets the state of the empty tape volumes so that they can be reused. The **reclaim** utility can be run from SSM, but it should generally be set up to run on a periodic basis via the **cron** facility. For more information on **reclaim**, see Section 3.8: *Reclaiming HPSS Tape Virtual Volumes* on page 76 of the *HPSS Management Guide* and Section 12.2.47: *reclaim — HPSS Volume Reclaim Utility* on page 438 of the *HPSS Management Guide*.

The **repack** utility can also be used to create empty tapes in a storage class. The administrator should determine whether a tape should be repacked based on the number of holes (due to file overwrite or deletion) on the tape. If a tape storage class is at the bottom of a hierarchy, **repack** and **reclaim** must be run periodically to reclaim wasted space. For more information on **repack**, see Section 3.7: *Repacking HPSS Volumes* on page 74 of the *HPSS Management Guide* and Section 12.2.51: *repack — HPSS Volume Repack Utility* on page 452 of the *HPSS Management Guide*.

The migration policy parameters which apply to the different tape migration algorithms are described below. Parameters which only apply to disk migration are not described.

- The **Last Read Interval** parameter is used by both tape volume migration algorithms as well as tape file migration with purge to determine if a file is actively being read or is inactive. A file which has been read more recently than the number of minutes specified in this field is considered active. If a file is read active, tape volume migration moves it laterally to another volume in the same storage class. Tape file migration with purge avoids moving read active files at all. If a file is read inactive, all three algorithms migrate it down the hierarchy. The purpose of this field is to avoid removing the higher level copy of a file which is likely to be staged again.

- The **Last Update Interval** is used by all of the tape migration algorithms to determine if a file is actively being written. A file which has been written more recently than the number of minutes specified in this field is considered active. If a file is write active, tape volume migration moves it laterally instead of migrating it downwards. Tape file migration avoids any operations on write active files. The purpose of this field is to avoid performing a migration on a file which is likely to be rewritten and thus need to be migrated yet again.
- The **Free Space Target** parameter controls the total number of free virtual volumes a site intends to maintain. It is suggested that this field always be set to 100%.
- The **Request Count** field is used by the tape file migration algorithms only to determine the number of concurrent migration threads used by each migration run. Each migration thread operates on a separate source volume, so, this parameter determines the number of volumes which are migrated at once.
- The **Runtime Interval** is used by all of the tape migration algorithms to control how often migration runs for a given storage class. In addition, the administrator can force a migration run to start via SSM.
- The **Migrate Volumes** flag selects the tape volume migration algorithm.
- The **Migrate Volumes and Whole Files** flag modifies the tape volume migration algorithm to avoid having the storage segments of a single file scattered over different tapes. When this field is set, if a file on a virtual volume is selected to be migrated to the next level, any parts of this file that are on different virtual volumes are also migrated even if they would ordinarily not meet the criteria for being migrated. This tends to pack all the storage segments for a given file on the same virtual volume.
- The **Migrate Files** flag selects the tape file migration algorithm.
- The **Migrate Files and Purge** modifies the tape file migration algorithm such that the higher level copy is removed once the lower level copy is created. Only one copy of each file is maintained.

2.8.2 Purge Policy

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

The disk purge is controlled by several parameters:

- The **Start purge when space used reaches <nnn> percent** parameter allows sites control over the amount of free space that is maintained in a disk storage class. A purge run will be started for this storage class when the total space used in this class exceeds this value.

- The **Stop purge when space used falls to <nnn> percent** parameter allows sites control over the amount of free space that is maintained in a disk storage class. The purge run will attempt to create this amount of free space. Once this target is reached, the purge run will end.

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

- The **Purge by** list box allows sites to choose the criteria used in selecting files for purge. By default, files are selected for purge based on the creation time of their purge record. Alternately, the selection of files for purging may be based on the time the file was created or the time the file was last accessed. Files may be purged in an unpredictable order if this parameter is changed while there are existing purge records already in metadata until those existing files are processed.

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

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

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

2.8.3 Accounting Policy and Validation

The purpose of the Accounting Policy is to describe how the site will charge for storage usage as well as the level of user authorization (validation) performed when maintaining accounting information.

A site must decide which style of accounting to use before creating any HPSS files or directories. There are two styles of accounting: UNIX-style accounting and Site-style accounting. In addition a site may decide to customize the style of accounting used by writing an accounting site policy module for the Gatekeeper.

The metadata for each file and directory in the HPSS system contains a number, known as an account index, which determines how the storage will be charged. Each user has at least one account index, known as their default account index, which is stamped on new files or directories as they are created.

In UNIX-style accounting, each user has one and only one account index, their UID. This, combined with their Cell Id, uniquely identifies how the information may be charged.

In Site-style accounting, each user may have more than one account index, and may switch between them at runtime.

A site must also decide if it wishes to validate account index usage. Prior to HPSS 4.2, no validation was performed. For Site-style accounting, this meant that any user could use any account index they wished without authorization checking. UNIX-style accounting performs de facto authorization checking since only a single account can be used and it must be the user's UID.

If Account Validation is enabled, additional authorization checks are performed when files or directories are created, their ownership changed, their account index changed, or when a user attempts to use an account index other than their default. If the authorization check fails, the operation fails as well with a permission error.

Using Account Validation is highly recommended if a site will be accessing HPSS systems at remote sites, now or in the future, in order to keep account indexes consistent. Event if this is not the case, if a site is using Site-style accounting, Account Validation is recommended if there is a desire by the site to keep consistent accounting information.

For UNIX-style accounting, at least one Gatekeeper server must be configured and maintained. No other direct support is needed.

For Site-style accounting, an Account Validation metadata file must also be created, populated and maintained with the valid user account indexes. See Section 12.2.23: *hpss_avaledit — Account Validation Editor* on page 366 of the *HPSS Management Guide* for details on using the Account Validation Editor.

If the **Require Default Account** field is enabled with Site-style accounting and Account Validation, a user will be required to have a valid default account index before they are allowed to perform almost any client API action. If this is disabled (which is the default behavior) the user will only be required to have a valid account set when they perform an operation which requires an account to be validated, such as a create, an account change operation or an ownership change operation.

When using Site-style accounting with Account Validation if the **Account Inheritance** field is enabled, newly created files and directories will automatically inherit their account index from their parent directory. The account indexes may then be changed explicitly by users. This is useful when individual users have not had default accounts set up for them or if entire trees need to be charged to the same account. When **Account Inheritance** is disabled (which is the default) newly created files and directories will obtain their account from the user's current session account, which initially starts off as the user's default account index and may be changed by the user during the session.

A site may decide to implement their own style of accounting customized to their site's need. One example would be a form of Group (GID) accounting. In most cases the site should enable Account Validation with Site-style accounting and implement their own site policy module to be linked with the Gatekeeper. See Section 2.6.6: *Gatekeeper* on page 68 as well as the appropriate sections of the *HPSS Programmers Reference Vol. 2* for more information.

Account Validation is disabled (bypassed) by default and is the equivalent to behavior in releases of HPSS prior to 4.2. If it is disabled, the style of accounting is determined for each individual user by looking up their DCE account information in the DCE registry. The following instructions describe how to set up users in this case.

If a user has their default account index encoded in a string of the form **AA=<default-acct-idx>** in their DCE account's **gecos** field or in their DCE principal's **HPSS.gecos** extended registry attribute (ERA), then Site-style accounting will be used for them. Otherwise it will be assumed that they are using UNIX-style accounting.

To keep the accounting information consistent, it is important for this reason to set up all users in the DCE registry with the same style of accounting (i.e. they should all have the **AA=** string in their DCE information or none should have this string.)

See Appendix D: *Accounting Examples* (page 491) for more information.

2.8.4 Security Policy

HPSS server authentication and authorization make extensive use of Distributed Computing Environment (DCE) authentication and authorization mechanisms. Each HPSS server has configuration information that determines the type and level of services available to that server. HPSS software uses these services to determine the caller identity and credentials. Server security configuration is discussed more in Section 6.5: *Basic Server Configuration* (page 262).

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

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

2.8.4.1 Client API

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

2.8.4.2 Non-DCE Client API

The Non-DCE Client API implements security in 3 modes:

- **DCE Authentication** (default) The client to enter a DCE principal and password which are then encrypted and sent to the Non-DCE Gateway. The gateway will then try to use this combination to acquire DCE credentials. The encryption is performed using either the DES algorithm or a simple hashing function (for sites where DES restrictions apply).

- **Kerberos Authentication** The client tries to authenticate to the gateway using a Kerberos ticket.

- **No Authentication** Disables the security features so the client is always trusted and authenticated.

2.8.4.3 FTP/PFTP

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

Alternatively, authentication may be performed using the DCE Registry or using password-less mechanisms such as MIT Kerberos.

2.8.4.4 DFS

DFS uses DCE authentication and authorization.

2.8.4.5 NFS

Though the HPSS NFS client interface does not directly support an end user login authorization mechanism, standard NFS export security features are supported to allow specification of read-only, read-mostly, read-write, and root access to HPSS subtrees for identified client hosts. HPSS NFS does not support Sun MicroSystems' Network Information Services to validate client hosts. HPSS NFS does provide an option to validate the network address of hosts attempting to mount HPSS directories. The default configuration disables this check. To enable client address validation, export the variable HPSS_MOUNTD_IPCHECK in the HPSS environments file (**hpss_env**). An option to specify mediation of user access to HPSS files by a credentials mapping is also provided. Export entry options are described further in Section 7.4: *NFS Daemon Configuration* (page 431).

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

2.8.4.6 Bitfile

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

2.8.4.7 *Name Space*

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

2.8.4.8 *Security Audit*

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

2.8.5 *Logging Policy*

The logging policy provides the capability to control which message types are written to the HPSS log files. In addition, the logging policy is used to control which alarms, events, and status messages are sent to the Storage System Manager to be displayed. Logging policy is set on a per server basis. Refer to Section 6.6.4: *Configure the Logging Policies* (page 293) for a description of the supported message types.

If a logging policy is not explicitly configured for a server, the default log policy will be applied. The default log policy settings are defined from the HPSS Log Policies window. If no Default Log Policy entry has been defined, all record types except for Trace are logged. All Alarm, Event, and Status messages generated by the server will also be sent to the Storage System Manager.

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

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

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

2.8.6 *Location Policy*

The location policy provides the ability to control how often Location Servers at an HPSS site will contact other servers. It determines how often remote Location Servers are contacted to exchange server location information. An administrator tunes this by balancing the local site's desire for accuracy of server map information against the desire to avoid extra network and server load.

2.8.7 Gatekeeping

Every Gatekeeper Server has the ability to supply the Gatekeeping Service. The Gatekeeping Service provides a mechanism for HPSS to communicate information through a well-defined interface to a policy software module to be completely written by the site. The site policy code is placed in a well-defined site shared library for the gatekeeping policy (**/opt/hpss/lib/ libgksite.[a|so]**) which is linked to the Gatekeeper Server. The default gatekeeping policy shared library does NO gatekeeping. Sites will need to enhance this library to implement local policy rules if they wish to monitor and/or load balance requests.

The gatekeeping site policy code will need to determine which types of requests it wants to monitor (authorized caller, create, open, and stage). Upon initialization, each BFS will look for a Gatekeeper Server configured into the same storage subsystem. If one is found, then the BFS will query the Gatekeeper Server asking for the monitor types by calling a particular Gatekeeping Service API (**gk_GetMonitorTypes**) which will in turn call the appropriate site implemented Site Interface (**gk_site_GetMonitorTypes**) which will determine which types of requests it wishes to monitor. This query by the BFS will occur each time the BFS (re)connects to the Gatekeeper Server. The BFS will need to (re)connect to the Gatekeeper whenever the BFS or Gatekeeper Server is restarted. Thus if a site wants to change the types of requests it is monitoring, then it will need to restart the Gatekeeper Server and BFS.

For each type of request being monitored, the BFS will call the appropriate Gatekeeping Service API (**gk_Create**, **gk_Open**, **gk_Stage**) passing along information pertaining to the request. This information includes:

Table 2-2 Gatekeeping Call Parameters

Name	Description	create	open	stage
AuthorizedCaller	Whether or not the request is from an authorized caller. These requests cannot be delayed or denied by the site policy.	Y	Y	Y
BitFileID	The unique BFS identifier for the file.	N/A	Y	Y
ClientConnectId	The end client's connection uuid.	Y	Y	Y
DCECellId	The HPSS DCE cell identifier for the user.	Y	Y	Y
GroupId	The user's group identifier	Y	Y	Y
HostAddr	Socket information for originating host.	Y	Y	Y
OpenInfo	Open file status flag (Oflag).	N/A	Y	N/A
StageInfo	Information specific to stage (flags, length, offset, and storage level).	N/A	N/A	Y
UserId	The user's identifier.	Y	Y	Y

Each Gatekeeping Service API will then call the appropriate Site Interface passing along the information pertaining to the request. If the request had AuthorizedCaller set to TRUE, then the

Site "Stat" Interface will be called (**gk_site_CreateStats**, **gk_site_OpenStats**, **gk_site_StageStats**) and the Site Interface will not be permitted to return any errors on these requests. Otherwise, if **AuthorizedCaller** is set to **FALSE**, then the normal Site Interface will be called (**gk_site_Create**, **gk_site_Open**, **gk_site_Stage**) and the Site Interface will be allowed to return no error or return an error to either retry the request later or deny the request. When the request is being completed or aborted the appropriate Site Interface will be called (**gk_site_Close**, **gk_site_CreateComplete**, **gk_site_StageComplete**). Examples of when a request gets aborted are when the BFS goes **DOWN** or when the user application is aborted.

NOTES:

1. All open requests to the BFS will call the Gatekeeping Service open API (**gk_Open**). This includes opens that end up invoking a stage.
2. Any stage call that is invoked on behalf of open will NOT call the Gatekeeping Service stage API (**gk_Stage**). (e.g. The ftp **site stage <filename>** command will use the Gatekeeping Service open API, **gk_Open**, rather than the Gatekeeping Service stage API, **gk_Stage**.)
3. Direct calls to stage (**hpss_Stage**, **hpss_StageCallBack**) will call the Gatekeeping Service stage API (**gk_Stage**).
4. If the site is monitoring Authorized Caller requests then the site policy interface won't be allowed to deny or delay these requests, however it will still be allowed to monitor these requests. For example, if a site is monitoring Authorized Caller and Open requests, then the site **gk_site_Open** interface will be called for open requests from users and the **gk_site_OpenStats** interface will be called for open requests due an authorized caller request (e.g. migration by the MPS). The site policy can NOT return an error for the open due to migration, however it can keep track of the count of opens by authorized callers to possibly be used in determining policy for open requests by regular users. Authorized Caller requests are determined by the BFS and are requests for special services for MPS, DFS, and NFS. These services rely on timely responses, thus gatekeeping is not allowed to deny or delay these special types of requests.
5. The Client API uses the environment variable HPSS_GKTOTAL_DELAY to place a maximum limit on the number of seconds a call will delay because of HPSS_ERETRY status codes returned from the Gatekeeper. See Section 7.1: *Client API Configuration* on page 413 for more information.

Refer to *HPSS Programmer's Reference, Volume 1* for further specifications and guidelines on implementing the Site Interfaces.

2.9 Storage Characteristics Considerations

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

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

A Storage Class is used by HPSS to define the basic characteristics of storage media. These characteristics include the media type (the make and model), the media block size (the length of each basic block of data on the media), the transfer rate, and the size of media volumes. These are the physical characteristics of the media. Individual media volumes described in a Storage Class are called Physical Volumes (PVs) in HPSS.

Storage Classes also define the way in which Physical Volumes are grouped to form Virtual Volumes (VVs). Each VV contains one or more PVs. The VV characteristics described by a Storage Class include the VV Block Size and VV Stripe Width.

A number of additional parameters are defined in Storage Classes. These include migration and purge policies, minimum and maximum storage segment sizes, and warning thresholds.

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

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

A file family is an attribute of an HPSS file that is used to group a set of files on a common set of tape virtual volumes. HPSS supports grouping of files only on tape volumes. In addition, families can only be specified by associating a family with a fileset, and creating the file in the fileset. When a file is migrated from disk to tape, it is migrated to a tape virtual volume assigned to the family associated with the file. If no family is associated with the file, the file is migrated to the next available tape not associated with a family (actually to a tape associated with family zero). If the file is associated with a family and no tape VV is available for writing in the family, a blank tape is reassigned from family zero to the file's family. The family affiliation is preserved when tapes are repacked.

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

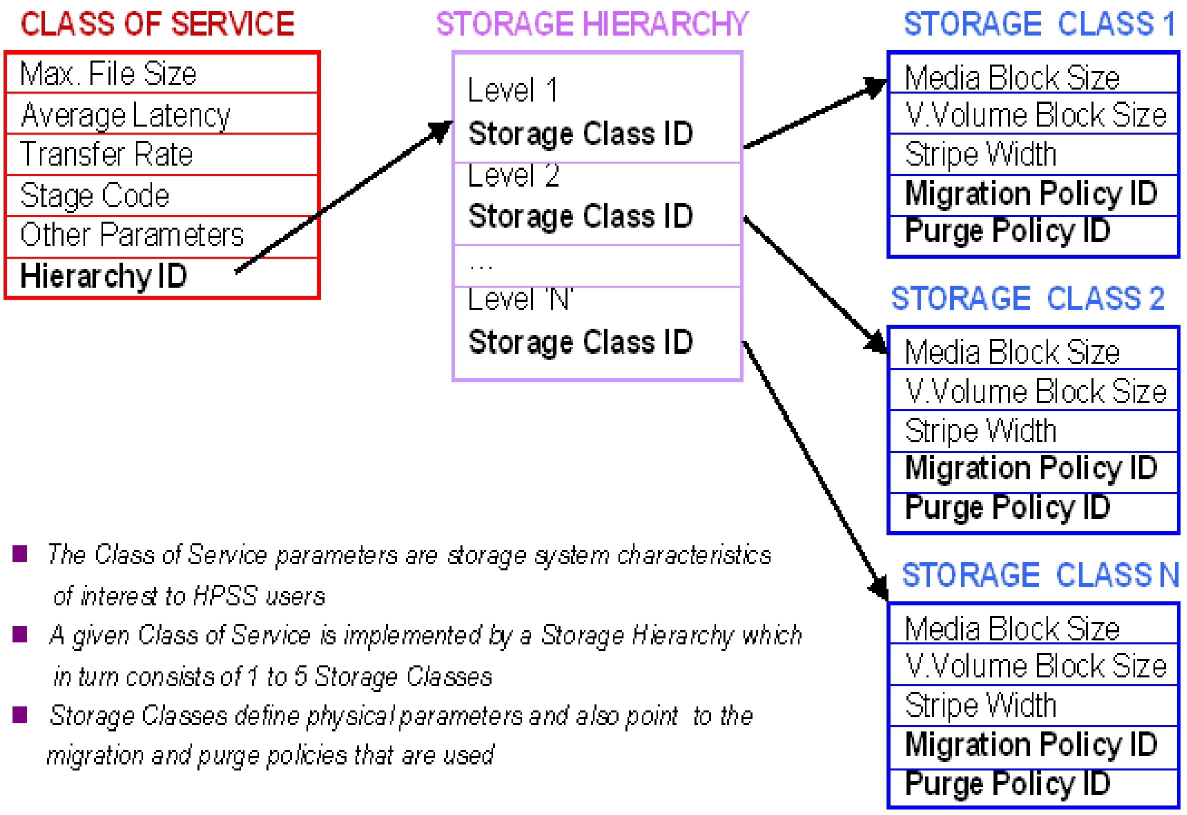


Figure 2-2 Relationship of Class of Service, Storage Hierarchy, and Storage Class

2.9.1 *Storage Class*

Each virtual volume and its associated physical volumes belong to some storage class in HPSS. The SSM provides the capability to define storage classes and to add and delete virtual volumes to and from the defined storage classes. A storage class is identified by a storage class ID and its associated attributes. For detailed descriptions of each attribute associated with a storage class, see Section 6.7.1: *Configure the Storage Classes* (page 305). Once a storage class has been defined, great care must be taken if the definition is to be changed or deleted. This especially applies to media and VV block size fields.

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

2.9.1.1 *Media Block Size Selection*

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

Explanation: For example, if a site has ESCON attached tape drives on an RS6000, the driver can handle somewhat less than 64 KB physical blocks on the tape. A good selection here would be 32 KB. See Section 2.9.1.12 for recommended values for tape media supported by HPSS.

2.9.1.2 Virtual Volume Block Size Selection (disk)

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

Explanation: This is needed for correct operation of striped I/O.

2.9.1.3 Virtual Volume Block Size Selection (tape)

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

Explanation: This is needed for correct operation of striped I/O.

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

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

Guideline 3: Disk and tape VV block sizes should be chosen so when data is migrated from disk to tape, or tape to disk, the VV block size doesn't change.

Explanation: The system is designed to maximize throughput of data when it is migrated from disk to tape or tape to disk. For best results, the sizes of the VV blocks on disk and tape in a migration path should be the same. If they are different, the data will still be migrated, but the Movers will be forced to reorganize the data into different size VV blocks which can significantly impact performance.

2.9.1.4 Stripe Width Selection

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

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

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

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

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

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

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

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

```
<File Size in MB> / 10 + 20
```

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

```
<File Size in MB> / 20 + 2 * 20
```

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

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

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

2.9.1.5 Blocks Between Tape Marks Selection

Blocks between tape marks is the number of physical media blocks written before a tape mark is generated. The tape marks are generated for two reasons: (1) To force tape controller buffers to flush so that the Mover can better determine what was actually written to tape, and (2) To quicken positioning for partial file accesses. Care must be taken, however in setting this value too low, as it can have a negative impact on performance. For recommended values for various media types, see Section 2.9.1.12.

2.9.1.6 Storage Segment Size Selection (disk only)

The Bitfile Server maps files into a series of storage segments. The size of the storage segments is controlled by the **Minimum Storage Segment Size** parameter, the **Maximum Storage Segment Size** parameter, and the **Average Number of Segments** parameter. The smallest amount of disk storage that can be allocated to a file is determined by the **Minimum Storage Segment Size** parameter. This parameter should be chosen with disk space utilization in mind. For example, if writing a 4 KB file into a storage class where the storage segment size is 1,024 KB, 1,020 KB of the space will be wasted. At the other extreme, each file can have only 10,000 disk storage segments, so it wouldn't even be possible to completely write a terabyte file to a disk storage class with a maximum storage segment size below 128 megabytes. When file size information is available the Bitfile Server will attempt to choose an optimal storage segment size between **Minimum Storage Segment Size** and **Maximum Storage Segment Size** with the goal of creating **Average Number of Segments** for the bitfile. The storage segment size will also be chosen as a power of 2 multiple of the **Minimum Storage Segment Size** parameter.

The smallest value that can be selected for the **Minimum Storage Segment Size** is the Cluster Length. Cluster Length is the size, in bytes, of the disk allocation unit for a given VV. Cluster Length is calculated when the VV is created using the **PV Size**, the **VV Block Size** and the **Stripe Width**. Once the Cluster Length for a VV has been established, it cannot be changed. If the characteristics of the Storage Class change, the Cluster Lengths of existing VVs remain the same.

The Cluster Length of a VV is always a multiple of the stripe length of the VV. If the VV has a stripe width of one, the stripe length is the same as the VV block size, and the Cluster Length will be an integer multiple of the VV block size. If the VV has a stripe width greater than one, the stripe length is the product of the VV block size and the stripe width, and the Cluster Length will be a multiple of the Stripe Length.

The number of whole stripes per Cluster is selected such that the number of Clusters in the VV is less than or equal to 16384. This means that any disk VV can contain no more than 16384 Clusters, which means it can contain no more than 16384 Disk Storage Segments. Since a user file on disk must be composed of at least one Storage Segment, there can be no more than 16384 user files on any given disk VV.

As the size of a disk virtual volume increases, its Cluster Length increases accordingly. This means that the Minimum Storage Segment Size also increases as the disk VV increases in size. These relationships are calculated and enforced by SSM and the Disk Storage Server automatically and cannot be overridden.

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

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

NFS is more of a challenge in this area. NFS puts all the files it creates into a particular class of service, independent of the file size. If a fileset has a class of service associated with it, files will be put into that class of service. Otherwise, the daemon's class of service will be used. Sites may want to set up different filesets to deal with large and small files, and assign a class of service accordingly. It is important to use the Storage Segment Size, Maximum Storage Segment Size, and Average Number of Segments parameters to allow for the range of file sizes that clients typically store in the fileset. Remember that NFS V2 files cannot be larger than 2GB.

2.9.1.7 Maximum Storage Size Selection (disk only)

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

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

2.9.1.8 Maximum VVs to Write (tape only)

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

2.9.1.9 Average Number of Storage Segments (disk only)

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

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

2.9.1.10 PV Estimated Size / PV Size Selection

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

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

Rule 1: For disk, the PV Size value must be the exact number of bytes available on the PV. This value must be a multiple of the media block size and the VV block size. The SSM will enforce these rules when the screen fields are filled in.

Rule 2: For disk, the PV Size value must be less than or equal to the Bytes on Device value described in section 6.9: *Configure MVR Devices and PVL Drives* (page 401).

2.9.1.11 Optimum Access Size Selection

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

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

2.9.1.12 Some Recommended Parameter Values for Supported Storage Media

Section Table 2-3: *Suggested Block Sizes for Disk* on page 98 and Section Table 2-4: *Suggested Block Sizes for Tape* on page 99 contain suggested values for storage resource attributes based on the type of storage media. The specified values are not the *only* acceptable values, but represent reasonable settings for the various media types. See Section 2.8.6 for more information about setting the storage characteristics.

2.9.1.12.1 Disk Media Parameters

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

Table 2-3 Suggested Block Sizes for Disk

Disk Type	Media Block Size	Minimum Access Size	Minimum Virtual Volume Block Size	Notes
SCSI Attached	4 KB	0	1 MB	1
SSA Attached	4 KB	0	1 MB	1

Table 2-3 Suggested Block Sizes for Disk

Disk Type	Media Block Size	Minimum Access Size	Minimum Virtual Volume Block Size	Notes
Fibre Channel Attached	4 KB	0	1 MB	1

In Table 2-3:

- Disk Type is the specific type of media to which the values in the row apply.
- Media Block Size is the block size to use in the storage class definition. For disk, this value should also be used when configuring the Mover devices that correspond to this media type. Note that this value will not limit the amount of data that can be read from or written to a disk in one operation—it is used primarily to perform block boundary checking to ensure that all device input/output requests are block aligned. This value should correspond to the physical block size of the disk device.
- Minimum Access Size is the smallest size data access requests that should regularly be satisfied by the corresponding media type. Any accesses for less than the listed amount of data will suffer severe performance degradation. A value of zero indicates that the media is in general suitable for supporting the small end of the system's data access pattern.
- Minimum Virtual Volume Block Size is the smallest block size value that should be used for the corresponding media type when physical volumes are combined to form a striped virtual volume. A value smaller than that specified may result in severely degraded performance when compared to the anticipated performance of the striped virtual volume.
- **Note**: When SCSI, SSA or Fibre Channel attached disks are combined to form striped virtual volumes, the minimum access size should become—at a minimum—the stripe width of the virtual volume multiplied by the virtual volume block size. If not, data access will only use a subset of the striped disks and therefore not take full advantage of the performance potential of the virtual volume.

2.9.1.12.2 Tape Media Parameters

Table 2-4 contains attributes settings for the supported tape storage media types.

Table 2-4 Suggested Block Sizes for Tape

Tape Type	Media Block Size	Blocks Between Tape Marks	Estimated Physical Volume Size
Ampex DST-312	1 MB	1024	50, 150, 330 GB
Ampex DST-314	1 MB	1024	100, 300, 660 GB
IBM 3480	32 KB	512	200 MB
IBM 3490	32 KB	1024	400 MB

Table 2-4 Suggested Block Sizes for Tape

Tape Type	Media Block Size	Blocks Between Tape Marks	Estimated Physical Volume Size
IBM 3490E	32 KB	512	800 MB
IBM 3580	256 KB	1024	100 GB
IBM 3590	256 KB	512	10, 20GB
IBM 3590E	256 KB	512	20, 40GB
IBM 3590H	256 KB	512	60, 120 GB
Sony GY-8240	256 KB	1024	60, 200 GB
StorageTek 9840	256 KB	1024	20 GB
StorageTek 9840 RAIT 1+0	128 KB	512	20 GB
StorageTek 9840 RAIT 1+1	128 KB	512	20 GB
StorageTek 9840 RAIT 2+1	256 KB	512	40 GB
StorageTek 9840 RAIT 4+1	512 KB	1024	80 GB
StorageTek 9840 RAIT 4+2	512 KB	1024	80 GB
StorageTek 9840 RAIT 6+1	678 KB	2048	120 GB
StorageTek 9840 RAIT 6+2	678 KB	2048	120 GB
StorageTek 9840 RAIT 8+1	1 MB	2048	160 GB
StorageTek 9840 RAIT 8+2	1 MB	2048	160 GB
StorageTek 9840 RAIT 8+4	1 MB	2048	160 GB
StorageTek 9940	256 KB	1024	60 GB
StorageTek 9940B	256 KB	1024	200 GB
StorageTek 9940 RAIT 1+0	128 KB	512	60 GB
StorageTek 9940 RAIT 1+1	128 KB	512	60 GB
StorageTek 9940 RAIT 2+1	256 KB	512	120 GB
StorageTek 9940 RAIT 4+1	512 KB	1024	240 GB
StorageTek 9940 RAIT 4+2	512 KB	1024	240 GB
StorageTek 9940 RAIT 6+1	678 KB	2048	360 GB
StorageTek 9940 RAIT 6+2	678 KB	2048	360 GB
StorageTek 9940 RAIT 8+1	1 MB	2048	480 GB
StorageTek 9940 RAIT 8+2	1 MB	2048	480 GB
StorageTek 9940 RAIT 8+4	1 MB	2048	480 GB

Table 2-4 Suggested Block Sizes for Tape

Tape Type	Media Block Size	Blocks Between Tape Marks	Estimated Physical Volume Size
StorageTek Redwood	256 KB	512	50 GB
StorageTek Timberline	64 KB	1024	800 MB

The STK RAIT PVR cannot be supported at this time since STK has not yet made RAIT generally available.

In Table 2-4:

- Tape Type is the specific type of media to which the values in the row apply.
- Media Block Size is the block size to use in the storage class definition. For tape, this will be the size of the data blocks that will be written to tape. Note that for tape devices, the Mover configuration does not contain the media block size. The selected block size is set on a per physical volume basis at the request of the Tape Storage Server. This value may have a significant impact on data transfer performance, as for most tape devices each input/output request must be for the media block size. If a large block size is used for relatively small write requests, space may be wasted on the tape if the drive can not compress the unused portion of the data blocks.
- Blocks Between Tape Marks is the number of media blocks to be written between tape marks, to be set in the storage class definition. A relatively small value has the benefits of lower access time for position to the middle of a file (the start of a file will be just after a tape mark). Small values have the penalties of poorer media utilization and a performance penalty when writing tapes. Since files are usually read in their entirety, and since modern tape controllers are designed to stream data to tape, larger values of the Block Between Tape Mark parameter are recommended.
- Estimated Physical Volume Size is the estimated size of the physical volumes to be set in the storage class definition. These values are based on the expected media to be used with the specified type. In some cases, different length tape media may be used, which may have an effect on the estimated size for a given physical volume (e.g., regular or extended length 3480/3490 format cartridges). Note that the values listed do not take into account any data compression that may be performed by the tape drive. Also note that this value is for informational purposes only and does not effect the amount of user data written to a tape volume by the Tape Storage Server. The server fills each tape Virtual Volume and the amount of data written to the tape varies with the compressibility of the data.

2.9.2 Storage Hierarchy

Each HPSS file is stored in a single storage hierarchy consisting of an ordered list of storage classes. A storage hierarchy can have up to 4 levels starting with level 0. The highest level is always level 0 and the lowest possible level is level 4. For example, a level 0 storage class could be fast disk while a level 4 storage class could be a slow, large capacity tape system. The SSM provides operational capabilities to define storage hierarchies. A storage hierarchy is identified by a storage hierarchy ID and its associated attributes. For detailed descriptions of each attribute associated with a storage hierarchy, see Section 6.7.2: *Configure the Storage Hierarchies* (page 315). The following is a list of rules and guidelines for creating and managing storage hierarchies.

Rule 1: All writes initiated by clients are directed to the highest level (level 0) in the hierarchy.

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

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

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

Rule 5: The client stage command can only stage data to the top level (level 0) in the hierarchy.

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

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

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

2.9.3 Class of Service

Each HPSS file belongs to a single Class of Service (COS) that is selected when the file is created. It is selected via Class of Service Hints information passed to the Bitfile Server when the bitfile is created. If using the Client API, the application program has full access to this hints information. If using NFS, the COS is the same for each file in a fileset. If no class of services is assigned to the fileset, then the COS is determined by the COS ID that is specified by the NFS Daemon's configuration parameters. For FTP, there is a quote command to set the desired COS ID. A **pput** request in PFTP automatically selects a COS based on file size by using the COS Hints unless the user explicitly selects the COS. The SSM provides operational capabilities to define classes of service. A COS is identified by a COS ID and its associated attributes. For detailed descriptions of each attribute associated with a class of service, see Section 5.5.3.

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

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

2.9.3.1 Selecting Minimum File Size

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

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

2.9.3.2 Maximum File Size

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

Explanation: If the **Enforce Max File Size** option is selected, an attempt to perform an operation on a file that would cause this value to be exceeded will be rejected. The underlying storage hierarchy should be set up so that the defined storage classes support files of this size in a reasonable fashion. For details, see Sections 2.9.1 and 2.9.2 on storage class and storage hierarchies. This field can be used via the COS Hints mechanism to affect COS selection. PFTP and FTP clients that support the **alloc** command will use the size hints when creating a new file. Ensure that the Minimum File Size and the Maximum File Size do not overlap; otherwise the data placement may be indeterminate.

2.9.3.3 Selecting Stage Code

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

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

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

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

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

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

Explanation: When this option is selected, the file is staged up in increments and the read and write calls that are accessing this file are blocked only until that portion of the file they wish to operate on has been completely staged. Normally, this option would be selected when the top level is disk and the files in this class are fairly large in size. This option is only available when the top level in the hierarchy is disk. If the top level is tape and this option is specified, the **No Stage** option will be used instead.

Guideline 4: Select the **Stage on Open Background** option if you want the stage to be queued internally in the Bitfile Server and processed by a background BFS thread on a scheduled basis.

Explanation: The open request will return with success if the file is already staged. If the file needs to be staged an internal staged request is placed in a queue and will be selected and processed by the Bitfile Server in the background. A busy error is returned to the caller. This option allows a large number of stages (up to 2000) to be queued in the bitfile server and processed as thread resources are available. The other stage options will result in a busy error if thread resources are not immediately available to process the request.

Guideline 5: Select the **Retry Stage Failures from Secondary Copy** option if you want to configure a storage class so that stage failures to the primary copy can be automatically retried from a valid second copy. The associated hierarchy must first be configured for multiple copies.

Explanation: When a stage fails form the primary copy, HPSS will examine the error, and for most errors, it will examine the hierarchy for a valid second copy and reissue the stage from the second copy. A typical usage would be to bypass a damaged tape. In this case, the tape may be locked out via SSM and HPSS will recognize this and automatically go to a second copy if one exists. A warning that this has happened will appear in the SSM Alarms and Events window.

2.9.3.4 Selecting Optimum Access Size

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

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

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

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

2.9.3.5 Selecting Average Latency

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

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

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

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

2.9.3.6 Selecting Transfer Rate

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

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

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

2.9.3.7 StripeLength and StripeWidth Hints

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

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

2.9.4 File Families

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

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

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

2.10 HPSS Sizing Considerations

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

2.10.1 HPSS Storage Space

HPSS files are stored on the media that is defined to HPSS via the import and create storage server resources mechanisms provided by the Storage System Manager. You must provide enough physical storage to meet the demands of your user environment. HPSS assists you in determining the amount of space needed by providing SSM screens with information on total space and used space in all of the storage classes that you have defined. In addition, alarms can be generated automatically based on configurable threshold values to indicate when space used in a given storage class has reached some threshold level. In a hierarchy where data is being migrated from one storage class level to a lower one, management of space in the storage class provided is done via the migration and purge policies that you provide. The basic factors involved here are the total amount of media space available in the storage class being migrated and the rate at which this space is used. This will drive how the migration and purge policies are set up for the storage class. For more details on this, see Sections 2.8.1 and 2.8.2. Failure to have enough storage space to satisfy a user request results in the user receiving a NO SPACE error. One important factor in storage space growth relates to how HPSS handles write requests. The Bitfile Server always writes data to the top level in the hierarchy. When the top level is out of space, the Bitfile Server will not attempt to allocate space in a lower level and write the remaining data to the lower level.

2.10.2 HPSS Metadata Space

During the HPSS planning phase, it is important to properly assess how much disk space will be required to support the HPSS production environment. The first step in this process is to understand the various metadata files managed by each HPSS server. The sections that follow explain the metadata files used by each HPSS server and provide hints on how to best estimate how many records will be in each file. The second step in the process is to use the metadata sizing spreadsheet, explained in Section 2.10.2.21, which helps to calculate disk space requirements based on various HPSS sizing assumptions.

The naming convention for various SFS files has changed to accommodate this addition. All files suffixed with a ".#" are associated with a subsystem instance ranging from 1 to n. All remaining SFS files are uniquely named and therefore do not have a suffix. The following list of files are considered part of a subsystem:

acctlog.#

bfcoschange.#

bfdiskallocrec.#

bfdisksegment.#

bfmigrrec.#

bfpurgerec.#

bfsssegchkpt.#

bfssunlink.#

bftapesegment.#

bitfile.#

mpchkpt.#

nsacls.#

nsfilesetattrs.#

nsobjects.#

nstext.#

sspvdisk.#

sspvtape.#

storagemapdisk.#

storagemaptape.#

storagesegdisk.#

storagesegtape.#

vvdisk.#

vvtape.#

The following files are part of an HPSS system, but are not associated with a particular subsystem:

accounting

acctsnap

acctsum

acctvalidate

bfs

cartridge_3494

cartridge_3495

cartridge_aml

cartridge_lto

cartridge_operator

cartridge_stk

cartridge_stk_rait

The STK RAIT PVR cannot be supported at this time since STK has not yet made RAIT generally available.

cos

dmg

dmgfileset

filefamily

gkconfig

globalconfig

hierarchy

logclient

logdaemon

logpolicy

lspolicy

migpolicy

mmonitor

mountd

mover

moverdevice

mps

ndcg

nfs

nsconfig

nsglobalfilesets

purgepolicy

pvl

pvlactivity

pvldrive

pvljob

pvlpv

pvr

sclassthreshold

serverconfig

site

ss

storageclass

storsubsysconfig

2.10.2.1 Global Configuration Metadata

Global Configuration File (*globalconfig*) - This file contains configuration data that is global to an HPSS system. Since it only ever contains one small metadata record, it plays a negligible role in determining metadata size.

2.10.2.2 Storage Subsystem Metadata

Storage Subsystem Configurations (*storsubsysconfig*) - Each storage subsystem defined in HPSS has a record describing its default COS, associated Gatekeeper, and allowed COS ids. This file is small and will only grow when new subsystems are added. Therefore it plays a negligible role in determining metadata size.

2.10.2.3 Server Configuration Metadata

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

- General Server Configurations (*serverconfig*)
- BFS Configurations (*bfs*)
- DMAP Gateway Configurations (*dmg*)
- Gatekeeper Server Configurations (*gkconfig*)
- Log Client Configurations (*logclient*)
- Log Daemon Configurations (*logdaemon*)

- Metadata Monitor Configurations (*mmonitor*)
- Migration / Purge Server Configurations (*mps*)
- Mount Daemon Configurations (*mountd*)
- Mover Configurations (*mover*)
- NFS Daemon Configurations (*nfs*)
- Non-DCE Client Gateway Configurations (*ndcg*)
- NS Configurations (*nsconfig*)
- PVL Configurations (*pvl*)
- PVR Configurations (*pvr*)
- Storage Server Configurations (*ss*)

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

BFS Configurations. Each BFS must have an entry in the BFS configuration metadata file describing various startup / control arguments.

Gatekeeper Server Configurations. Each Gatekeeper Server (GK) must have an entry in this configuration metadata file describing various startup / control arguments. Zero or more GKs may be defined. If the GK is going to be used for Account Validation Services then at least one GK must be defined. If the GK is going to be used for Gatekeeping Services, then at least one GK must be defined and the GK also needs to be configured into the appropriate storage subsystems' configuration records.

NS Configurations. Each NS must have an entry in the NS configuration metadata file describing various startup / control arguments.

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

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

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

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

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

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

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

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

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

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

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

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

2.10.2.4 Name Server Metadata

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

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

Name Space Objects. Each name space object (file, fileset, directory, hard link, junction or soft link) uses one record in the object file. The total number of objects permitted in the name space is limited by the number of SFS records available to the Name Server. The number of SFS records available to the Name Server should therefore be large enough to handle the projected capacity of the name space. For example, in a name space expected to consist of 50,000 directories that have 1,000,000 files distributed among them, should have at least 1,050,000 SFS records allocated to the Name

Server. However, that would leave no space for any symbolic links, hard links, or growth. To cover these needs, the total number of SFS records might be rounded up to 1,500,000.

If more name space is needed, additional space can be obtained by allocating more SFS records, by adding more storage subsystems, and/or by “attaching” to a Name Server in another HPSS. Refer to Section 10.7.3: *Name Server Space Shortage* on page 272 in the *HPSS Management Guide* for information on handling an Name Server space shortage.

Access Control List Extensions. This metadata file is used to store overflow object ACL entries and to store any Initial Container or Initial Object ACL entries. An object with more than four object ACL entries (in addition to the standard **user_obj**, **group_obj**, and **other_obj** entries) will have one or more ACL records in the ACL overflow file. Each of these ACL extension records can contain up to 16 additional ACL entries. Any directory object that has Initial Container or Initial Object ACL entries will have one or more ACL records in the ACL overflow file. Sites that anticipate using a large number of ACLs of any type should ensure that sufficient SFS space is available for the ACL overflow file.

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

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

Global Filesets. This file is shared by all Name Servers running in a particular DCE cell. If more than one HPSS is running in a DCE cell or there is more than one storage subsystem within an HPSS system, each Name Server should share this file. The Global Filesets file is used to guarantee the uniqueness of fileset names and fileset identification numbers, and to identify which Name Servers and which DMAP Gateways manage which fileset.

2.10.2.5 Bitfile Server Metadata

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

- Storage Classes (*storageclass*)
- Hierarchies (*hierarchy*)
- Classes of Service (*cos*)
- Bitfiles (*bitfile.#*)
- Bitfile Disk Segments (*bfdisksegment.#*)
- Bitfile Disk Allocation Maps (*bfdiskallocrec.#*)

- Bitfile Tape Segments (*bftapesegment.#*)
- BFS Storage Segment Checkpoint (*bfsssegchkpt.#*)
- BFS Storage Segment Unlinks (*bfssunlink.#*)
- Bitfile COS Changes (*bfcoschange.#*)
- Bitfile Migration Records (*bfmigrrec.#*)
- Bitfile Purge Records (*bfpurgerec.#*)
- Accounting Summary Records (*acctsum*)
- Accounting Logging Records (*acctlog.#*)

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

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

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

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

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

To arrive at the best estimate of the number of records you need in the bitfile disk segment, bitfile tape segment, and bitfile disk map files, you need to take into account the way you have set up your classes of service and hierarchies. For example, suppose you set up a three-level hierarchy that has disk at the top level and two levels of tape. The migration policy calls for creating duplicate copies on tape. Assume that you have a 2 GB disk in the disk storage class and that the storage segment size is 512K. The maximum number of disk storage segments that could be created in this case would be 4,096. Thus you would need a maximum of 4,096 disk map records. Assume that you also plan to store 100,000 files in this hierarchy. You will have two tape copies for each file. Assuming an average of 2 bitfile tape segment records per file, you would end up creating 400,000 bitfile tape segment records. You may not have this level of detail, but the more you know, the more accurate your estimates can be.

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

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

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

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

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

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

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

Bitfile Migration Records. The Bitfile Server creates a migration record for a file that needs to be migrated and deletes this record when the migration is completed. In the worst case scenario, this file would contain a record for every bitfile that is stored on disk. If a site is able to estimate how many files are in an active and unmigrated state, then this file can be limited to this number.

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

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

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

2.10.2.6 Disk Storage Server Metadata

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

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

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

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

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

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

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

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

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

2.10.2.7 Tape Storage Server Metadata

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

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

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

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

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

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

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

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

2.10.2.8 Gatekeeper Metadata

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

- Account Validation File (*acctvalidate*)

Account Validation File. This metadata file is used by the Account Validation Services of the Gatekeeper to determine which users are authorized to use which account indices. This file is only used when Account Validation is enabled and Site-style accounting is being used. Each record contains information about a user, an account index number, and an account name. One or more users may share a single account index. This file must be maintained by the site using the Account Validation Metadata Editor (see Section 12.2.23: *hpss_avaledit — Account Validation Editor* on page 366 of the *HPSS Management Guide* for details).

2.10.2.9 PVL Metadata

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

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

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

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

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

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

2.10.2.10 PVR Metadata

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

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

The STK RAIT PVR cannot be supported at this time since STK has not yet made RAIT generally available.

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

2.10.2.11 Mover Metadata

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

- Mover Devices (*moverdevice*)

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

2.10.2.12 Migration/Purge Server Metadata

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

- Threshold Policies (*sclassthreshold*)
- Migration/Purge Checkpoints (*mpchkpt.#*, where # is the storage subsystem ID)

Threshold Policies. Each storage subsystem specific storage class threshold requires a record in this file. The maximum number of records is the product of the number of storage subsystems and the number of storage classes. This file name is configured on the HPSS Global Configuration window.

Migration/Purge Checkpoints. MPS creates a single checkpoint record in this file for each disk storage class which is configured for migration (i.e. with a migration policy). This checkpoint record is used to round-robin the starting hierarchy for each disk migration run. This file name is configured on the Migration/Purge Server Configuration window.

The MPS shares the following SFS metadata files with the Bitfile Server (the default SFS file names are shown in parentheses).

- Migration Policies (*migpolicy*)
- Purge Policies (*purgepolicy*)
- Migration Records (*bfmigrrec.#*, where # is the storage subsystem ID)
- Purge Records (*bfpurgerec.#*, where # is the storage subsystem ID)

Migration Policies. Each basic migration policy and each storage subsystem specific migration policy requires a record in this file. The maximum number of records is the product of the number of storage subsystems and the number of storage classes. This file name is configured on the HPSS Global Configuration window.

Purge Policies. Each basic purge policy and each storage subsystem specific purge policy requires a record in this file. The maximum number of records is the product of the number of storage subsystems and the number of storage classes. This file name is configured on the HPSS Global Configuration window.

Migration Records. The BFS in a storage subsystem uses this file to indicate to the MPS in the subsystem which disk files are migration candidates. Each disk file in the subsystem which is a migration candidate requires one migration record in this file. Once a file is migrated, its migration record is removed. This file name is configured on the Storage Subsystem Configuration window.

Purge Records. The BFS in a storage subsystem uses this file to indicate to the MPS in the subsystem which disk files are purge candidates. Each disk file in the subsystem which is a purge candidate requires one purge record in this file. Once a file is purged, its purge record is removed. This file name is configured on the Storage Subsystem Configuration window.

2.10.2.13 Logging Services Metadata

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

- Log Policies (*logpolicy*)

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

2.10.2.14 Metadata Monitor Metadata

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

2.10.2.15 Storage System Management Metadata

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

- File Family (*filefamily*)

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

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

2.10.2.16 NFS and Mount Daemons Metadata

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

2.10.2.17 DMAP Gateway Metadata

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

- DMAP Gateway Filesets (*dmgfileset*)

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

2.10.2.18 Location Server Metadata

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

- Location Policy (*lspolicy*)
- Remote Site Policy (*site*)

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

2.10.2.19 Non-DCE Client Gateway

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

2.10.2.20 Metadata Constraints

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

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

2.10.2.21 Metadata Sizing Spreadsheet

To help with projecting disk space requirements for HPSS metadata, an Excel spreadsheet is available on the HPSS web site at **`https://www4.clearlake.ibm.com/hpss/support/ToolsRepository/md43.xls`**. To access the website, an account is required. To request an account, use URL **`http://www4.clearlake.ibm.com/cgi-bin/hpss/requests/id_request.pl`**. The spreadsheet is divided into two worksheets: the first defines the various HPSS and SFS sizing assumptions; the second calculates the sizing estimates.

2.10.2.21.1 Metadata Sizing Assumptions

The metadata sizing assumptions are organized into multiple sections: HPSS dynamic variables, HPSS dynamic variables for each subsystem, HPSS static configuration variables, and SFS sizing assumptions. Note that all assumptions are with respect to sizing the maximum amount of metadata required to support a planned configuration. It is recommended that sites use values for multiple years into a project’s future to determine how the metadata sizing requirements will increase with system growth.

HPSS Dynamic Variables. Table 2-5 defines the HPSS dynamic variables associated with the HPSS system outside those specific to a subsystem. These variables will moderately impact the metadata sizing estimates.

Table 2-5 HPSS Dynamic Variables (Subsystem Independent)

Variable	Description
Total Number of Users	Defines the total number of users in the HPSS system. This value is used in conjunction with “Avg. Number of Levels Per Hierarchy” to define the size of accounting records.
Avg. Number of Levels Per Hierarchy	This value is the average number of levels defined in the hierarchies. For most hierarchies which are defined as disk and tape, this value would be 2.
Global Number of Filesets	The total number of filesets that will be managed by all Name Servers in a given DCE Cell.
Max Filesets Per DMAP Gateway	This value is the maximum number of filesets that a DMAP Gateway can be expected to manage. This value can have an impact on the overall storage sizing estimate.
Max Queued PVL Jobs	The maximum number of jobs that might be queued in the PVL at any one time.
Avg. Activities Per PVL Job	Each PVL job has one or more activities associated with it (mount tape #1, mount tape #2, etc. for a 2-way tape stripe). This variable defines the average number of PVL activities per PVL job, which should represent the average stripe width for tape virtual volumes.
Total PVR #1 Tape Cartridges	The spreadsheet allows up to three PVR’s to be defined, each with a different number of cartridges. This field represents the maximum number of cartridges that will be managed by PVR #1.
Total PVR #2 Tape Cartridges	If a second PVR will be used, enter the maximum number of cartridges managed by the second PVR; otherwise, use zero.
Total PVR #3 Tape Cartridges	If a third PVR will be used, enter the maximum number of cartridges managed by the third PVR; otherwise, use zero.

HPSS Dynamic Variables for each Subsystem. Table 2-6 defines the HPSS dynamic variables associated with each HPSS subsystem. All systems will have a “primary subsystem” and this section of the spreadsheet must be completed. Values for subsystem #2 and #3 will need to be filled out only if additional subsystems are planned. Many of these variables contribute significantly to the size of the metadata estimate.

Table 2-6 HPSS Dynamic Variables (Subsystem Specific)

Variable	Description
Max Total Bitfiles	The maximum number of bitfiles that will exist in HPSS. The spreadsheet also considers this value to also be the total number of bitfiles on tape, since it is assumed that every HPSS bitfile will eventually migrate to tape. If it is expected for 2 million files to be placed in HPSS, enter 2,000,000. This value significantly impacts the overall metadata sizing estimate.
Max Bitfiles on Disk	Used to determine how many bitfiles might reside on disk at any point in time. This number must be less than or equal to *Max Total Bitfiles*. For example, if disk space is sufficient to hold/cache 20,000 of the 2 million total files, enter 20,000. This value significantly impacts the overall metadata sizing estimate.
Avg. Copies Per Bitfile	Defines the average number of copies per bitfile (since HPSS supports duplicate bitfile copies). For example, if there is approximately 1 extra bitfile copy for every 100 bitfiles created, enter a value of 1.01. This value significantly impacts the overall metadata sizing estimate. If not using duplicate bitfile copies, use a value of 1.0.
Percent of Extra Copies Stored on Tape	When duplicate bitfile copies are created, the extra copy(s) are written to a lower level in the storage hierarchy. Usually, lower levels are defined as tape storage classes. Enter the average percentage of these extra bitfile copies that will be stored on tape (versus another disk storage class). This value significantly impacts the overall metadata sizing estimate if the duplicate copy feature is used.
Max Accounting Log Records	This value is the maximum number of log records that is expected at any given time when the BFS is updating the (acctsum) file. The number of records will depend upon system load, but should typically not exceed 3000.
Avg. Name Space Objects Per Bitfile	One name-space object is created for each HPSS bitfile; however, other name space objects, such as directories and links, are also created. This value defines, on average, how many name space entries there are per HPSS bitfile. For example, if there is one additional name-space object created for each 20 bitfiles (directory or link) on average, a value of 1.05 would be entered. This value significantly impacts the overall metadata sizing estimate.
Avg. Object ACL Entries Per Name Space Object	A name-space object record can store up to 4 Object ACL entries (in addition to the user_obj, group_obj, and other_obj ACL entries). If an object has more than four Object ACL entries, an ACL record must be created. Each ACL record can hold up to 16 ACL entries. This value defines, on average, how many ACL records will be created per name-space object. For example, if 1 in 100 name-space objects will have between 5 and 20 ACL entries, a value of .01 would be entered.
Avg. Initial Container and Initial Object ACL Entries Per Name Space Directory Object	If Initial Container (IC) and/or Initial Object (IO) ACL entries are used, at least one ACL record will be created for each type of ACL. Each ACL record can hold up to 16 ACL entries. This value defines, on average, how many IC/IO ACL records will be created per name-space directory object. For example, if 5 in 100 name-space objects will have between 1 and 16 IC/IO ACL entries, a value of .1 would be entered.

Table 2-6 HPSS Dynamic Variables (Continued)(Subsystem Specific)

Variable	Description
Avg. Text Overflows Per Name Space Object	A name-space object record can store a filename that is 23 characters long (the base name, not the full pathname). If a filename is longer than 23 characters, a text overflow record must be generated. Also, if a comment is attached to a name-space object, a text overflow record must be created. This value defines, on average, how many text overflow records will be created per name-space object. For example, if 5 in 100 name-space objects will either have a filename longer than 23 characters or a comment attached, a value of .05 would be entered.
Avg Length of Text Overflows	This is the average length of each name space object which will exceed 23 characters. This value in conjunction with "Avg Text Overflows Per Name Space Object" is used to determine the amount of metadata in 'nstext' required to store all text overflows. This value can significantly impact the size of the metadata estimate.
Filesets per Name Server	The number of filesets a Name Server will be managing.
Avg. Bitfile Segments Per Tape Bitfile	The average number of bitfile segments created for each tape bitfile. When files are migrated to tape from disk, there normally will be only one tape bitfile segment for the file. However, if the file is partially modified, additional bitfile segments will be migrated to tape. This value significantly impacts the overall metadata sizing estimate.
Avg. Storage Segments Per Disk VV	The average number of storage segments that will be created per disk virtual volume. Since there are a maximum of 16,384 blocks per disk VV, and each storage segment uses one or more VV blocks, the number must be less than or equal to 16,384. This value can be determined by dividing the average disk VV size by the average storage segment size.
Avg. Storage Segments Per Disk Alloc Rec	Each disk allocation record stores information for up to 8 disk storage segments. If, on average, 1 extra disk allocation record is required per 5 disk bitfiles, a value of 1.2 would be entered.
Avg. Storage Segments Per Tape Bitfile	The average number of storage segments per tape bitfile. If files are only written to tape through migration, the value can be 1, assuming the storage class characteristics have been set up properly. If users are allowed to write directly to tape, the average should be something higher than 1.
Max BFS Storage Segment Checkpoints	The maximum number of storage segment checkpoint records created by BFS. A value of 500 should be sufficient for planning purposes.
Max Queued Bitfiles Changing COS	The maximum number of bitfiles that have a pending class of service change. A value of 1,000 should be sufficient for planning purposes.
Max Queued BFS Storage Segment Unlinks	The maximum number of storage segments that could be queued to be unlinked. A value of 1,000 should be sufficient for planning purposes.
Max Disk Bitfiles Queued for Migration	The maximum number of unmigrated bitfiles that may exist on disk at one time. The value should generally equal *Max Bitfiles on Disk*.
Max Disk Bitfiles Queued for Purging	The maximum number of migrated bitfiles that may exist on disk at one time. The value should generally equal *Max Bitfiles on Disk*.

Table 2-6 HPSS Dynamic Variables (Continued)(Subsystem Specific)

Variable	Description
Total Disk Physical Volumes	The maximum total number of disk physical volumes.
Total Disk Virtual Volumes	The total number of disk virtual volumes that will be created, which is equal to the number of disk physical volumes divided by the average disk stripe width.
Total Tape Physical Volumes	This is the total number of Physical Volumes that will be managed by this subsystem.
Total Tape Virtual Volumes	The total number of tape virtual volumes that will be created, which is equal to the total number of all PVR cartridges divided by the average tape stripe width.

HPSS Static Configuration Variables. Table 2-7 defines the HPSS static configuration variables, which have more of a static nature in that they are not particularly relevant to how large the HPSS system will be. They relate more to how many HPSS servers, storage classes, etc. will be defined and therefore are not significant in the overall metadata size projection.

Table 2-7 HPSS Static Configuration Values

Variable	Description
Total Subsystems	The number of subsystems controlled by this HPSS instance.
Total Storage Classes	The maximum number of storage classes that will be defined.
Total Storage Hierarchies	The maximum number of storage hierarchies that will be defined.
Total Storage Families	The maximum number of file families that will be defined.
Total Classes of Service	The maximum number of classes of service that will be defined, which typically will equal the total number of storage hierarchies created.
Total Tape Drives	The total number of tape drives used by HPSS.
Total HPSS Servers	The overall number of HPSS servers that will be defined. This includes the Name Server, BFS, Storage Servers, Log Daemon, Log Clients, Startup Daemons, Movers, etc.
Total BFS Servers	The total number of BFS servers that will be configured. This value should be equal to the number of subsystems.
Total Number of DMAP Gateways	This defines the total number of DMAP Gateway Servers that will be defined in the system.
Total Name Servers	The total number of Name Servers that will be created. This value should be equal to the number of subsystems.

Table 2-7 HPSS Static Configuration Values (Continued)

Variable	Description
Total Log Clients	The total number of Log Clients that will be used, which will be equal to the total number of nodes running any type of HPSS server.
Total Metadata Monitor Servers	The total number of Metadata Monitor servers, which should equal the total number of Encina SFS servers used by the HPSS system.
Total Movers	The total number of Movers that will be created.
Total NFS Mount Daemons	The total number of NFS Mount Daemons that will be configured.
Total NFS Servers	The total number of NFS servers that will be configured.
Total PVR Servers	The total number of PVR servers that will be used.
Total Storage Servers	The total number of Storage Servers. Normally, sites will have at least one disk Storage Server and at least one tape Storage Server.
Total Migration/ Purge Servers	The total number of Migration/Purge Servers. Normally, sites will have one MPS server per subsystem.
Total Gatekeeper Servers	The total number of Gatekeeper Servers that will be used. Sites may have 0 or more Gatekeeper Servers.

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

2.10.2.21.2 Sizing Computations

The second worksheet in the metadata sizing spreadsheet calculates the projected total number of records for each metadata file and the corresponding amount of required disk space, based on the assumptions previously entered. The spreadsheet allows the record count for any given metadata file to be manually overridden, if desired, and allows the required disk space to be allocated to one of 15 SFS data volumes, allowing individual SFS data volumes to be properly sized. For systems which will easily exceed several GBs in size, it is recommended that each site consider moving files associated with subsystems to separate SFS servers. The data volume selection is not necessarily restricted to just one SFS instance. It is a general way to distinguish files from one data volume from another whether of the same SFS server or not.

The various columns in this worksheet are described below.

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

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

The following are subsystem independent:

ACCT / Accounting Policies (accounting) = 1

ACCT / Summary Records (acctsum) = Total Number of Users * (Avg Number of Levels Per Hierarchy + 1)

ACCT / Snapshot Records (acctsnap) = Total Number of Users * (Avg Number of Levels Per Hierarchy + 1)

ACCT / Validation Records (acctvalidate) = Total Number of Users * (Avg Number of Levels Per Hierarchy + 1)

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

BFS / Server Configs (bfs) = Total BFS Servers

BFS / Storage Classes (storageclass) = Total Storage Classes

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

BFS / Storage Hierarchies (hierarchy) = Total Storage Hierarchies

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

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

GK / Server Configs (gkconfig) = Total Gatekeeper Servers

LOC / Location Server Policies (lspolicy) = 1

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

LOG / Log Daemon Server Configs (logdaemon) = 1

LOG / Log Policies (logpolicy) = Total HPSS Servers

MMON / Servers (mmonitor) = Total Metadata Monitor Servers

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

MOVR / Server Configs (mover) = Total Movers

MPS / Migration Policies (migpolicy) = Total Storage Classes

MPS / Purge Policies (purgepolicy) = Total Storage Classes

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

NDCG Non-DCE Gateway Configuration = Total Non-DCE Gateways

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

NFS / Server Configs (nfs) = Total NFS Servers

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

NS / Server Configs (nsconfig) = Total Name Servers

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

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

PVL / Jobs (pvljob) = Max Queued PVL Jobs

PVL / Physical Volumes (pvlpv) = Total Disk Physical Volumes + Total PVR #1 Tape Cartridges + Total PVR #2 Tape Cartridges + Total PVR #3 Tape Cartridges

PVL / Server Configs (pvl) = 1

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

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

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

PVR / Server Configs (pvr) = Total PVR Servers

Remote Site Configs (site) = Total Remote Sites

SS / Server Configs (ss) = Total Storage Servers

SSM / Server Configs (serverconfig) = Total HPSS Servers

SSM / Storage Subsystem Configs (storsubsysconfig) = Total Subsystems

SSM / Subsystem Storage Class Thresholds (sclassthreshold) =Total Subsystems * Total Storage Classes

For each subsystem, compute the following:

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

BFS / Bitfiles (bitfile.#) = Max Total Bitfiles

BFS / Change COS Records (bfcoschange.#) = Max Queued Bitfiles Changing COS

BFS / Disk Alloc Records (bfdiskallocrec.#) = [Computed] Disk Bitfile Segments / Avg. Storage Segments Per Disk Alloc Rec

BFS/Disk Bitfile Segments (bfdisksegment.#) = Avg. Storage Segments Per Disk VV * Total Disk Virtual Volumes

BFS/Storage Segment Checkpoint (bfsssegchkpt.#) = Max BFS Storage Segment Checkpoints

BFS/Tape Bitfile Segments (bftapesegment.#) = Avg Bitfile Segments Per Tape Bitfile * (Max Total Bitfiles + (Max Total Bitfiles * (Avg Copies Per Bitfile - 1) * Percent of Extra Copies Stored on Tape))

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

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

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

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

NS/ACL Extensions (nsacls.#) = [Computed] NS Objects * Avg. Object ACL Entries Per Name Space Object + (Name Space Directory Objects * Avg. Initial Container and Initial Object ACL Entries Per Name Space Directory Object)

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

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

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

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

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

SS/Disk Storage Segments (storagesegdisk.#) = Avg Storage Segments Per Disk VV * Total Disk Virtual Volumes

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

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

SS/Tape Physical Volumes (sspvtape.#) = Total PVR #1 Tape Cartridges + Total PVR #2 Tape Cartridges + Total PVR #3 Tape Cartridges

SS/Tape Storage Segments (storagesegtape.#) = Avg Storage Segments Per Tape Bitfile * (Max Total Bitfiles + (Max Total Bitfiles * (Avg Copies Per Bitfile - 1) * Percent of Extra Copies Stored on Tape))

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

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

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

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

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

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

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

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

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

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

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

2.10.2.22 Encina SFS Disk Space

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

- Encina SFS Data

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

The metadata sizing spreadsheet, discussed in the previous section, needs to be used to calculate the size of each SFS data volume and the distribution of the SFS data files across the data volumes.

Any SFS data file that is estimated to be greater than 1 GB needs to reside on its own SFS data volume.

- Encina Transaction log

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

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

The size of the transaction log is dependent on the volume of HPSS transactions. Start with no less than 256 MB (512 MB for larger installations). It is best to start small and increase the size if necessary.

A large log volume will increase SFS start time.

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

- Encina Media Recovery Archive Files (MRA files)

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

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

If the file systems used to store the MRA files becomes full, SFS will not run.

MRA files are written to the directory **/opt/encinalocal/encina/sfs/hpss/archives**. Before Encina is configured as described in Section 5.5.2: *Configure Encina SFS Server* (page 243), a separate mirrored file system should be created (e.g. **/sfsbackups/mra**), and an appropriate link created from the directory mentioned above. For example, if the file system was called **/sfsbackups/mra**, then create the following link:

```
% ln -s /sfsbackups/mra \
        /opt/encinalocal/encina/sfs/hpss/archives
```

The MRA files should not be on the same disk as the Encina transaction log or the Encina data files.

- Encina SFS Backup Files (TRB files)

SFS has the ability to generate on-line backup files while continuing to service HPSS metadata requests. It is important to generate regular backups of SFS in case of future disk failures.

The HPSS utility, **sfsbackup**, should be used to generate the Encina SFS backup files, sometimes referred to as the TRB files. A single TRB file will be generated for each Encina SFS data volume on each run of the utility. TRB files should be stored on a separate file system that is mirrored (e.g. **/sfsbackups/trb**). The TRB files should not be on the same disk as the MRA files, Encina transaction log, or Encina data files. If you are limited by the number of physical volumes (disk drives) that are available for your HPSS system, you can put the MRA file system and TRB file system on the same disk as long as the file systems are both mirrored (this is NOT the recommended configuration). It is recommended that at least 2 GB of disk space be available for the TRB files.

sfsbackup is also used to backup and manage the MRA files. As TRB files are generated, the older MRA files are no longer needed. The **sfs_backup_util** should be configured to remove the unneeded MRA and TRB files.

If the file systems used to store the TRB files becomes full, the sfsbackup will fail.

2.10.3 System Memory and Disk Space

2.10.3.1 Disk Space Requirements for HPSS Installation

The HPSS software is installed in the **/opt/hpss** directory. The installation package sizes and disk requirements are listed in Table 4-1: *Installation Package Sizes and Disk Requirements* on page 206.

2.10.3.2 Disk Space Requirements for Running HPSS Servers

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

2.10.3.2.1 Disk Space Requirements for Core files and Encina Trace Buffers

The **/var/hpss/adm/core** is the default directory where HPSS creates core and Encina trace files resulting from subsystem error conditions. The actual size of the files differ depending on the subsystems involved, but it is recommended that there should be at least 512 MB reserved for this purpose on the core server node and at least 256 MB on Mover nodes. This directory should also be analyzed periodically to remove any old core files or Encina trace buffers.

2.10.3.2.2 Disk Space Requirements for HPSS Files

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

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

2.10.3.2.3 Disk Space Requirements for Running Log Daemon

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

2.10.3.2.4 Disk Space Requirements for Running Log Client

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

2.10.3.2.5 Disk Space Requirements for Running MPS

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

2.10.3.2.6 Disk Space Requirements for Running GK

If the Gatekeeper Server is configured to do Gatekeeping Services, then the site may wish to create a site policy configuration file specified in the Gatekeeper Server configuration record, usually **/var/hpss/gk/gksitepolicy**. The size of this file depends on the site-implemented gatekeeping policy.

2.10.3.2.7 Disk Space Requirements for Running an Accounting Report

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

2.10.3.2.8 Disk Space Requirements for Running NFS Daemon

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

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

Directory name memory requirements are determined by multiplying the estimated number of cached directories by the directory size configuration. Estimating the number of cached directories can be done by multiplying the number of entries in the attribute cache LRU list by the percentage of directories to files at the site. For example, if the LRU list maximum is set to 100 and the percentage of directories at the site is 1 directory to 10 files (10%), the estimated number of cached directories is 10.

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

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

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

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

2.10.3.2.9 Disk Space Requirements for Running SSM

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

2.10.3.3 System Memory and Paging Space Requirements

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

For nodes that will run one or more of the HPSS servers (and most likely an SFS server), excluding the Movers, it is recommended that at least 1 GB of memory be configured. However, more memory will certainly be required for systems that run most of the servers on one node and/or are supporting many concurrent users. The availability of memory to the HPSS servers and SFS servers will be critical to providing acceptable response time to many end user operations. Disk space requirements are primarily covered by Section 2.10.2, for SFS space, and the preceding subsections under Section 2.10.3 for the individual HPSS servers. In addition, sufficient disk space should be available for paging space based on the recommendations made in the system documentation for the amount of memory configured. Also, be aware that running many SAMMI clients on a machine will quickly consume available system memory.

For nodes that will only run Movers and no SFS server, the amount of memory is dependent on the number and types of devices configured on those nodes, the expected usage of those devices, and the configuration of the Movers on those nodes. In general, Movers supporting a disk device will require more memory than Movers supporting tape devices. This is because many outstanding requests are likely for a disk device, but there is usually only one outstanding request for a tape device. The size of the buffers used internally by the Mover will have an impact on the Mover memory requirements because each Mover request process will use two buffers to handle I/O requests. It is recommended that at least 512 MB of memory be configured for these nodes. However, more memory will certainly be required for nodes that are supporting many devices, especially disk devices that will support a large number of concurrent end-user requests.

Paging space should be sized according to the following rules:

Table 2-8 Paging Space Info

Amount of physical memory	Minimum recommended amount of paging space
< 256MB	2 * amount of physical memory
= 256MB and < 1GB	512MB + ((amount of physical memory – 256MB) * 1.25)
= 1GB and < 2GB	1.5 * amount of physical memory
= 2GB	1 * amount of physical memory

2.11 HPSS Performance Considerations

This section provides some additional hints for enhancing HPSS performance:

2.11.1 DCE

Due to several problems observed by HPSS, it is highly recommended that all DCE client implementations use the **RPC_SUPPORTED_PROTSEQS=ncadg_ip_udp** environment variable. Frequent timeouts may be observed if this is not done. Each HPSS system should be periodically checked for invalid/obsolete endpoints. Failure to comply may cause miscellaneous failures to occur as well as significantly decreasing HPSS performance.

DCE will attempt to utilize every available network interface when attempting RPC communication. This can cause performance problems when using multi-homed machines. It is highly recommended that all DCE client and server machines use the **RPC_UNSUPPORTED_NETIFS** environment variable to specify those interfaces that should *not* be used for RPC communications.

A Global Directory Access Daemon (GDAD) is required in all cooperative DCE Cells. The **gdad** uses information from the Domain Name Service, DNS, to resolve communication requirements between DCE Cells. In DCE 3.1, the dced maintains the executing state of the local gdad. Both the AIX and SOLARIS DCE implementations of gdad provide the ability to initiate the **gdad** with a **"–r filename"** flag allowing the specification of a specific **"resolv.conf"** file to be used by **gdad.**

2.11.2 Encina

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

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

2.11.2.1 SFS Server Parameters

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

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

2.11.3 Workstation Configurations

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

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

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

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

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

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

2.11.4 Bypassing Potential Bottlenecks

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

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

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

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

2.11.5 Configuration

The configuration of the HPSS storage resources (see Section 2.8.6) is also an important factor in overall HPSS performance, as well as how well the configuration of those resources matches the client data access patterns.

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

2.11.6 FTP/PFTP

Data transfers performed using the standard FTP interface are primarily affected by the buffer size used by the FTP Daemon. The buffer size can be configured as described in Section 7.3: *FTP Daemon Configuration* (page 422). It should be a multiple of the storage segment size, if possible. If not, it should be, at least, a multiple of the virtual volume block size. If the buffer size is too small, the FTP Daemon will need to issue a large number of individual read or write requests; however, if the buffer size is too large, the FTP Daemon will require a large amount of memory, which may cause additional paging activity on the system.

The size of the FTP Daemon buffer is extremely important if the FTP clients write files directly to a tape storage class, as described in Section 2.11.5.

Parallel FTP (PFTP) can be used to move data using TCP/IP.

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

Note that PFTP clients that use the standard FTP data transfer commands (e.g., **put** and **get**) have the same performance considerations as standard FTP clients as described in Section 2.11.6.

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

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

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

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

2.11.7 Client API

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

The size of the transfers is extremely important because the clients may write files directly to a tape storage class, as described in Section 2.11.5.

2.11.8 Name Server

Minor performance degradations may be seen when the Name Server is processing path names with a large number of components, listing directories containing entries with long (greater than 23 characters) names, servicing requests for objects which have a large (greater than 4) number of Object ACL entries, and servicing requests for objects which have Initial Container/Initial Object ACL entries.

2.11.9 Location Server

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

The Location Server itself will give warning when a problem is occurring by posting alarms to SSM. Obtain the information for the Location Server alarms listed in the *HPSS Error Manual*. To get a better view of an alarm in its context, view the Location Server's statistics screen.

If the Location Server consistently reports a heavy load condition, increase the number of request threads and recycle the Location Server. Remember to increase the number of threads on the Location Server's basic server configuration screen as well. If this doesn't help, consider replicating the Location Server on a different machine. Note that a heavy load on the Location Server should be a very rare occurrence.

2.11.10 Logging

Excessive logging by the HPSS servers can degrade the overall performance of HPSS. If this is the case, it may be desirable to limit the message types that are being logged by particular servers. The Logging Policy can be updated to control which message types are logged. A default Log Policy may be specified to define which messages are logged. Typically, Trace, Security, Accounting, Debug, and Status messages are not logged. Other message types can also be disabled. Once the Logging Policy is updated for one or more HPSS servers, the Log Clients associated with those servers must be reinitialized.

2.11.11 MPI-IO API

MPI-IO client applications must be aware of HPSS Client API performance on certain kinds of data transfers (see Section 2.11.7), which is basically that HPSS is optimized for transferring large, contiguous blocks of data.

The MPI-IO interface allows for many kinds of transfers that will not perform well over the HPSS file system. In particular, MPI-IO's use of file types enables specification of scatter-gather operations on files, but when performed as noncollective reads and writes, these discontiguous accesses will result in suboptimal performance in the best case, and in failure to complete the access, if excessive file fragmentation results, in the worst case. Collective I/O operations may be able to minimize or eliminate performance problems that would result from equivalent noncollective I/O operations by coalescing discontiguous accesses into a contiguous one.

An MPI-IO application should make use of HPSS environment variables (see Section 7.1: *Client API Configuration* on page 413) and file hints at open time to secure the best match of HPSS resources to a given task, as described in the *HPSS Programmers Reference, Volume 1*.

HPSS MPI-IO includes an automatic caching facility for files that are opened using **MPI_MODE_UNIQUE_OPEN** when each participating client node has a unique view of the opened file. When caching is enabled, performance for small data accesses can be significantly improved, provided the application makes reasonable use of locality of references. MPI-IO file caching is described in the *HPSS Programmer's Reference, Volume 1*.

2.11.12 Cross Cell

Cross Cell Trust should be established with the minimal reasonable set of cooperating partners (N-squared problem). Excessive numbers of Cross Cell connections may diminish Security and may cause performance problems due to Wide Area Network delays. The communication paths between cooperating cells should be reliable.

Cross Cell Trust must exist to take advantage of the HPSS Federated Name Space facilities.

2.11.13 DFS

DFS performance for HPSS is dependent on a number of factors: fileset type (mirrored or archived), CPU performance, memory throughput rates, DFS client caching, etc. Mirrored filesets will perform at HPSS rates for name space changes. Name space accesses will perform at normal DFS rates. Archived filesets will typically perform close to DFS rates for both name space changes and accesses. For both mirrored and archived filesets, access and changes will perform at DFS rates when data is resident, but will be delayed if HPSS must stage the data onto the Episode disks.

When setting up an aggregate, it is suggested that the fragment size be set to 1024 and the blocksize be set to 8192. These are the defaults and have been tested much more thoroughly than other settings. An important factor to consider is that any file smaller than the blocksize currently can not be purged from the Episode disk, and setting the blocksize larger than 8192 may cause space and resource problems on the disk. (This is a limitation of the Episode implementation of XDSM on which the HPSS/DFS code is implemented).

During testing the biggest gains were realized by altering the DFS client caching. A large memory cache instead of a disk cache may improve performance dramatically if the client machine can spare memory for client caching buffers. For more information on DFS configuration for AIX please refer to the DCE document "*Distributed File Service Administration Guide and Reference*".

Since HPSS must read DFS anodes to determine which files to migrate or purge, it is suggested that aggregates kept to a maximum size of 250,000 files and directories. This will allow the migration and purge algorithms to determine which files to process in a reasonable amount of time. With current Episode limitations on the amount of space allowed for anodes per aggregate and the design of the HPSS DFS code, the maximum number of anodes per HPSS-managed aggregate is around 1,000,000 files (2GB / 2K per migrated file). When planning the system, assume that it may not be possible to expand an aggregate to accommodate more files when this limit is reached. It may be necessary to add a new aggregate. In fact, since migration and purge are aggregate-based, the system may perform better with data distributed among a larger number of well-balanced aggregates than with all data concentrated a few large ones.

2.11.14 XFS

XFS performance for HPSS is dependent on a number of factors: CPU performance, available memory, disk speeds, etc.. XFS archived filesets will typically perform close to native XFS rates for both namespace and data activity; however, accessing or modifying data for files which have been migrated to HPSS and purged from XFS will be delayed while the file's data is staged.

The XFS HDM keeps an internal record of migration and purge candidates and will therefore be capable of quickly completing migration and purge runs which would otherwise take a good deal of time. This makes a 'migrate early, migrate often' strategy a feasible way to keep XFS disks clear of inactive data.

The only inherent size limitation for XFS is a 2 TB maximum filesystem size, which is a limitation of the Linux kernel. The file size limit is 16-64TB (depending on page size) which is limited in practice by the maximum filesystem size.

2.11.15 Gatekeeping

Sites may choose to implement site policy in the Gatekeeper Server for load balancing create, open, and / or stage requests. The site policy could limit the maximum number of non-AuthorizedCaller requests allowed at once by either delaying or denying particular requests. To delay the request, the site policy may return a special retry status along with the number of seconds to wait before the Client API retries the request. Delaying requests should limit the number of create, open, and / or stage requests performed at a particular point in time, thus decreasing the load on the system. However, care must be taken to figure out the best retry wait scheme to meet the requirements for each site and to configure the correct number of Gatekeeper Servers if the load on one Gatekeeper Server is heavy. (Note: The maximum number of Gatekeeper Servers per storage subsystem is one.) Also, sites need to write their Site Interfaces optimally to return in a timely manner.

Two special error status codes (**HPSS_ETHRESHOLD_DENY** and **HPSS_EUSER_DENY**) may be used to refine how a site may deny a create, open, or stage requests. If the BFS receives either of these errors, then it will return this error directly to the Client API rather than performing a retry. Errors other than these two or the special **HPSS_ERETRY** status will be retried several times by the BFS. See either volume of the *HPSS Programmer's Reference* for more information.

Create, open, and stage requests from Authorized Callers (DMG, MPS, NFS) can NOT be delayed or denied due to timing sensitivity of the special requests these servers make to the BFS. For example, migration of a file by MPS is an Authorized Caller Open request. The site policy could keep track of Authorized Caller requests to further limit non-AuthorizedCaller requests.

If a Gatekeeper Server is being used for Gatekeeping Services, then the BFS for each storage subsystem configured to use a particular Gatekeeper Server will return errors for the create, open, and / or stage requests being monitored by that Gatekeeper Server when that Gatekeeper Server is down. For example, if storage subsystem #2 is configured to use Gatekeeper Server #2, and Gatekeeper Server #2 is monitoring open requests and is DOWN, then each open by the BFS in storage subsystem #2 will eventually fail after retrying several times.

2.12 HPSS Metadata Backup Considerations

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

Also, if running a Highly Available HPSS, make sure to review Section G.7: Metadata Backup Considerations on page 569 as it covers issues related to reliable metadata backup in an HA environment.

The remainder of this section is a set of "rules" associated with backing up HPSS metadata. Though tools like **sfsbackup** as well as site specific generated scripts and procedures can be used to backup and protect the SFS metadata, it is important that each site review this list of “rules” and check to insure that their site’s backup is consistent with these policies.

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

1. At a minimum, the transaction log and SFS data files should be on different disks.
2. Ideally, several physical disks should be available for SFS data files (NS data can be on one, BFS data on another, etc.).
3. The media log archiving is simply a running backup of the transaction log. The transaction log is wrap-around, but the media log archives are not. A large amount of data is generated by these archives and must routinely be backed up and removed. A separate disk should be devoted to the log archives, if possible.

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

2.12.1 Rules for Backing Up SFS Log Volume and MRA Files

- Media archiving must always be enabled while HPSS is running in production mode. If it is ever temporarily disabled (e.g., during a one-time UniTree migration), then before HPSS is placed back into production mode, media archiving must be re-enabled and a complete backup of all SFS data volumes must be made using TRB files.
- The SFS log volume must be mirrored on at least two separate physical disks or be stored on a redundant RAID device.
- The file system used to store MRA files must be mirrored on at least two separate physical disks or be stored on a redundant RAID device.
- Separate disks must be used to store the SFS log volume versus the file system used to store MRA files.
- The latest (i.e., most current) MRA file must never be manipulated, moved, copied, or deleted since SFS may still be actively writing to it.
- MRA files (except the latest) must be copied to tape at least once per day.
- MRA files (except the latest) must be copied to at least two different tapes.

2.12.2 Rules for Backing Up SFS Data Volumes and TRB Files

- SFS data volumes must be mirrored on at least two separate physical disks or be stored on a redundant RAID device.

- The file system used to store TRB (i.e., data volume backup) files must be mirrored on at least two separate physical disks or be stored on a redundant RAID device.
- Separate disks must be used to store the SFS data volumes versus the file system used to store TRB files.
- Separate disks must be used to store the SFS log volume versus the SFS data volumes.
- A sufficient number of TRB files must be generated such that a complete backup of each SFS data volume is made at least every 5 days (recommend using a TRB file size such that it takes no more than 20 files to cover the entire data volume).
- TRB files must be copied to tape as soon as they are generated.
- TRB files must be copied to at least two different tapes.
- A complete set of TRB files must be generated (i.e., a complete SFS data volume backup) prior to running HPSS in production mode.
- A data dump (dd) of all SFS log and data volumes should be made at least once per month while HPSS and SFS are not running and should be copied to at least two different tapes.

2.12.3 Miscellaneous Rules for Backing Up HPSS Metadata

- Once a month, a site must tell SFS to truncate old backups and retain backup information for at least the latest 10 complete backups.
- At least one image of the tape backups must be moved offsite at a frequency that meets the site’s disaster recovery objectives. For example, if a site always wants to be able to recover to the previous day in case of a disaster, then tapes must be moved offsite each day.
- The Encina SFS configuration information must be backed up to at least two separate tapes after SFS is initially configured and immediately after the SFS configuration changes (i.e., a data volume is added, deleted, or expanded). It must also be backed up at least once per week.
- The DCE servers information (e.g., the security registry and CDS name space) must be backed up to at least two separate tapes immediately after HPSS is configured and then at least once per week.
- If configured, the HPSS PFTP configuration files must be backed up at least once per week.
- If configured, the HPSS NFS configuration files must be backed up at least once per week.
- HPSS must not be used to read / write tapes used to store backup files (Encina, DCE, etc.).

Chapter 3 System Preparation

This section will cover the steps that must be taken to appropriately prepare your system for installation and configuration of HPSS and its infrastructure.

3.1 General

- Each HPSS administrator should request a login id and password to the IBM HPSS web site at `http://www4.clearlake.ibm.com/hpss/support.jsp`.

- Download a copy of the HPSS Installation and Management Guides for the version of HPSS being installed. It will be useful to print a copy and keep it handy while deploying HPSS.

- Install, configure, and verify the correct prerequisite operating system version on all HPSS, DCE server, and SFS nodes.

  To verify the current operating system level on AIX:

  ```
  % oslevel
  ```

  To verify the current operating system level on non-AIX platforms:

  ```
  % uname -a
  ```

- Create appropriate HPSS Unix user accounts. The **hpss** user ID and **hpss** group ID should be created at this time.

- Install the prerequisite software for DCE, Encina, Sammi, C compiler, Java, Perl, ssh, and any other specific software that will be used by HPSS. Verify the correct patch levels are / will be installed. Refer to Section 2.3: *Prerequisite Software Considerations* on page 46 for additional information.

- Configure the PERL prerequisite software on any HPSS node and, if planning to interface with DFS, on any node that may be used for compiling the HPSS HDM server.

- Configure the SSH prerequisite software on the core HPSS server node (at a minimum) and configure SSH to accept connections from IBM Houston. Include the Houston subnet IP address **192.94.47** into the firewall routing rules of the site, as necessary.

- Download and install the HPSS deployment tool package (**http://www4.clearlake.ibm.com/hpss/support/ToolsRepository/deploy.jsp**) on each HPSS node in **/opt/hpss/tools/deploy**. Run and become familiar with the **lsnode** tool, which will be helpful in other steps.

  To run **lsnode** and save the output to **/var/hpss/stats/lsnode.out**:

  ```
  % mkdir -p /var/hpss/stats
  % cd /opt/hpss/tools/deploy/bin
  % lsnode > /var/hpss/stats/lsnode.out
  ```

3.2 Setup Filesystems

3.2.1 DCE

Note: The /var/dce filesystem, whether on the DCE server machine or client, must have free space available for DCE to work properly. /var/temp can fill up due to activity like editing a large file. When this happens, DCE is unable to create temp files for credentials and DCE processes will fail. Therefore, /var/dce should be made separate so that it will be unaffected by other activity requiring the use of /var. It is also highly recommended that general filesystem monitoring tools be installed on all DCE/HPSS/SFS machines to alert administrators of full filesystems so appropriate action can be taken immediately. The /var/hpss directory should also be a separate filesystem to protect it from being affected by the same problem or it in turn affecting /var or /var/dce adversely.

- Configure **/var/dce** as a separate file system on each HPSS server, DCE server, and SFS node with at least 20 MB of space initially and make sure this file system is automatically mounted at system reboot. On DCE server nodes, **/var/dce** should be mirrored across separate physical disks.

  To verify that the correct IP addresses (i.e., networks) are being used, on each DCE node:

  ```
  % rpccp show mapping | grep string | awk -F: '{print $2}' | \
    awk -F[ '{print $1}' | sort -u
  ```

- Verify that an appropriate time-synchronization mechanism is in place for all nodes in the DCE cell.

- Add a cron job (as root) on each DCE node to periodically clean up expired credentials.
  ```
  % crontab -e
  ```

  and add the following line:

  ```
  15 0,6,12,18 * * * /bin/rmxcred > /dev/null 2> /dev/null
  ```

- Verify on all DCE nodes that DCE is automatically started in **/etc/inittab** after system reboot.

- Obtain your HPSS Cell Id from IBM. This information will be needed when **mkhpss** is used to configure HPSS with DCE.

3.2.2 Encina

Configure **/opt/encinalocal** and **/opt/encinamirror** such that the contents are either mirrored or each of these two directories is stored on separate disks. Make sure these file systems are automatically mounted at system reboot.

3.2.3 HPSS

Configure **/var/hpss** as a separate file system on each HPSS server node while considering the following:

- Any node needs a minimum of 30 MB
- The node that will run the HPSS Log Daemon will need an additional 100 MB
- Any node running an HPSS NFS Daemon will need an additional 200 MB (or more, depending on the desired size of the NFS cache) if not using separate / dedicated disks for NFS
- Make sure this file system is automatically mounted at system reboot.

3.3 Planning for Encina SFS Servers

Plan the SFS configuration in terms of number of data volumes to use, which metadata files will be allocated to each data volume, the size of each data volume, the disks to be used, and mirroring options:

- Use the metadata sizing spreadsheet **mdsizing.xls** to determine final amount of SFS data volume space to configure. When completed, this spreadsheet should be added to the site configuration information and not lost. (To obtain the metadata sizing spreadsheet, contact your customer support representative.)
- Use multiple data volumes as opposed to one large volume
- Keep in mind that in the future as the size of HPSS grows, it may be necessary to move certain SFS files or data volumes to different disks and / or nodes. The movement of data volumes is easy because standard AIX LVM tools can be used. Splitting data volumes (i.e., moving an individual SFS file to another data volume) is much more costly because the SFS file must be exported and re-imported, which is a time-consuming task. The SFS bulk copy feature can be used to improve the performance for these types of operations for TX Series 4.3 and later.
- Make sure the larger SFS files are on different data volumes (e.g., tape storage segments, NS objects, BFS bitfiles, etc.)
- An initial starting point for allocating SFS files to data volumes is as follows:
    - ◆ Name Server metadata files
    - ◆ Bitfile Server metadata files

    - ◆ Disk Storage Server metadata files (separate volume per disk storage server)
    - ◆ Tape Storage Server metadata files (separate volume per tape storage server)
    - ◆ All others

- As a common rule, target having 5-8 data volumes
- The SFS log volume must be mirrored on at least two(2) separate physical disks or be stored on a redundant RAID device
- Each SFS data volume must be mirrored on at least two(2) separate physical disks or be stored on a redundant RAID device
- Separate disks should be used to store the SFS log volume versus the SFS data volumes

3.3.1 AIX

The recommended way of allocating data volumes across disks using AIX is to pool all SFS data volume disks into one volume group and, when creating the data volumes as raw AIX logical volumes, set the **Maximum Number of Physical Volumes to use for allocation** field in SMIT to the total number of disks in the volume group and set the **Number of Copies of each logical partition** to 2.

This causes each data volume to be spread out evenly across all disks, while keeping the mirrored image of each partition on a different physical disk from the primary copy.

For example, suppose we have 8 physical disks to be used for SFS data volumes. We create a volume group called sfsvg and put all 8 disks in this VG. When creating a logical volume to be used as a data volume, we specify sfsvg as the volume group, raw as the Logical volume type, 8 as the Maximum Number of Physical Volumes to use for allocation, and 2 as the Number of Copies of each logical partition. The first part of this disk device will be on disk #1 with its mirrored image on disk #2. The next part of the device will be on disk #2 with its mirrored image on disk #3, and so on.

The result of this approach is a nice distribution of the metadata files and associated access activity evenly spread across all SFS disks. The load distribution across multiple disk heads yields better overall performance.

This **mkhpss** utility is used to configure Encina SFS servers. The utility will prompt the administrator for logical volume names to use for SFS log and data volumes. If these logical volumes do not already exist, **mkhpss** will prompt you for the information necessary to create them (e.g. volume group name, number of partitions to use, number of copies) and do the creation for you.

3.3.2 Solaris

The **mkhpss** utility will prompt for a disk name to be used to create a logical volume for the SFS server. It then initializes the disk, creates a physical volume, and creates a logical volume.

mkhpss then prompts for whether mirroring is desired for this logical volume. If so, it will prompt for the name of the disk to use as the mirror.

3.4 Setup for HPSS Metadata Backup

- Install and configure the automated SFS backup toolset. Keep in mind the following:
  - ◆ Media archiving must always be enabled while HPSS is in production mode. If it is ever temporarily disabled (e.g., during a one-time Unitree migration), then before HPSS is placed back into production mode, media archiving must be re-enabled and a complete backup of all SFS must be made.
  - ◆ The file systems used to store TRB and MRA files must be mirrored on at least two(2) separate physical disks or stored on redundant RAID devices.
  - ◆ The latest (i.e., most current) MRA file must never be manipulated, moved, copied, or deleted since SFS may still be actively writing to it.
  - ◆ MRA files must be copied from disk to at least two(2) different tapes at least once per day.
  - ◆ A sufficient number of TRB files must be generated such that a complete backup of each SFS data volume is made at least every 5 days.
  - ◆ TRB files must be copied to at least two(2) different tapes.
  - ◆ At least one image of the tape backups should be moved offsite at a frequency that meets the site's disaster recovery objectives.

3.5 Setup Tape Libraries and Drives

3.5.1 IBM 3584

If using an IBM 3584 tape library, install the Atape driver, configure the SCSI Medium Changer (SMC) library device on the node that will run the HPSS LTO PVR, and verify that you can talk to the library.

To install the Atape driver use smitty in AIX:

```
% smitty install
```

To configure the library:

```
root% cfgmgr
root% lsdev -Cc tape | grep smc
smc0 Available 40-58-00-0,1 IBM 3584 Library Medium Changer
```

To test communications with the library:

```
% tapeutil -f <smc device filename> inventory
```

To test tape mounts:

```
% tapeutil -f <smc device filename> move <Source Slot> <Drive Slot>
```

To test tape dismounts:

```
% tapeutil -f <drive device filename> unload
% tapeutil -f <smc device filename> move <Drive Slot> <Source Slot>
```

Run this before HPSS has started since only one process can have an open smc device file descriptor.

For drives in an IBM 3584 robot, identify the robot-specific device id (LTO drive locations) for each Mover tape device. This will be required when configuring the tape drives within HPSS.

To identify robot-specific device ids for each Mover tape device in an LTO robot:

```
% tapeutil -f <smc device filename> inventory
```

Look for addresses of drive devices in the output.

Refer to Section 6.8.13.2: *LTO PVR Information* on page 387 for more information.

3.5.2 3494

For a 3494 tape library:

- Identify the robot-specific device id for each Mover tape device. This will be required when configuring the tape drives within HPSS.

  To identify robot-specific device id for each Mover tape device in a 3494 robot:

  ```
  % /opt/hpss/bin/GetESANumbers <device>
  ```

- Identify range(s) of tape cartridge labels to be used by HPSS.

- Configure the **lmcp** daemon on the node that will run the HPSS 3494 PVR, verify that it is working properly, and configure the lmcp daemon to be started automatically during system reboot.

  To see if **lmcp** daemon is running:

  ```
  % ps -e | grep lmcpd
  ```

  To start the **lmcp** daemon:

  ```
  root% /etc/methods/startatl
  ```

  To test whether **lmcp** daemon is configured and working correctly:

```
% mtlib -l<lmcpDevice> -qL
```

where **lmcpDevice** is usually **/dev/lmcp0**.

To test ability to use **lmcp** daemon to mount a tape:

```
% mtlib -l/dev/lmcp0 -m -V<tapeLabel> -x<deviceNumber>
```

Test ability to dismount the tape:

```
% mtlib -l/dev/lmcp0 -d -V<tapeLabel> -x<deviceNumber>
```

To automatically start the **lmcp** daemon after system reboot, add **/etc/methods/startatl** to the **/etc/inittab** file

Refer to Section 6.8.13.3: *IBM 3494/3495 PVR Information* on page 388 for more information.

3.5.3 STK

For an STK tape library:

- If using an STK tape library, configure the ACSLS and SSI software properly and verify that it is working correctly.

  To test the ability to mount and dismount a tape in an STK library, use the **stk_ctl** utility that is provided by the HPSS automated SFS backup toolset.

  To mount a tape:

  ```
  % stk_ctl mount <driveSpec> <tapeLabel>
  ```

  where **driveSpec** is four integers separated by commas (no spaces), identifying the ACS, LSM, panel, and drive (e.g., **`0,0,1,2`**).

  To dismount a tape:

  ```
  % stk_ctl dismount <driveSpec>
  ```

  To query a drive:

  ```
  % stk_ctl query <driveSpec>
  ```

  Refer to Section 6.8.13.4: *StorageTek PVR and RAIT PVR Information* on page 389 for more information.

3.5.4 AML

For AML tape libraries:

- If using an AML PVR, configure the Insert/Eject ports using the configuration files **/var/hpss/etc/AML_EjectPort.conf** and **/var/hpss/etc/AML_InsertPort.conf**.

Refer to Section 6.8.13.5: *ADIC Automatic Media Library Storage Systems Information* on page 392 for more information.

3.5.5 Tape Drive Verification

Verify that the correct number and type of tape devices are available on each Tape Mover node.

3.5.5.1 AIX

The tape devices section of the **lsnode** report displays all available tape drives.

To manually find out the type and number of available tape devices:

```
% lsdev -C -S a -c tape
```

- On each Tape Mover node, verify that each tape drive has variable-length block size set.

  The tape devices section of the **lsnode** report indicates whether the available tape drives have variable-length block size set.

  To manually check whether variable block size is enabled, the following should return a value of zero(**0**):

  ```
  % lsattr -E -l <tapeDevice> -a block_size -F value
  ```

  To change the device to use variable block size:

  ```
  root% chdev -l <tapeDevice> -a block_size=0
  ```

- If using STK drives (e.g., Redwoods), verify that the drive type is not incorrectly set to Generic tape drive or IBM Emulation mode.

- On each Tape Mover node, verify that the raw read and write I/O performance of all HPSS tape drives are at expected levels. Create one or more tables documenting the results.

  To measure uncompressed write performance (see warning below) on **rmt1** (Note that specifying **rmt1.1** causes the tape not to rewind):

  ```
  % iocheck -w -t 20 -b 1mb /dev/rmt1.1
  ```

  To measure the maximum-compressed write performance on **rmt1** (and then rewind the tape):

  ```
  % iocheck -w -t 20 -f 0 -b 1mb /dev/rmt1
  ```

  To measure read performance on drive **rmt1** using the previously-written uncompressed and compressed files:

```
% iocheck -r -t 20 -b 1mb /dev/rmt1.1
% iocheck -r -t 20 -b 1mb /dev/rmt1.1
```

WARNING: The contents of this tape will be overwritten so be sure to mount the correct tape cartridge.

To unload a tape:

```
% tctl -f <device> rewoffl
```

Repeat the above steps for each tape drive.

3.5.5.2 Solaris

On each Tape Mover node, verify that each tape drive has variable-length block size set.

To manually check whether variable block size is enabled (see warning below), the following should complete successfully:

```
% dd if=/dev/null of=/dev/rmt/0n bs=80 count=1
% dd if=/dev/null of=/dev/rmt/0n bs=1024 count=1
```

If variable-length block sizes are not enabled, consult your driver documentation for procedures to enable it.

On each Tape Mover node, verify that the raw read and write I/O performance of all HPSS tape drives are at the expected levels. Create one or more talbes documenting the results.

To measure uncompressed write performance (see warning below) on 0 (Note that specifying 0n will cause the tape not to rewind):

```
% iocheck -w -t 20 -b 1mb /dev/rmt/0n
```

To measure the maximum-compressed write performance on 0 (and then rewind the tape):

```
% iocheck -w -t 20 -f 0 -b 1mb /dev/0
```

To measure read performance on drive 0 using the previously-written uncompressed and compressed files:

```
% iocheck -r -t 20 -b 1mb /dev/0n
% iocheck -r -t 20 -b 1mb /dev/0n
```

To empty the tape:

```
% mt -f /dev/0n rewind
% mt -f /dev/0n weof 2
```

WARNING: The contents of this tape will be overwritten so be sure to mount the correct tape cartridge.

3.5.5.3 IRIX

On each Tape Mover node, verify that each tape drive has variable-length block size set.

To manually check whether variable block size is enabled (see warning below), the following should complete successfully:

```
% dd if=/dev/null of=/dev/rmt/tps2d6nr bs=80 count=1
% dd if=/dev/null of=/dev/rmt/tps2d6nr bs=1024 count=1
```

If variable-length block sizes are not enabled, consult your driver documentation for procedures to enable it.

On each Tape Mover node, verify that the raw read and write I/O performance of all HPSS tape drives are at the expected levels. Create one or more talbes documenting the results.

To measure uncompressed write performance (see warning below) on tps2d6 (Note that specifying tps2d6nr will cause the tape not to rewind):

```
% iocheck -w -t 20 -b 1mb /dev/rmt/tps2d6nr
```

To measure the maximum-compressed write performance on tps2d6 (and then rewind the tape):

```
% iocheck -w -t 20 -f 0 -b 1mb /dev/tps2d6nrc
```

To measure read performance on drive tps2d6 using the previously-written uncompressed and compressed files:

```
% iocheck -r -t 20 -b 1mb /dev/tps2d6nr
% iocheck -r -t 20 -b 1mb /dev/tps2d6nr
```

To empty the tape:

```
% mt -f /dev/tps2d6 rewind
% mt -f /dev/tps2d6 weof 2
```

WARNING: The contents of this tape will be overwritten so be sure to mount the correct tape cartridge.

3.5.5.4 Linux

On each Tape Mover node, verify that the raw read and write I/O performance of all HPSS tape drives are at the expected levels. Create one or more talbes documenting the results.

To measure uncompressed write performance (see warning below) on st1 (Note that specifying nst1 will cause the tape not to rewind):

```
% iocheck -w -t 20 -b 1mb /dev/rmt/nst1
```

To measure the maximum-compressed write performance on st1 (and then rewind the tape):

```
% iocheck -w -t 20 -f 0 -b 1mb /dev/nst1
```

To measure read performance on drive st1 using the previously-written uncompressed and compressed files:

```
% iocheck -r -t 20 -b 1mb /dev/nst1
% iocheck -r -t 20 -b 1mb /dev/nst1
```

To empty the tape:

```
% mt -f /dev/st1 rewind
% mt -f /dev/st1 weof 2
```

WARNING: The contents of this tape will be overwritten so be sure to mount the correct tape cartridge.

3.6 Setup Disk Drives

Verify that Ultra SCSI is enabled for all SCSI devices via smit on AIX and by the appropriate means on non-AIX platforms.

3.6.1 AIX

- Verify that the correct number and type of disk devices are available on each SFS and Disk Mover node.

  The disk devices section of the **lsnode** report displays all available disk devices.

  To manually find out the type and number of available disk devices:

  ```
  % lsdev -C -S a -c disk
  ```

- If using SSA disks, on each appropriate Disk Mover spread the SSA disks equally across the two loops on each SSA adapter.

  The SSA configuration section of the **lsnode** report provides details on the SSA configuration.

  Note the following:

  ◆ The section shows which SSA adapter to which each disk is attached.

- ◆ There are two loops (a and b) per adapter and two ports per loop (a1, a2, b1, b2)
- ◆ The physical order of the disks are shown from the perspective of each port
- ◆ A disk is accessed according to its closest port (e.g., either a1 or a2, b1 or b2)
- ◆ When planning to configure striped SSA disks in HPSS, it is important to select disks for each striped virtual volume that span ports, loops, and/or adapters. These decisions can be made after determining the probable bottlenecks and then selecting individual disks in a virtual volume to alleviate bottlenecks in the port, loop, or adapter, as desired.

For SSA disks on an AIX SP node, use **maymap** to identify which loop the SSA disk is on.

- Create volume groups for all disks to be used by HPSS.
- Create all necessary raw disk logical volumes to be used by the HPSS Disk Mover(s).

  To create a volume group for a physical disk, use SMIT or the following:

  ```
  % mkvg -f -y<volumeGroup> -s<partitionSize> <physicalDisk>
  ```

  To create a logical volume, use SMIT or the following:

  ```
  % mklv -y<logicalVolume> -traw <volumeGroup> <numPartitions>
  ```

  Note that there are additional options for specifying exactly where on the physical disk the logical volume should be placed, if that is considered important.

  To view all physical disks and associated volume groups:

  ```
  % lspv
  ```

  To view all logical volumes on a given volume group:

  ```
  % lsvg -l <volumeGroup>
  ```

- On each Disk Mover node, measure the raw read and write I/O performance of all HPSS disks and verify that they are at expected levels. Create one or more tables documenting the results. An example table can be found above. The output of these test should be stored in **/var/hpss/stats** for later analysis.

  Use the **iocheck.ksh** script from the deployment tools package to show the performance of one or more individual disk devices as well as show the peak aggregate performance of concurrent I/O across multiple disks (e.g., to show the peak performance of adapters).

  To measure the individual and aggregate throughput of hdisks 4, 5, 6, and 7:

  ```
  % iocheck.ksh 4 5 6 7
  ```

  To measure read performance on a single disk:

  ```
  % iocheck -r -t 20 -b 1mb /dev/r<logicalVolume>
  ```

where logicalVolume is a raw logical volume that is sized to provide at least 20 seconds of I/O throughput.

To measure write performance on a single disk (see warning below):

```
% iocheck -w -t 20 -b 1mb -o 1mb /dev/r<logicalVolume>
```

where logicalVolume is a raw logical volume that is sized to provide at least 20 seconds of I/O throughput.

WARNING: The contents of this logical volume will be overwritten so be sure to use the correct logical volume name.

3.6.2 Linux

For Linux, verify that there is a raw device mapping for each block device to be accessed by HPSS. To list all current device mappings:

```
% raw -a -q
  /dev/raw/raw1:  bound to major 8, minor 1
  /dev/raw/raw2:  bound to major 8, minor 2
```

Block devices can be mapped to raw devices at boot time by adding mapping information to the **/etc/sysconfig/rawdevices** file. To map the first partition of SCSI disk **a** to raw device **raw1** and the second partition of SCSI disk **b** to raw device **raw2**, add the following lines:

```
/dev/raw/raw1 /dev/sda1
/dev/raw/raw2 /dev/sdb2
```

On each Disk Mover node, measure the raw disk read and write I/O performance of all HPSS disks and verify that they are at expected levels.

To measure read performance on a single disk:

```
% iocheck -r -t 20 -b 1mb /dev/raw/rawX
```

Where X is the number of the raw device that is sized to provide at least 20 seconds of I/O throughput.

To measure write performance on a single disk (see warning below):

```
% iocheck -w -t 20 -b 1mb -o 1mb /dev/raw/rawX
```

Where X is the number of the raw device that is sized to provide at least 20 seconds of I/O throughput.

WARNING: The contents of this disk will be overwritten so be sure to use the correct disk name.

3.6.3 Solaris & IRIX

For Solaris & IRIX platforms, specific commands and syntax are not listed. Perform the following steps using the appropriate commands for the OS used:

- Verify that the correct number and type of disk devices are available on each SFS and Disk Mover node.
- Create all necessary raw disk volumes to be used by the HPSS Disk Mover(s).
- On each Disk Mover node, measure the raw read and write I/O performance of all HPSS disks and verify that they are at expected levels. Create one or more tables documenting the results. The output of these test should be stored in **/var/hpss/stats** for later analysis.

3.7 Setup Network Parameters

- Install and configure all network interfaces and corresponding network connections.

  Refer to IBM's internal network technologies home page for resources on configuring and tuning networks and TCP/IP.

  The network interfaces section of the **lsnode** report from each node shows the network interfaces that are configured. See also the network options section

  To manually find out how many network interfaces are available:

  ```
  % lsdev -C -S a -c if
  % netstat -i
  ```

  To manually determine if an SP switch is available:

  ```
  % lsdev -C -S a -l css0
  % netstat -i
  ```

  To view all IP addresses associated with a local host (i.e., for all network interfaces): *Note: Does not work for SP switch interface.*

  ```
  % for x in `lsdev -C -S a -c if -F name`; do
    lsattr -E -a netaddr -l $x -F value
  done
  ```

  To view additional information for each network interface:

  ```
  % for x in `lsdev -C -S a -c if -F name`; do
    ifconfig $x
  done
  ```

  For non-AIX platforms, the use the Name column from **netstat -i** in the **ifconfig** commands. Note the IP address follows the inet phrase in the output from the **ifconfig** command.

To test whether an IP address is reachable (non-zero exit status indicates the **ping** was not successful):

```
% ping -c 1 <ipAddress>
```

- Determine which networks will be used for control vs. data paths. DCE should not use all available networks on a multi-homed system unless each of those networks is guaranteed to have connectivity to other DCE services. If a particular network is removed (physically, or routing is changed), that connection remains in DCE's RPC mappings. DCE will continue to try and use that network despite the loss of connectivity. The timeouts and retries can have an adverse affect on HPSS as well as other applications relying on DCE.

  ◆ Isolate DCE communication to the designated control network(s) on all HPSS, DCE, and SFS nodes in order to separate the HPSS control and data paths.

  ◆ As desired, tell DCE to ignore certain network interfaces in order to separate the HPSS control and data paths. For example, to ignore FDDI (**fd0**) and HIPPI (**hp0**), add the following line to **/etc/environment**:

```
RPC_UNSUPPORTED_NETIFS=fd0:hp0
```

- Place all HPSS, SFS, and DCE server machine IP addresses in a local host table (**/etc/hosts**). For AIX, configure the machine to use the table as backup in the event of a DNS failure. The file **/etc/netsvc.conf** should be modified to look like the following:

```
hosts=bind,local
```

  The **netsvc.conf** file is used to specify the ordering of host name resolution. In the above ordering, DNS will be used first, if the host name is not found, then the local **/etc/host** file will be used.

4. For each ethernet network interface, verify that both the **en0** and **et0** interfaces are not configured at the same time (we recommend only using **en0** unless the other machines in the network are all using the **802.3 et*** interface). Configure the local name service with the unique hostname for each network interface on all nodes and verify that each hostname is resolvable from other nodes.

To understand and optimize the operation of HPSS, some baseline measurement, evaluation, and tuning of the underlying network and IO subsystems is necessary. Appropriate tuning of these components is necessary for HPSS to perform as expected. Any one component that is not performing at its optimal rate will have an adverse affect on the performance of the entire system. The steps and tools listed here will help a site make the best use of their available resources. The measurements taken should be saved for future reference. If performance problems occur later on, these values can be compared with new measurements to determine if the performance problems are related to changes in the subsystems. Though disk and tape configurations rarely change without an administrator's knowledge, networks can "mysteriously" down grade for a variety of reasons. This is especially true for client access to the HPSS system.

- Verify that network TCP throughput has been optimized and the performance of each network is at expected levels in both directions (especially check HPSS data networks between Movers and between Mover and client nodes). Using ttcp or another network tool, measure the performance of all the networks that will be used in communications between DCE, HPSS, SFS, and client machines. If multiple paths between machines are to be used, then all of them need to be measured as well. The transfer rates for networks differ greatly depending upon the individual technology and network settings used. It is important to gather performance data using a variety of settings to determine the optimal combinations. The primary values that govern performance include send / receive buffers, size of reads / writes, and **rfc1323** value for high performance networks (HIPPI, G-Enet). Create a table showing these values. An example table can be found below:

To test the receiving machines performance issue:

```
% ttcp -r -s -b<bsize> -l<lsize>
```

To test the sending machines performance issue:

```
% ttcp -t -s -b<bsize> -l<lsize> <hostname/interfacename>
```

The following is an example of the information to be gathered from the above commands (assumes **en0** is 10 MB Ethernet, if **en0** is Fast or G-Enet, **rfc1323** should be on:

```
Receiver Table
Interface      Bsize Lsize RFC1323     Performance CPU utilization
en0            16k   16k   Off         --          --
en0            16k   8k    Off         --          --
en0            64k   64k   Off         --          --
en0            64k   32k   Off         --          --
...
ccs0           64k   64k   On          --          --
ccs0           64k   32k   On          --          --
...
Sender Table
Interface      Bsize Lsize RFC1323     Performance CPU utilization
en0            16k   16k   Off         --          --
en0            16k   8k    Off         --          --
en0            64k   64k   Off         --          --
en0            64k   32k   Off         --          --
...
ccs0           64k   64k   On          --          --
ccs0           64k   32k   On          --          --
...
```

What you are looking for are the best values possible for each network connection. These values will be used in turn by HPSS to optimize its data transfers. By no means is this a complete picture of what controls network performance. In fact, the assumption is made that the customer already has the networks optimized by their local network group or has contacted outside assistance to perform this task. The process described here is to determine the best user-level values to optimize HPSS performance on an already tuned network, rather then trying to fix underlying network problems.

To test the TCP socket performance over a network connection, issue the following on the receiving node:

```
% ttcp -r -s -p<port>
```

where a typical port is 4321. Then issue the following on the transmitting node:

```
% ttcp -t -s -p<port> <hostname>
```

Note that the ttcp tool is included in the deployment package and is not related to the Unix ToolTalk service.

HPSS makes extensive use of a system's networking capabilities. Therefore, the setting of the tunable networking parameters for the systems on which the various HPSS servers and clients will run can have a significant impact on overall system performance.

Under AIX, a utility is provided to display and modify a number of networking parameters. The utility is **_no_** (Network Options). Refer to the *AIX Performance Tuning Guide,* for details of each option and its impact on networking and system performance.

Some options that typically impact performance within an HPSS system environment are:

Table 3-1 Network Options

Network Option	Description
thewall	Controls the maximum amount of system memory that can be used by the system networking code. A value that is too low can cause networking requests to be delayed or denied. The recommendation from AIX development is to set this value to at least two times the maximum number of concurrent connections times the size of the socket send/receive buffers. The default setting for AIX 4.3.2 and later is the smaller of (1) half the size of physical memory or (2) 1 GB.
sb_max	Controls the maximum size allowed for send and receive buffers for a socket.
udp_recvspace	Controls the default size of the receive buffer for UPD/IP sockets. A value that is too small can cause server RPC sockets to be overrun.
tcp_recvspace, tcp_sendspace	Controls the default size for the receive and send buffers for TCP/IP sockets. Internally, HPSS servers and clients attempt to set these buffers sizes explicitly, but other utilities may not.
rfc1323	Controls whether large TCP window sizes are used. Usually set to ON for higher throughput networks (e.g., HiPPI, SP switch) and set to OFF for lower throughput networks (e.g., ethernet, FDDI).

AIX also provides a configuration attribute that controls the maximum amount of memory that can be allocated to **mbufs**. It can be viewed or modified via **smit** (select "**Process Environments**", then "**Change / Show Characteristics of Operating System**") or via the command line ("**lsattr -E -l sys0**", "**chdev -e sys0 -a maxmbuf = <new value>**").

It is recommended that the available combination of options be tested as part of the initial HPSS system testing. In particular, poor network performance has been experienced where options on one system do not match options on other remote systems.

There are also attributes that are specific to the individual network interface that may affect network performance. For example, the network interface for the IBM SP TB3 switch provides settings for the size of the send and receive pool buffer size, which have had an effect on throughput. It is recommended that the available interface specific documentation be referenced for more detailed information.

The anticipated load should also be taken into account when determining the appropriate network option settings. Options that provide optimal performance for one or a small number of transfers may not be the best settings for the final multi-user workload.

3.7.1 HPSS.conf Configuration File

The **HPSS.conf** configuration file contains tuning options to be used by HPSS clients and servers.

The **HPSS.conf** file may also contain options used by non-HPSS applications. Application developers are asked to please observe the “Reserved for Future Use” components specified in Section 3.7.7.

The **HPSS.conf** configuration file is located in the directory named by either:

- The **HPSS_PATH_ETC** environment variable,
- the directory **/var/hpss/etc**, or
- the directory **/usr/local/etc**

in that order (except for when accessed by the Authentication Managers **- Realms to DCE Cell Mappings** stanza). If the file is not present or no matching entry is found, the Parallel FTP Client, Client API, and Mover will use system defaults.

General HPSS.conf Rules:

- Lines are comprised of *comments*, *blank lines*, *simple specifiers*, *specifiers* and *values* - “abc = def”, or *compound specifiers* and *terminators* - “def = { …}.”
- Only four levels of specification are allowed: Stanzas, SubStanzas, Sections, and SubSections.
- SubStanzas may only exist in Compound Stanzas. Sections may only exist in Compound SubStanzas, and SubSections may only exist in Compound Sections.
- No line may exceed 128 characters, including the “= {“.
- Comments may be included by entering either a semicolon (“;“) or a pound sign (“#”) as the first non-white character in a line. Use of either of these characters other than as the first character will not be interpreted as a comment (It will be interpreted as part of the specifier or value!).
- Indentation is optional but is encouraged (assists in diagnosis).
- Closing braces (“}“) must be used to terminate opening braces (“{“).
- Spaces before or after the equal sign (“=”) are optional.

- Blank Lines are ignored.

NOTE: HPSS and Network Tuning are highly dependent on the application environment. The values specified herein are NOT expected to be applicable to any installation!

3.7.1.1 PFTP Client Stanza

The Parallel FTP Client configuration options are in two distinct stanzas of the **HPSS.conf** file (Section 3.7.1.1: *PFTP Client Stanza* on page 163, and Section 3.7.1.2: *PFTP Client Interfaces Stanza* on page 165).

Table 3-2 PFTP Client Stanza Fields

Configuration Type	Abbreviated Description
Stanza (Compound)	**`PFTP Client = {`** *Reserved* Stanza specifier. Must be terminated with a matching "}"
SubStanza	**`Default COS = <value>`** E.g. **`Default COS = 99`** *Optional* SubStanza specifying the default Class of Service if not explicitly specified by the user. Use with caution – standard procedure is to allow the Bitfile Server to determine the optimal COS!
SubStanza	**`Protocol = <value>`** E.g. **`Protocol = PDATA_ONLY`** *Optional* SubStanza specifying the default protocol. May contain either of the two protocols supported, **PDATA_AND_MOVER** or **PDATA_ONLY.** The default is **PDATA_AND_MOVER.**
SubStanza	**`Auto Parallel Size = <value>`** E.g. **`Auto Parallel Size = 4MB`** *Optional* SubStanza specifying the minimum file size to start using the "auto-parallel" features of the PFTP Client. May be specified as a decimal number or "*x*MB" style notation.
SubStanza	**`PortRange = <value>`** E.g. **`PortRange = ncadg_ip_tcp[10100-12100]`** *Optional* SubStanza specifying the TCP port range to use between the PFTP Client and the Mover(s). This may be necessary when Port Range Filters are used for Security.

Table 3-2 PFTP Client Stanza Fields

SubStanza	**`Transfer Buffer Size = <value>`** E.g. **`Transfer Buffer Size = 16MB`** *Optional* SubStanza specifying the PFTP Buffer sizes. May be specified as a decimal number or “*x*MB” style notation.
SubStanza	**`Socket Buffer Size = <value>`** E.g. **`Socket Buffer Size = 16MB`** *Optional* SubStanza specifying the Pdata Socket Buffer sizes. May be specified as a decimal number or “*x*MB” style notation.
SubStanza	**`MAX Ptran Size = <value>`** E.g. **`MAX Ptran Size = 4GB`** *Optional* SubStanza specifying a larger transfer size between socket open and closure. For disk COSs, the segment sizes may potentially override this specification! May be specified as a decimal number or “*x*MB” style notation.

Note: All PFTP Client SubStanzas are optional.

The **PFTP Client = { ... }** stanza contains several optional specifications for the **pftp_client** and/or **krb5_gss_pftp_client** executables.

The **DEFAULT COS = value** substanza assigns a default Class of Service (COS) for HPSS. If this option is not specified, the Bitfile Server is responsible for determining the default COS unless the user explicitly issues a “**`site setcos <ID>`**” command or specifies “**`-C<ID>`**“ on the command line to change the class of service. If this substanza is specified, the class of service provided in **value** will be used unless the user explicitly issues a “**`site setcos <ID>`**” command to change the class of service. For this reason, caution should be used in specifying this option.

The **Protocol = value** substanza is used to specify the desired PFTP protocol. Currently, either of two values may be specified: **PDATA_AND_MOVER** or **PDATA_ONLY**. The default specification is **PDATA_AND_MOVER**. The **PDATA_ONLY** specification provides improved performance in high latency WAN environments.

PFTP processes perform automatic conversion of **get** and **put** commands to their parallel equivalents, **pget** and **pput**. Some sites have reported degraded performance as a result of this substitution occurring with small file transfers. To accommodate this problem, the **Auto Parallel Size = value** substanza may be specified in the **HPSS.conf** file where the “automatic” parallel features will not be invoked if the size of the file is less than the value provided. The value may be specified as a decimal number (1048576) or in the format: *x*MB.

For sites with “Firewalls/ Diodes” installed, it may be necessary to provide specific ports for the data to move between the PFTP Client and the Mover(s). This can be accomplished by specifying the **PortRange = value** substanza. The syntax of this statement is:

```
PortRange = ncadg_ip_udp[10100-12100]:ncadg_ip_tcp[10100-12100]
```

This syntax is identical to DCE's **RPC_RESTRICTED_PORTS** environment variable. Only the **ncacn_ip_tcp[start_port-end_port]** (*TCP* component) is used so the **ncadg_ip_udp** component may be omitted.

Additional options are available for controlling the size of the PFTP transfer buffers, **Transfer Buffer Size**, and the buffer size for the sockets in the **PDATA_ONLY** protocol, **Socket Buffer Size**. The value may be specified as a decimal number (1048576) or in the format: *x*MB.

The PFTP parallel protocol opens and closes the sockets between the PFTP Client child processes and the Mover(s). The default value for tape was every 512 MB and for disk was the smaller of the size of 64 storage segments or 512 MB. With transfers increasing in performance into the 100MB/sec and greater range, the opening and closing of sockets is another performance problem. The **MAX Ptran Size = value** substanza has been provided to allow for larger transfers between socket open and closing.

NOTE: in the case of disks, the 64 storage segments is still the overriding specification, so Storage Classes need careful specification to provide very large segments if the value associated with **MAX Ptran Size** is large. An artificial limit of 250 GB is compiled into the PFTP Client, which should not cause a great concern any time in the near future. Even at 1 GB/sec, this is several minutes! The value may be specified as a decimal number (4294967296) or in the format: *x*GB.

PFTP Client Stanza Example:

```
PFTP Client = {
    ;    Set Default Class of Service
    DEFAULT COS = 2
    ;    Set Default Protocol
    Protocol = PDATA_AND_MOVER
    ;    Set Minimum File Size for auto mapping Non-parallel
    ;    commands to Parallel commands
    Auto Parallel Size = 4MB
    ;    Set the Port Range for Port Restrictions on Parallel Transfers
    ;    Only the port range for ncacn_ip_tcp is parsed.
    ;    The syntax has been taken from DCE.
    PortRange = ncadg_ip_udp[10100-12100]:ncadg_ip_tcp[10100-12100]
    ;    PDATA Options
    Transfer Buffer Size = 1MB
    Socket Buffer Size = 16MB
    ;    PFTP sets an Artificial (Compiled in) Maximum of 250GB
    MAX Ptran Size = 10GB
}
```

3.7.1.2 PFTP Client Interfaces Stanza

Many systems have multiple interfaces, some of which may not have access to all destination hosts. The PFTP Client Interfaces stanza is used to specify which interfaces on the source host should be used to communicate to destination Parallel FTP Daemons and/or HPSS Movers. This is particularly useful if both low speed and high speed interfaces are available to the client host and the PFTP data transfers should use the high speed interfaces.

Table 3-3 PFTP Client Interfaces Stanza Fields

Configuration Type	Abbreviated Description
Stanza (Compound)	**`PFTP Client Interfaces = {`** *Reserved* Stanza specifier. Must be terminated with a matching "}"
SubStanza (Compound)	**`<Hostname> <hostname.domain> = {`** E.g. **`my_host my_host.my.domain ={`** Contains the hostname(s) executing the PFTP Client. Must be terminated with a matching "}"
Section (Compound)	**`<PFTP_Daemon_host>`** **`<PFTP_Daemon_host.domain> = {`** E.g. **`storage storage.my.domain = {`** Contains the hostname(s) executing the PFTP Daemon. Must be terminated with a matching "}"
SubSection	**`<Dotted IP Address>`** E.g. **`100.101.103.103`** Contains the Dot Notation IP Address specification for the interface on the local host to use to connect to the Mover(s) associated with the PFTP Daemon.

The **PFTP Client Interfaces = { ... }** stanza contains several configuration options for the **pftp_client** and / or **krb5_gss_pftp_client** executables.

SubStanzas refer to the **hostname(s)** associated with the local system.

Sections refer to the **Parallel FTP Daemon hostname(s)**.

For HPSS, this is not necessarily the name of the Mover(s).

SubSections specify Dot Notation IP Addresses of the interfaces on the local host to be used. For HPSS, all of these interfaces must be able to connect to the Mover(s). NOTE: If and only if a specific COS is specified, these interfaces need only provide connection to the Mover(s) associated with the specific COS.

PFTP Client Interfaces Stanza Rules:

- Source hostnames may contain one or more hostnames separated by white spaces (subject to the 128 character limit).
- "Default" is a reserved notation to be used if the local host is not in the Stanza.

- Destination Host (FTPD Host) may contain one or more hostnames separated by white spaces (subject to the 128 character limit).
- Interface Specification must be IP Address Dot Notation.
- Interfaces *must* be able to connect to destination.

Communication failures that are not easily diagnosed will occur if the interface specification is invalid.

PFTP Client Interfaces Stanza Example:

```
PFTP Client Interfaces = {
    ; PFTP Client Host Name(s)
    water.clearlake.ibm.com water = {
    ; Next Specification is the PFTP Daemon Host
    ; water has 4 interfaces that can talk to the HPSS Movers
    ; associated with the PFTP Daemon Host "water"
    water.clearlake.ibm.com water = {
        192.94.47.227
        192.175.14.35
        192.222.197.1
        192.2.1.1
    }
    ; water has ONLY 1 interfaces that can talk to the HPSS Movers
    ; associated with the PFTP Daemon Host "sneezy"
    sneezy sneezy.clearlake.ibm.com = {
        192.94.47.227
    }
    ; Use the default water interface, 192.100.101.1, to talk to
    ; any other PFTP Daemons.
    Default = {
        192.100.101.1
    }
}

    sneezy sneezy.clearlake.ibm.com = {
        larry larry.clearlake.ibm.com = {
            192.94.47.226
        }
        sneezy sneezy.clearlake.ibm.com = {
            192.94.47.226
        }
    }

    ; For all other Client Hosts - This allows a single HPSS.conf file
    ; to be available using a common files
    ; system.  This is ONLY useful for cluster systems that specify
    ; "Common" interfaces for multiple
    ; nodes of the cluster (I/O Nodes)
```

```
        Default = {
            ; Client Host Name
            water water.clearlake.ibm.com = {
                134.253.14.227
            }
        }
    }
```

3.7.1.3 Multinode Table Stanza

The HPSS PFTP Client normally forks children to provide multiple network paths between the PFTP Client and the Mover(s). In some instances, it may be preferable to have these processes (pseudo children) running on independent nodes. In this case, it is necessary to setup a **multinoded** daemon on the independent node/host and have the PFTP client initiate the data transfer process(es) with these child processes. The **Multinode Table** stanza is used to specify what remote hosts are to perform the "pseudo" PFTP Client child processes functions.

Table 3-4 Multinode Table Stanza Fields

Configuration Type	Description
Stanza (Compound)	**`Multinode Table = {`** *Reserved* Stanza specifier. Must be terminated with a matching "}"
SubStanza (Compound)	**`<Local Hostname(s)>`** E.g. **`my_host my_host.my.domain = {`** Contains the local hostname(s) this SubStanza represents. MUST be terminated with a matching "**}**"

Section	**`<remote_host>`**
	`or`
	`<remote_host> = <Dot Notation Interface>`
	E.g. **`his_name or his_name = 100.102.103.45`**
	Contains the hostname in either string format or Dot Notation IP Address of the host to act as a "Pseudo" PFTP Child. If a secondary name is specified after the "=", the first interface is to be used for the "control" connection and the second specification is the interface to be used for the "data" connection(s) to the Mover(s). If only one value is provided, it represents both the "control" and "data" connections.

The **Multinode Table = { ... }** stanza contains one or more substanzas specifying the names of the host initiating the PFTP session.

Each section contains one or more names/IP addresses of remote hosts executing a Multimode Daemon (**multinoded**). The remote host must have appropriate entries for the **inetd** superdaemon (**/etc/inetd.conf** and **/etc/services**) to initiate the multinoded.

The sections may be either a simple section or a valued section. A simple substanza is a single name/Dot Notation IP Address to be used for both "Control" connection and "Data" connection. The valued substanza is used to specify the name/Dot Notation IP Address for the "Control" connection (specifier) and the name/Dot Notation IP Address for the "Data" connection (value.)

Multinode Table Stanza Rules:

- SubStanza hostnames (local hosts) may contain one or more hostnames separated by white spaces (subject to the 128 character limit.)
- Section hostnames (remote hosts) [and / or values] may be specified as either string-based hostnames or Dot Notation IP Addresses. Only one entry per line.

Multinode Table Example:

```
; Options read by the Multinode Daemon
Multinode Table = {
    ; Hostname of the Client
    water water.clearlake.ibm.com = {
        ; Specification of the Multinode Host
        ; If the Data Node is a different interface than the interface
        ; specified by the Control Node; then enter the Data Node/
        ; Interface after the "=" otherwise the Control and Data are
        ; the same.
        ; Control and/or Data may be dot notation OR string hostnames.
        water = 134.253.14.227
    }
    Default = {
        ; If the Data Node is different than the Control Node
        ; Enter the Data Node after the "=" otherwise the
        ; Control and Data are the same.
        ; Control and/or Data may be dot notation OR string hostnames.
        larry = sneezy
    }
}
```

3.7.1.4 Realms to DCE Cell Mappings Stanza

The **hpss_pftpd_amgr** is used to provide Kerberos credential-based authentication. This feature is required by the Kerberos FTP client or the HPSS **krb5_gss_pftp_client**. The **hpss_pftpd_amgr** daemon invokes an Authentication Manager, typically **auth_krb5gss**, to perform security functions on its behalf. When the Kerberos Parallel FTP Daemon is used in an environment where the **Kerberos Realm Name** is not the same as the **DCE Cell Name** (**NOT** Recommended!), it is necessary for the Authentication Manager to map the name of the incoming user appropriately. The **Realms to DCE Cell Mappings** stanza provides this feature.

Table 3-5 Realms to DCE Cell Mappings Stanza Fields

Configuration Type	Description
Stanza (Compound)	`Realms to DCE Cell Mappings = {` *Reserved* Stanza specifier. MUST be terminated with a matching "**}**"

Table 3-5 Realms to DCE Cell Mappings Stanza Fields

SubStanza	`<Realm> = <Cell>` E.g. `Kerberos.Realm = /…/my_dce_cell` Contains Kerberos Realms and their associated DCE Cell Names.

The **Realms to DCE Cell Mappings = { … }** stanza contains one or more substanzas providing Kerberos Realm to DCE Cell Mappings. The substanza(s) are used to specify the appropriate mappings.

Realms to DCE Cell Mappings specific rules:

For security reasons, only the **HPSS.conf** in either the path specified by the **HPSS_PATH_ETC** environment variable or **/var/hpss/etc** may be used. The **HPSS.conf** file must be owned by root and *cannot* be writeable by group or other. NOTE: if **HPSS_PATH_ETC** refers to an NFS mounted directory and root is not allowed access as root, the lookup will fail.

Realms to DCE Cell Mappings Example:

```
; Interface Specification (MUST be dot Notation)
Realms to DCE Cell Mappings = {
    ; Kerberos Realm = DCE Cell Name
    kerberos.realm.name = /.../dce.cell.name
}
```

3.7.1.5 Network Option Stanza

The Network Options are in the **Network Options = { … }** stanza of the **HPSS.conf** file.

The **Network Options** stanza allows different options to be specified based on: Source IP address [Local Interface Name(s)] AND Destination IP Address(es). When the Parallel FTP Client, Client API, or Mover establish connections, they will search the contents of this file for entries matching Source / Destination IP addresses and use the options specified in the matching entry.

The configuration file entries contain values / flags to be used for applying assorted socket and network options including: whether to enable / disable TCP Delay **(TcpNoDelay),** the socket send sizes **(SendSpace)** and / or socket receive sizes **(RecvSpace)**, the desired write buffer size **(WriteSize)**, and **RFC1323** support (“Large” Window.)

Which configuration entry to use is determined based on the Local Interface Name and the Destination IP address “masked” by the **NetMask** value. The calling application (PFTP Client, Client API, or Mover) will apply the value of the NetMask specification in the configuration file entry to the specified destination address. A "**Default**" destination may be specified for all sources/destinations not explicitly specified in the HPSS.conf file.

Table 3-6 Network Options Stanza Fields

Configuration Type	Description
Stanza (Compound)	**`Network Options = {`** *Reserved* Stanza specifier. Must be terminated with a matching "}"
SubStanza	**`Default Write Size = <value>`** E.g. **`Default Write Size = 4MB`** *Optional* SubStanza specifying the default write size if not specified explicitly.
SubStanza (Compound)	**`<Source IP Interface Name> = {`** E.g. **`my_host my_host.my.domain = {`** May contain one or more names separated by white spaces. May contain: "**`Default = {`**" (*Reserved* Specification) for inclusion of entries not explicitly specified. Must be terminated with a matching "}"
Section (Compound)	**`<Destination IP Address> = {`** E.g. **`100.101.102.0 = {`** Contains the dotted decimal address of the destination interface. Only one address is allowed; however, networks and sub-networks may be chosen by appropriate specification of the NetMask. May contain: "**`Default = {`**" (*Reserved* Specification) for inclusion of entries not explicitly specified. Must be terminated with a matching "}"
SubSection	**`NetMask = <Dotted IP Notation>`** E.g. **`NetMask = 255.255.255.0`** Contains the dotted decimal net mask to apply to the Destination IP Address to determine whether the entry applies
SubSection	**`RFC1323 = <value>`** E.g. **`RFC1323 = 1`** Indicates whether the RFC1323 option should be disabled (0) or enabled (any other value)
SubSection	**`SendSpace = <value>`** E.g. **`SendSpace = 1048576`** Contains the value to be used for the socket send buffer space May be specified as a decimal number or "*x*MB" style notation
SubSection	**`RecvSpace = <value>`** E.g. **`RecvSpace = 256KB`** Contains the value to be used for the socket receive buffer space. May be specified as a decimal number or "*x*MB" style notation

Table 3-6 Network Options Stanza Fields

SubSection	`WriteSize = <value>` E.g. `WriteSize = 1MB` Size to be used for each individual write request to the network May be specified as a decimal number or "*x*MB" style notation
SubSection	`TcpNoDelay = 0 | 1` E.g. `TcpNoDelay = 1` Indicates whether the TCP Delay option should be disabled (0) or enabled (any other value)

SendSpace & RecvSpace Controls the size of the receive and send buffers for TCP/IP sockets. Internally, HPSS servers and clients attempt to set these buffers sizes explicitly, but other utilities may not. Typically, the RecvSpace and the SendSpace are equal; however, this is not mandated. Setting either of these values in excess of the system maximum will result in a value <= the system maximum. The maximum can be observed/changed on AIX using the "**no**" command and observing/setting the **sb_max** parameter. There is no portable mechanism for determining the system maximum. Consequently, the specified values may be reduced until an acceptable value is obtained. This process involves a bit-shift operation (divide by 2.)

RFC1323 Controls whether large TCP window sizes are used. Usually turned on (1) for higher throughput networks (e.g. SP/x switch, Gigabit Ethernet, etc.) and turned off (0) for lower throughput networks (e.g. 10/100 Mb ethernet, FDDI, etc.) Large windows provide for increased performance over some networks, but may have a negative performance impact on others. NOTE: currently the ability to enable or disable **RFC 1323** support in this manner is specific to AIX. (An equivalent setting for Solaris is the **tcp_wscale_always** flag, set with the command "**ndd /dev/tcp tcp_wscale_always**".)

The **Default Write Size** allows the size of the individual write requests to the TCP/IP connections to be configured. The default behavior (if no entry in the file matches a connection or if zero is entered for the value of this field) is that the size of the write request is the size of the data buffer. On some networks (e.g. the SP/x switch), improved performance has been measured by using a smaller value (e.g. 32KB) for the size of the individual writes to the network. If no entry is found that matches a network connection or the value specified is zero, HPSS will query an environment variable, **HPSS_TCP_WRITESIZE**, and use that value, if set and non-zero, for the write size.

The **TcpNoDelay** option determines whether HPSS will enable or disable the algorithm that tries to improve performance from small network writes. This algorithm attempts to coalesce small writes to a TCP/IP connection so they can be sent in a single packet by delaying physical writes to the network. HPSS typically disables this algorithm so that delays are not experienced while sending Mover Protocol and parallel data transfer headers. However, if this causes a performance degradation on a specific network (e.g., causes smaller than optimal packet sizes for large transfers), this can be disabled for data transfer connections.

Network Options Stanza Specific Rules:

- The first matching entry found in the file will be used to determine the network options used for that connection.
- Multiple "Source Interface Name" SubStanzas may be included within the "Network Options" Stanza. A "Default" Source Interface Name SubStanza may be specified.

- The Source Interface Name SubStanza may specify one or more names [subject to the 128 character limit (including the "= {".)] NOTE: Do not include the quotes when specifying Default.
- Destination IP Address must be specified in Decimal Dot Notation.
- Multiple Sections may be included in any SubStanza. A "Default" Destination Interface Name Section may be specified. NOTE: Do not include the quotes when specifying Default.
- The NetMask must be specified in Decimal Dot IP Address Notation
- All SubSections must be specified in every Section.

Network Options Stanza Example:

NOTE: Tuning is a "fine art" and may vary dramatically within any network configuration and may change with only very minor network configuration modifications. Values provided below are not necessarily "good" numbers!

```
#  HPSS Network Options
Network Options = {
    # My Interface specification(s) using Interface Name
    # Notation
    my_host my_host.domain = {
        # Destination IP Address in Dot Notation
        100.101.102.103 = {
            # The netmask to be applied to the Dest. IP Address
            255.255.255.0
            # Use large IP Windows
            RFC1323 = 1
            # Socket Transmission Size.
            SendSpace = 1048576
            # Socket Receive Size.
            RecvSpace =  1MB
            # The overall buffer size to use for writing.
            WriteSize = 2MB
            # The TCP No Delay Flag is disabled
            TCPNoDelay = 0
        }
        # Default Destination – options to be used for destinations
        # NOT explicitly specified.
        Default = {
            NetMask = 255.255.255.0
            RFC1323 = 1
            SendSpace = 512KB
            RecvSpace =  512KB
            WriteSize = 256KB
            TCPNoDelay = 1
        }
    }
    # Values to be used for source hosts not explicitly specified
```

```
            Default  = {
                #  Destination IP Address in Dot Notation
                200.201.202.203 = {
                    NetMask = 255.255.255.0
                    RFC1323 = 1
                    SendSpace = 1048576
                    RecvSpace =  1MB
                    WriteSize = 2MB
                    TCPNoDelay = 0
                }
                # Default Destination – options to be used for destinations
                # NOT explicitly specified.
                Default = {
                    NetMask = 255.255.255.0
                    RFC1323 = 1
                    SendSpace = 256KB
                    RecvSpace =  128KB
                    WriteSize = 512KB
                    TCPNoDelay = 0
                }
            }
        }
```

3.7.1.6 "Reserved for Future Use" Stanzas

The following stanza names (specifiers) are reserved for future implementation in HPSS and should not be used by application developers.

- Pipe File
- Local File Path
- PFTP Daemon
- PSI

3.7.2 SP/x Switch Device Buffer Driver Buffer Pools

IBM SP/x systems provide the capability to tune the buffer pool allocation in the switch device driver. Two variables can be changed: **rpoolsize**, which is the size of the buffer pool for incoming data, and **spoolsize** which is the buffer pool size for outgoing data. If these values are too small, then buffer overruns may occur.

The current values of these variables can be interrogated with the **lsattr** command (e.g., "**lsattr -E -l css0**"), and can be changed with the **chgcss** command (e.g., "**chgcss -l css0 -a spoolsize=<new size> -a rpoolsize=<new size>**"). Refer to the *IBM Parallel System Support Programs for AIX Administration Guide, GC23-3897-02* for more details.

3.8 Install and Configure Java and hpssadm

3.8.1 Introduction

The **hpssadm** utility and the modifications to the SSM Data Server necessary to support **hpssadm** require the installation and configuration of Java 1.3.0 and the Java Secure Sockets Extension. The default prebuilt Data Server executable and shared library require Java. If the **hpssadm** utility is not used, these can be replaced with the no-Java prebuilt versions of these files, which are also shipped with HPSS. See Section 3.8.1.2: *Using the No-Java Version of the Data Server* on page 179 for more information on how to use the no-Java version.

Section 3.8.1.1: *Up and Running: Quick Configuration* on page 175 provides a quick configuration cookbook. If you use all the default values, you should be able to configure your system for the Java SSM using only these instructions. For more detailed information on the configuration procedure, see the remainder of Section 3.8.

3.8.1.1 Up and Running: Quick Configuration

This is a quick procedure for configuring Java and HPSS to support **hpssadm**, the Command Line SSM. For more detailed information on the configuration procedure, see the remainder of Section 3.8.

These examples assume that SSMDS and **hpssadm** will be executed on different machines, referred to here as the "SSMDS machine" and the "hpssadm machine". If SSMDS and **hpssadm** will be executed on the same machine, perform each step on that machine except where otherwise noted.

These procedures should be executed as **root**. Some of them require the DCE **cell_admin** password.

These procedures assign the same password, referenced as **<cacerts_password>**, to the trusted store file on all machines. They assign a unique password, referenced as **<ds_keystore_password>**, to the SSMDS keystore file.

Whenever the **keytool** utility prompts for a password, remember that it will echo the password you type in plain text.

For AIX, the environment variable **JAVA_HOME** should be **/usr/java130/jre**. For Solaris, it should be **/usr/j2se/jre**.

1. On the SSMDS and each **hpssadm** machine:

   A. Install Java 1.3.0 and JSSE 1.0.2.

   B. Reset password for the Certificate Trusted Store (**cacerts** file)

      This password is delivered as "changeit". Verify that this is the password by using it to access and read the trusted store, and then change the password to the desired value, referenced here as **<cacerts_password>**:

      i. Work in Java security directory:

```
% cd $JAVA_HOME/lib/security
```

ii. Check password for the trusted store (**cacerts**):

```
% $JAVA_HOME/bin/keytool -keystore cacerts -list
```

Type "changeit" when prompted for password.

iii. Change password for trusted store:

```
% $JAVA_HOME/bin/keytool -keystore cacerts -storepasswd \
-new <cacerts_password>
```

Type "changeit" when prompted for password.

iv. Verify new password for trusted store:

```
% $JAVA_HOME/bin/keytool -keystore cacerts -list
```

Type "**<cacerts_password>**" when prompted for password.

C. Configure the Secure Socket Layer (SSL)

These procedures define the Security Provider:

```
% vi $JAVA_HOME/lib/security/java.security
```

Look for the following lines:

```
% List of providers and their preference orders (see above):
% security.provider.1=sun.security.provider.Sun
security.provider.2=com.ibm.crypto.provider.IBMJCA
```

Add the following line:

```
security.provider.N=com.sun.net.ssl.internal.ssl.Provider
```

where "N" is the next available number (3 in this example)

2. On the machine where SSMDS will be executed:

These procedures use the keytool utility to create the ds keystore file. The administrator-selected password is set on the keystore file at creation time. For each use of the **keytool** utility here, type **<ds_keystore_password>** when prompted for password.

A. Work in the **/var** directory:

```
% cd /var/hpss/ssm
```

B. Configure key and certificates for SSMDS:

i. Generate a public and private key and a certificate for the SSMDS, and store them in the file **keystore.ds**:

```
% $JAVA_HOME/bin/keytool -genkey \
-dname "cn=HPSS Data Server" -alias hpss_ssmds \
-keystore keystore.ds -validity 365
```

ii. Display the fingerprint for the certificate:

```
% $JAVA_HOME/bin/keytool -keystore keystore.ds -list -v
```

iii. Export the certificate to the temporary file **ds.cer**:

```
% $JAVA_HOME/bin/keytool -keystore keystore.ds -export \
-alias hpss_ssmds -file ds.cer
```

C. Set up SSMDS for normal or low security mode:

i. For normal security mode, the administrator will be prompted for the password to the keystore file **<ds_keystore_password>** when the SSMDS begins execution. This means the Data Server may not be started automatically from **inittab**. To run in normal security mode, make sure the **HPSS_SSMDS_KEYSTORE_PASSWORD** variable is set to "**PROMPT**" in the **hpss_env** file.

This is the recommended operational mode.

ii. For low security mode, the SSMDS will read the password to its keystore from a file at startup. To run in low security mode, make sure the **HPSS_SSMDS_KEYSTORE_PASSWORD** variable is **not** set to anything in **hpss_env**, and store and protect the keystore password:

```
% vi keystore.ds.pw
```

Enter **<ds_keystore_password>** into file & save.

```
% chmod 600 keystore.ds.pw
```

This is not the recommended operational mode.

D. Set up SSMDS Java security policy file

```
% cp /opt/hpss/config/templates/java.policy.ds.template \
java.policy.ds
% chmod 640 java.policy.ds
% vi java.policy.ds
```

Change "***.hpss.acme.com**" to appropriate host info, such as "***.clearlake.ibm.com**", in the following section:

```
grant { permission java.net.SocketPermission
"*.hpss.acme.com:1024-",
"connect,accept,listen,resolve"; };
```

Save change and exit **vi**.

E. Set up the Client authorization file

```
% cp /opt/hpss/config/templates/hpssadm.config.template \
hpssadm.config
% chmod 640 hpssadm.config
```

Add any additional authorized users to the **hpssadm.config** file. To add user **joe**:

```
% vi /var/hpss/ssm/hpssadm.config
```

Add line "**HPSS_SSMDS_AUTH_USER=joe**" at the end of file.

Save and exit.

3. On each machine where **hpssadm** will be executed:

   A. Copy SSMDS certificate (**/var/hpss/ssm/ds.cer**) from SSMDS machine to **/var/hpss/ssm** on **hpssadm** machine.

      Note: Skip this step if SSMDS and each **hpssadm** client will run on the same machine.

   B. Import SSMDS certificate into the trusted store on the **hpssadm** machine:

```
% cd $JAVA_HOME/lib/security
% cp cacerts cacerts.orig
% $JAVA_HOME/bin/keytool -keystore cacerts -import \
-file /var/hpss/ssm/ds.cer -alias hpss_ssmds
```

      Type **<cacerts_password>** when prompted for password.

      At the "Trust this certificate?" prompt, check the fingerprint against the one displayed in step 2.B.ii. above. If it matches, answer "**yes**".

      The temporary file **/var/hpss/ssm/ds.cer** can be deleted after a successful import.

   C. Set up Java security policy file for **hpssadm**:

```
% cd /var/hpss/ssm
% cp /opt/hpss/config/templates/java.policy.hpssadm.template \
java.policy.hpssadm
% chmod 640 java.policy.hpssadm
% vi java.policy.hpssadm
```

      Change "***.hpss.acme.com**" to appropriate host info, such as "***.clearlake.ibm.com**", in the following section:

```
grant {
permission java.net.SocketPermission
"*.hpss.acme.com:1024-",
"connect,accept,listen,resolve";
};
```

      Save change and exit **vi**.

D. Create **hpssadm** user keytab files

```
% mkdir -p /var/hpss/ssm/keytabs
% cd /var/hpss/ssm/keytabs
```

For each **hpssadm** user on this machine, create DCE keytab file. This example creates a keytab file for user **joe**:

```
% dce_login cell_admin
% rgy_edit
rgy_edit> ktadd -f keytab.joe -p joe
```

(Need to enter joe's password twice)

```
rgy_edit> ktadd -f keytab.joe -p joe -r
rgy_edit> exit
```

Each **hpssadm** user must specify his keytab file as an argument to the utility:

```
hpssadm -k /var/hpss/ssm/keytabs/keytab.joe
```

4. Remember to update the SSMDS certificate when it expires. See Section 3.8.3: *Configuring SSL* on page 183 for instructions.

3.8.1.2 Using the No-Java Version of the Data Server

To use the no-Java version of the Data Server:

1. Save the Java versions of the executable and shared library files:

   ◆ On AIX:

   ```
   cp bin/hpss_ssmds bin/hpss_ssmds.java
   cp lib/libssmds.a lib/libssmds.a.java
   ```

   ◆ On Solaris:

   ```
   cp bin/hpss_ssmds  bin/hpss_ssmds.java
   cp lib/libssmds.so lib/libssmds.so.java
   ```

2. Then install the no-Java versions:

   ◆ On AIX:

   ```
   cp bin/hpss_ssmds.nojava bin/hpss_ssmds
   cp lib/libssmds.a.nojava lib/libssmds.a
   ```

   ◆ On Solaris:

   ```
   cp bin/hpss_ssmds.nojava  bin/hpss_ssmds
   cp lib/libssmds.so.nojava lib/libssmds.so
   ```

To use the **hpssadm** utility and the Java version of the Data Server, continue following the instructions for the remainder of this section.

3.8.1.3 Prerequisite Software

This required software is:

1. One of the following:
    - Java 1.3.0 JRE (Java Runtime Environment)
    - Java 1.3.0 SDK (Software Development Kit)
2. Java 1.0.2 JSSE (Java Secure Sockets Extensions)

This software is available for download for AIX, Solaris, and Windows at no cost. Section 3.8.2: *Installing Java* on page 181 lists the locations from which the software is available.

3.8.1.4 Security Mechanisms

Access by **hpssadm** clients to the Data Server is restricted by DCE authentication mechanisms and by a flat file authorization mechanism.The transmission of sensitive information such as passwords from the **hpssadm** utility to the Data Server is encrypted. In addition, both the Data Server and **hpssadm** client are executed under a Java Security Manager, which imposes restrictions on file system and network accesses.

Encryption mechanisms:

Encryption of the connection over which the user's DCE password is transmitted to the Data Server is implemented with Secure Sockets Layer (SSL). JSSE (Java Secure Sockets Extensions) is the Java implementation of SSL. SSL requires the creation and management of a public key and X.509 certificate for the Data Server. The creation and distribution of these certificates and other aspects of SSL are discussed in Section 3.8.3: *Configuring SSL* on page 183.

Java Security Manager restrictions:

The Java Security Manager requires that a Java security policy file for the Data Server and for each **hpssadm** user be created and maintained. This limits the file system and network socket access of each program, over and above the regular system protections. The policy file is discussed in Section 3.8.4: *Configuring the Java Security Policy File* on page 186.

Authorization mechanisms:

Only users specified in the **hpssadm.config** file will be authorized to connect to the Data Server using **hpssadm**. This file is discussed in Section 3.8.5: *Setting up the Client Authorization File* on page 188.

Authentication mechanisms:

Only users with valid DCE login ids and passwords will be allowed to connect to the Data Server using **hpssadm**. Each user's login name and password are stored in a private keytab on the host from which the **hpssadm** utility is executed and are transmitted to the Data Server, who authenticates them against the DCE registry. This file is discussed in Section 3.8.6: *Setting up the hpssadm Keytab File* on page 189.

Section 3.8.9: *Background Information* on page 191 provides a high level discussion of DCE keytab files, the Java Security Policy, X.509 certificates and public key encryption as they are used by the hpssadm utility and the Data Server, and includes references to additional documentation for these technologies.

3.8.1.5 Environment Variables

For the examples in this section, the environment variable **JAVA_HOME** is the directory used by the Java runtime. If the Java SDK (Software Developer Kit), sometimes called the JDK (Java Developer Kit), is installed on the system, then the **JAVA_HOME** directory is the **jre** subdirectory of the top-level directory into which the SDK was installed. If only the JRE (Java Runtime Environment) is installed on the system, then the **JAVA_HOME** directory is the top-level directory in which the JRE was installed.

Examples in this section are for Unix systems.

3.8.2 Installing Java

3.8.2.1 Obtaining the Software

For AIX:

With AIX 5.1, Java 1.3.0 is available as part of the OS and can be installed during the initial AIX installation, at AIX upgrade time, or later as an individual product. However, due to a conflict with the JSSE libraries used by **hpssadm**, do not install the **Java130.jsse-us** component. If this component is already installed, disable it as follows:

```
% su -
% cd ${JAVA_HOME}/lib/ext
% mv ibmjsse.jar ibmjsse.jar.org
```

For Solaris:

Java 1.3.0 may be downloaded from:

http://java.sun.com/j2se/1.3/

For AIX and Solaris:

JSSE 1.0.3 may be downloaded for AIX, Solaris, or Windows from

http://java.sun.com/products/jsse/

There are two versions of JSSE, one for download to U.S. and Canada sites and one for export to other countries.

Follow the instructions with the downloads for installation.

Please observe these notes about the JSSE installation:

It is recommended but not required that you download both the JSSE package and the documentation.

The JSSE zip file may be unpacked anywhere desired. It is recommended that it be unpacked directly under the **${JAVA_ROOT}** directory to make it easier to find.

The JSSE installation instructions say the libraries may be installed as an "installed extension" or bundled with the application. The **hpssadm** utility and SSM Data Server expect JSSE to be an installed extension, so that the libraries are stored in

```
${JAVA_HOME}/lib/ext
```

The JSSE installation instructions explain two ways to register the provider. Use the static registration, which is also explained in Section 3.8.3.1: *Installing the Security Provider* on page 183.

3.8.2.2 OS Patch Levels

Please check the recommended operating system patch sets listed by the web sites listed above (Section 3.8.2.1: *Obtaining the Software* on page 181). Remember that the recommended patch sets are subject to change at any time by Sun or IBM. HPSS was tested with Java 1.3.0 under AIX 5.1 Maintenance Level 2, and under Solaris 5.8 with Recommended Patch Cluster 73001.

3.8.2.3 Setting the Password for the Certificate Trusted Store

A trusted store of X.509 certificates is delivered with the JDK, even if you don't install the JSSE. This file contains several root level certificates from Verisign and other companies. This file is shipped with an initial password of "**changeit**". You should change this password when you install the JDK on each host where the **hpssadm** utility will execute. The default file shipped with the JDK is "**cacerts**". If someone else has installed the JDK on your system, they may already have set this password or renamed or removed this file. The JSSE software will look first for the file "**jssecacerts**" as its default trusted store and then for "**cacerts**", so if someone else has installed the software on your system, they may have added the **jssecacerts** file. No trusted store file should be left with the default password.

To change the default password on the delivered file:

1. cd to the directory holding the trusted store. This should be

   ```
   $JAVA_HOME/lib/security/
   ```

2. Find the trusted store file(s), **cacerts** and / or **jssecacerts**.

3. Check the password by listing the file; in this example, we are checking the **cacerts** file; do this for each trusted store file, substituting the correct file name for the "**-keystore**" option:

   ```
   $JAVA_HOME/bin/keytool -keystore cacerts -list
   ```

You will be prompted for the password, WHICH WILL BE ECHOED AS YOU TYPE IT, so make sure you are working from a location where the password cannot be compromised. Type in the default password ("**changeit**"). The utility should list the certificates in the file.

4. Change the password with the **-storepasswd** option of the **keytool** command. In this example, the new password is "**XXXXXX**". Again, we are changing the password for the **cacerts** file; do this for each trusted store file, substituting the correct file name for the "**-keystore**" option:

   ```
   $JAVA_HOME/bin/keytool -keystore cacerts -storepasswd \
   -new XXXXXX
   ```

5. Verify that the password was changed properly by listing the file again:

   ```
   $JAVA_HOME/bin/keytool -keystore cacerts -list
   ```

   Again, your password will be echoed as you type it, so be sure no one can read your screen.

This change should be performed on the Data Server host machine and on any host from which **hpssadm** will be executed.

The installation instructions for Java 1.3.0 also include directions for changing this password.

See the **keytool** man page with your Java installation for more information on using the **keytool** utility.

3.8.3 Configuring SSL

SSL (Secure Sockets Layer) must be configured for the Data Server even if the **hpssadm** utility is not executed, because the Data Server reads its private key as part of its initialization, even if he subsequently never needs it for any hpssadm client. The steps below which are necessary for the Data Server are distinguished from those necessary only for **hpssadm**.

SSL is used to encrypt the transmission of the user's DCE user name and password from the **hpssadm** utility to the Data Server. In fact, the entire session by which the **hpssadm** utility submits commands to the Data Server is encrypted with SSL.

Be aware, however, that there is a second session between the Data Server and the **hpssadm** utility. This second, independent session is the one by which the Data Server sends the **hpssadm** client asynchronous notifications of changes in HPSS statuses, such as a notice that a server has gone down or a device opstate has changed. No password information is transmitted across this session, and it is not encrypted.

This section explains how to configure the Java SSL extension and the Data Server for use with SSL.

3.8.3.1 Installing the Security Provider

In order for Java to access the SSL extension, the SSL provider must be installed. To do this, add the provider to the Java security file

```
$JAVA_HOME/lib/security/java.security
```

There should already be at least one security provider listed in this file, probably in a format something like:

```
security.provider.1=sun.security.provider.Sun
```

If there is more than one provider listed, they should be numbered in increasing numerical order:

```
security.provider.2=XXX.security.provider.foox
security.provider.3=YYY.security.provider.fooy
security.provider.4=ZZZ.security.provider.fooz
etc.
```

Add the line for the SSL provider like this, substituting for "**N**" in this example the next available number:

```
security.provider.N=com.sun.net.ssl.internal.ssl.Provider
```

3.8.3.2 Configuring Keys and Certificates for the Data Server

Step 1 below is necessary for the proper configuration of the Data Server. All the other steps in this section are required only for the configuration of the **hpssadm** utility.

The use of the SSL protocol between **hpssadm** and the Data Server requires that a public/private key pair be generated for the Data Server and that the Data Server present an X.509 certificate to identify himself to the **hpssadm** client. The **hpssadm** client must have access to a trusted store of certificates which includes either the Data Server's certificate or the certificate of a certificate authority who has signed the Data Server's certificate. If your site requires certificates to be signed by an authority such as Verisign, see your site security personnel for instructions for generating the public/private key pair and obtaining a signed certificate for the Data Server. If a self-signed certificate for the Data Server is acceptable to your site, follow the instructions in this section.

On the machine where the Data Server will be executed:

1. Create a public/private key pair and a certificate for the Data Server using the **keytool** utility.

   You can choose any name you wish for the Data Server; in this example, we have called it "**HPSS Data Server**". You must also specify an alias for the Data Server, for which we have used "**hpss_ssmds**". The key pair and certificate must be stored in a keystore, a file that will be private to the Data Server. The default name for this keystore file is

   ```
   /var/hpss/ssm/keystore.ds
   ```

   This name can be changed in the **hpss_env** file by setting the **HPSS_SSMDS_KEYSTORE** variable as desired. The keystore file will be protected with a password, which should be unique and used only for protecting this keystore and the key within it.

   ```
   % cd /var/hpss/ssm
   % $JAVA_HOME/bin/keytool -genkey -dname "cn=HPSS Data Server" \
   -alias hpss_ssmds -keystore keystore.ds -validity 365
   ```

This command will generate a public key and an associated private key for the Data Server with alias "**hpss_ssmds**". It will also generate a self-signed certificate for **hpss_ssmds** which includes his public key. The key will be valid for 365 days. The keys and certificate will be stored in the file "**keystore.ds**". This is the file the Data Server will read to obtain his key and certificates when he first begins execution.

After typing this command, you will be prompted for the password for the keystore. It will be echoed to the terminal, so don't do it while anybody is watching!

You will also be prompted for a password for the key itself. Individual keys within a keystore are additionally protected by their own password, which may be different from the keystore password. The Data Server expects the key password to be the same value as the password to the keystore itself, so use the same one.

Anyone who must start the Data Server in normal security mode must know this password. If the Data Server is started in low security mode, this password must be stored on disk, as described in Section 3.8.3.3: *Storing the Password to the Data Server's Keystore File* on page 186.

This is the only step in this section (3.8.3.2) which is necessary for the proper configuration of the Data Server. The remaining steps in this section are necessary only for the configuration of the **hpssadm** utility.

2. Obtain and record the fingerprint for the Data Server's certificate using the **keytool** utility:

```
% $JAVA_HOME/bin/keytool -keystore keystore.ds -list -v
```

   This will list every key in the keystore (which should be just the one for the Data Server) and its certificate fingerprint, a long number representing the certificate. This fingerprint will be used like a checksum to verify the validity of the certificate as it is transferred to hpssadm client machines.

3. Export the Data Server's certificate from the keystore:

```
% $JAVA_HOME/bin/keytool -keystore keystore.ds -export \
-alias hpss_ssmds -file /tmp/ds.cer
```

   You will be prompted for the keystore password. Then the file "**/tmp/ds.cer**" will be created, which will hold a binary representation of the Data Server's certificate.

   The **/tmp/ds.cer** file is just a temporary file for transferring a copy of the Data Server's certificate to the **hpssadm** utility's trusted store. You can name it anything you want and remove it once you are finished with it.

On each machine from which the hpssadm utility will be executed:

1. Transfer the certificate file **ds.cer** to the **hpssadm** client machine.

   Use the mechanism (**ftp**, etc.) of your choice. scp is recommended.

2. Import the Data Server's certificate into the trusted store on the **hpssadm** client machine. It is a good idea to save the original trusted store file (**cacerts**) first:

```
% cd $JAVA_HOME/lib/security
% cp cacerts cacerts.ORIG
% $JAVA_HOME/bin/keytool -keystore cacerts -import \
-file /tmp/ds.cer -alias hpss_ssmds
```

The **keytool** utility will print out the information about the certificate, including the fingerprints, and will ask whether the certificate should be trusted. Compare the owner, issuer, and fingerprints carefully with those obtained from the original certificate in step 2. If they match, answer "**yes**". If they do NOT match, DO NOT import the certificate at all; it has been corrupted in transit.

If you confirm that you want the certificate added as trusted, the utility should respond that the certificate was added to the keystore.

This **cacerts** file is the file the **hpssadm** client will use to verify the Data Server's certificate.

The **/tmp/ds.cer** file is just a temporary file for transmitting a copy of the Data Server's certificate. It may be named anything you like, and may be removed once you have used it to import the certificate into the **hpssadm** trusted store.

3.8.3.3 Storing the Password to the Data Server's Keystore File

This step is necessary for the proper configuration of the Data Server.

When the Data Server is executed in Low Security mode, the password to its keystore file must be stored in a file on the Data Server host. This is one reason it is so important to secure this machine. This file must be protected against access by any user except root, and the Data Server must be executed as root. Low Security mode is the only mode in which the Data Server may be started automatically from a script, without human intervention.

The default name for the file to store the password is

```
/var/hpss/ssm/keystore.ds.pw
```

This name can be changed in the hpss_env file by setting the **HPSS_SSMDS_KEYSTORE_PASSWORD** variable as desired.

To run the Data Server in Normal Security mode, set the **HPSS_SSMDS_KEYSTORE_PASSWORD** variable in the **hpss_env** file to the string "**PROMPT**". Then, rather than reading the password from a file, the Data Server will prompt the user for the password when it begins execution. If you always run in Normal Security mode, you do not need to store the password to the Data Server's keystore anywhere in a file, but neither can you start it automatically from a script.

3.8.4 Configuring the Java Security Policy File

A Java security policy file is required for the Data Server. If the **hpssadm** utility is used, it must have its own Java security policy file.

Versions of Java beginning with 1.2 allow you to fine tune many permissions given to particular code by means of system wide, user, and application specific policy files, and by providing for applications to run under the Java Security Manager. If the application is not executed with a

Security Manager, or if none of these policy files exists, the default policy is the original Java sandbox policy, which is rather liberal.

Any system access is further limited by whatever protections the local operating system supplies. So, for example, if the policy file allows access to file "**foo**", but the file system permissions do not permit access to "**foo**" by the user executing hpssadm, then the user cannot access the file.

The SSM Data Server and the **hpssadm** utility have been written to be executed under a Security Manager so that we may impose further restrictions than the sandbox, particularly the ability to restrict accesses to a specified set of network addresses. The Security Manager is set up inside the Data Server and **hpssadm** code. The HPSS administrator controls the privileges granted to the code by means of the policy files.

The names of the policy files are specified in the system security properties file, **$JAVA_HOME/lib/security/java.security**. By default, a system wide policy file **$JAVA_HOME/lib/security/java.policy** is checked first, and then the file "**.java.policy**" in the user's home directory. Alternate policy files can be specified for an application at runtime. The Data Server and **hpssadm** utility expect an alternate policy file at runtime. By default, these files are

```
/var/hpss/ssm/java.policy.ds
/var/hpss/ssm/java.policy.hpssadm
```

on the machine where the Data Server or **hpssadm** utility is executing, respectively. These file names can be changed in the **hpss_env** file by setting the **HPSS_SSMDS_JAVA_POLICY** and **HPSS_HPSSADM_JAVA_POLICY** variables as desired. See the files **config/templates/java.policy.ds.template** and **config/template/java.policy.hpssadm.template** for sample policy files. These files should be copied to the **/var/hpss/ssm** area on the appropriate machines and customized as desired for your site.

The minimum privileges which must be granted to the Data Server are those to allow it to load its native library, to read its configuration file, and to communicate across the network with **hpssadm** clients. The **hpssadm** client must have privileges to read the user's keytab file and to communicate across the network with the Data Server:

1. Native library access requires **RuntimePermission loadLibrary**. The name of the Data Server's native library is **libssmds.a** (**libssmds.so** on Solaris), so the entry for the policy file is:

   ```
   grant {
      permission java.lang.RuntimePermission "loadLibrary.ssmds";
   };
   ```

   Additionally, although we can find no reference to it in any of the Java documentation, the interpretation of this library name requires read permission on the **java.execsuffix** property. This policy file entry is:

   ```
   grant {
      permission java.util.PropertyPermission "java.execsuffix",
   "read";
   };
   ```

   These two entries are necessary only in the Data Server policy file. They are not needed in the **hpssadm** policy file.

2. The Data Server requires **read FilePermission** on its user authorization file, whose default location is **/var/hpss/ssm/hpssadm.config**. The **hpssadm** utility requires **read FilePermission** for the user's keyfile file, the default location for which is **/var/hpss/ssm/keytab**

    ```
    grant {
       permission java.io.FilePermission "/var/hpss/-", "read";
    };
    ```

    The dash ("**-**") in the pathname in this example signifies that the permission is to be granted to everything in the **/var/hpss** tree, recursively. Sites which wish to be more restrictive can write a separate grant clause for each file or directory to which they want to allow access.

    Java **FilePermission** is applied as an additional layer of protection on top of the local operating system file protections, not as a replacement for them. If the Java permission is not granted, the application will not be allowed to access the file, regardless of the local file system permissions. If the Java permission is granted but the local file system permissions deny access to the file, the application will not be allowed access.

3. The Data Server and the **hpssadm** utility may restrict the remote hosts with which they will communicate by setting their **SocketPermission**.

    According to the documentation, and upheld by some of our testing, you should not need an explicit **SocketPermission** in the policy file just to listen on public ports nor to connect to applications on the same or other hosts; that permission is supposed to be granted implicitly. But we've found some implementations on which, even with the system security and policy files set the same, the applications required that at least **connect** and **listen** permission be granted explicitly from a policy file. So, partly for this reason, we include this permission in the default policy files for both the Data Server and **hpssadm**.

    The other reason we include this permission entry is that it can be restricted to a single host or set of hosts and / or ports. The following example grants access to all hosts from the **ornl.gov** domain:

    ```
    grant {
       permission java.net.SocketPermission
       "*.ornl.gov:1024-",
       "connect,accept,listen,resolve";
    };
    ```

Sites which wish to operate under tighter security can set the Java security file so that only the system wide policy file is recognized and specification of an alternate or additional policy file on the Java command line is not allowed.

See the document on Java policy file syntax listed in Section 3.8.9.2: *References* on page 194 for more information on settings policies.

3.8.5 Setting up the Client Authorization File

This file must exist in order for the Data Server to be initialized, but it may be empty if there is no desire to use the **hpssadm** utility.

The **hpssadm.config** file is a flat ASCII file which lists the users who are authorized to use the **hpssadm** utility. The template for this file is **config/templates/hpssadm.config.template**. The default name for this file is

```
/var/hpss/ssm/hpssadm.config
```

This pathname can be changed in the **hpss_env** file by setting the **HPSS_SSMDS_JAVA_CONFIG** variable as desired. For each user who is to be allowed to use the utility, a line must be added to this file of the form:

```
HPSS_SSMDS_AUTH_USER=dce_name
```

For example:

```
HPSS_SSMDS_AUTH_USER=joe
HPSS_SSMDS_AUTH_USER=hpss_ssm
```

3.8.6 Setting up the hpssadm Keytab File

This step is necessary only for use of the **hpssadm** utility.

Each user of the **hpssadm** utility must have a private keytab file containing his dce username and password. Keytab files can be created using **rgy_edit**. This procedure works for creating a keytab file. Perform these steps on each machine from which the **hpssadm** utility will be executed. This example is for a user named "**joe**", and creates a keytab file named "**keytab.joe**" for him:

1. login as either **root** or **hpssadm**.

2. **dce_login** as **cell_admin**

3. **cd** to the directory in which you wish to store the keytab file. This must be a directory which is accessible by the **hpssadm** user. The directory must be protected so that no one other than the **hpssadm** user can delete his keytab file. **/var/hpss/ssm** is recommended.

4. Make sure the user has a valid dce account and password.

5. Add the user's entry to the keytab file using **rgy_edit**:

   ```
   % rgy_edit
   rgy_edit> ktadd -f keytab.joe -p joe
   ```

   You will be prompted for joe's dce password, twice.

6. Sync the entry with the registry:

   ```
   rgy_edit> ktadd -f keytab.joe -p joe -r
   ```

   You will be prompted for **joe**'s dce password again, twice.

7. Save the file.

   ```
   rgy_edit> exit
   ```

The keytab file must be stored on each host from which the user will execute the **hpssadm** utility, and must be specified on the **hpssadm** command line with the **-k** option:

```
hpssadm -k keytab_file_path_name
```

The keytab file should be owned by the user and protected so that it is readable only by the user.

The keytab is interpreted on the host on which the Data Server runs, not that on which the **hpssadm** client utility runs. Therefore, it is not necessary to have DCE on the client machines.

3.8.7 Securing the Data Server and Client Host Machines

It is critical that the Data Server be executed on a secured machine in order to protect its keystore password and in order to avoid RMI registry hijacking.

As described in Section 3.8.3: *Configuring SSL* on page 183, the Data Server must have access to the password to its keystore file. If your site runs the Data Server in Low Security Mode, the Data Server obtains this password from a cleartext file. Compromise of this file could allow an illicit program to obtain the Data Server's private key and impersonate him. Such a program could then collect the DCE passwords of your **hpssadm** users. This file must be readable only by root, stored on a machine not accessible to untrusted users, and stored in an area not network-shared with any other host.

The Data Server should be executed as root so that it can read its password file. It is not necessary that the SSM System Manager be executed as root. The **start_ssm** script has been modified so that if it is executed as root, it will start the Data Server as root but the System Manager as user "**hpss**". If the **start_ssm** script is executed under any other userid, it will start both the Data Server and System Manager under that userid. This is not recommended, but if the site chooses to do it, then the Data Server's password file must be readable by this userid, and it should at least be protected against access by any other user.

If the site chooses to run in Normal Security Mode, in which the password is not stored on disk and the administrator is prompted for it at Data Server startup, then the userid under which the Data Server executes doesn't matter.

A second way an imposter could impersonate the Data Server is to override his binding in the RMI registry. The RMI registry does not allow such access to processes from other hosts, but any process on the local host can rebind any entry in the RMI registry and so pretend to be the Data Server. No special privileges are required. Since an unprivileged program would still not have access to the Data Server's private key, it ought not to be able to certify its identity to hpssadm clients nor induce them to hand it their passwords, but it is still desirable that only trusted users have access to the machine where the Data Server and its RMI registry are executing.

Each host from which the **hpssadm** utility is executed must be secure enough to insure that the user's keytab file and trusted certificate store cannot be compromised. An illicit process which gained access to the keytab file could gain the user's credentials anywhere in the DCE cell. A process which could modify the trusted certificate store could insert certificates for any entity, and the **hpssadm** program would trust processes run by that entity.

3.8.8 Updating Expired SSL Certificates

When the Data Server certificate expires, the Data Server itself will be able to start up and execute, but any **hpssadm** client attempting to connect to it will fail with the error "untrusted server cert chain". A new certificate must be generated for the Data Server and disseminated to all the client machines. To do this, follow these steps:

1. Check the keystore and the **cacerts** file to be sure the certificate has expired. On the host where the Data Server executes, check the Data Server keystore:

   ```
   % cd /var/hpss/ssm
   % $JAVA_HOME/bin/keytool -keystore keystore.ds -list -v
   ```

   On each host where an **hpssadm** client executes, check the **cacerts** file:

   ```
   % cd $JAVA_HOME/lib/security
   % $JAVA_HOME/bin/keytool -keystore cacerts -list -v
   ```

   Look in the output for the Data Server certificate and its expiration date. If the certificate has not expired, there is no need to continue with this procedure; pursue the problem diagnosis steps in Chapter 13: *HPSS Problem Diagnosis and Resolution* (page 485) in the *HPSS Management Guide*.

2. On the host where the Data Server executes, delete the **keystore.ds** file with the expired certificate.

   ```
   % cd /var/hpss/ssm
   % rm keystore.ds
   ```

3. On each host where an **hpssadm** client executes, delete the expired certificate from the **cacerts** file:

   ```
   % cd $JAVA_HOME/lib/security
   % $JAVA_HOME/bin/keytool -keystore cacerts -delete -alias \
   hpss_ssmds
   ```

4. Recreate the **keystore.ds** file, export the certificate, and import it into the **cacerts** file on every **hpssadm** client machine, using the original installation procedures in this chapter.

3.8.9 Background Information

3.8.9.1 Basic Security Technologies Relevant to the SSM Command Line Utility

This section is intended to provide an overview of the basic principles of the Java security policy, public key encryption, SSL, and X.509 certificates as they are used in the SSM Data Server and Command Line Utility. For a more thorough discussion of these technologies, see the references in Section 3.8.9.2: *References* on page 194.

Java allows code to run under a Security Manager. This is basically a library that gets called any time a security-related operation, such as an access of the local file system, is requested. The library call returns silently if it determines the code is allowed the requested access, and otherwise throws an exception, which halts the program.

Applet code runs under a security manager (usually) because most browsers implement one. The security manager won't let the applet do anything not allowed by the policy file(s). Applets are not allowed to install security managers; browsers do it first thing, anyway, and nobody can install a second one in a running program

Applications don't have to run under a security manager. If they choose to run under a security manager, then, like applets, they can do only what the policy file(s) allow. Code must have **java.lang.RuntimePermission setSecurityManager** in order to set the security manager, or else it gets the default Security Manager.

By default, the **java.security** file lets you pass additional java policy file on command line; this can be disabled in **java.security** by changing

```
policy.allowSystemProperty=true
```

to **false**:

```
policy.allowSystemProperty=false
```

By default, the **java.security** file specifies system wide and user policy files; this, too, can be changed in the file.

SSL provides the SSM Data Server and the **hpssadm** utility a secure encrypted channel over which to transport the **hpssadm** user's password. SSL requires the use of two kinds of keys, symmetric and public, and of X.509 certificates.

A key is a number used with an encryption algorithm to encrypt or decrypt data.

Anything encrypted with a symmetric key can be decrypted only by the same symmetric key. If two parties have access to the same symmetric key, they can use it to shared encrypted information.

Public keys are created in pairs consisting of a public key and a private key. Anything encrypted with the public key can be decrypted only with the private key, and vice versa, anything encrypted with the private key can be decrypted only with the public key. In general, a user is the only one who has access to his own private key, but he makes his public key known to everybody. Then anybody can encrypt data for him in his public key, and he is the only one who can decrypt it.

Symmetric key encryption is faster than public key encryption, but public key encryption is easier to manage, because you don't have to distribute and protect a shared key to all the parties involved. The private key is retained by one party and protected; the public key is distributed to everyone and need not be protected.

A digital signature is an encrypted piece of data used to validate the identity of the sender. Digital signatures are created by having a party encrypt some known piece of data in his private key. Then anybody can decrypt the data using his public key, and if the decryption works properly, they know the signature is valid and only the true party could have sent it, since he's the only one with access to the private key.

An X.509 certificate is a digitally signed electronic document identifying a party. It includes, among other things, a name representing the party, a representation of his public key, and a digital signature of some certificate authority. A certificate authority is a company, like Verisign, whom you can pay to issue X.509 certificates to you. Certificates can also be created by individuals and self-signed by the party owning the certificate. A program uses a file of these certificates as its "trusted store", the set of certificates of parties it will trust.

Whereas a digital signature confirms that the issuing party possesses the private key corresponding to a particular public key, a certificate confirms that some verification has been done as to the identity of its owner. After all, anyone can generate a public/private key pair, publish the public key, and claim to be User X or Company Y. A certificate proves that some verification has been done (by the certificate authority) to insure that the party really is User X or Company Y, and it binds the party to his public key. If you pay a company like Verisign for a certificate, they contact you and follow some procedures to ascertain that you really are who you claim to be.

The validity of a certificate itself can be verified by a checksum called a fingerprint. Before the system administrator places a certificate in his trusted store, he checks its fingerprint against the known value for that certificate. For certificates from authorities like Verisign, these fingerprints are published somewhere, like a web page. For self-signed certificates, the administrator gets the fingerprint from the certificate owner/creator, who of course should be someone the administrator trusts.

When one program presents its certificate to a second program, the second program checks its trusted store for the certificate. If it is there, then the second program assumes the first program is really who he claims to be. More often, the certificate of the first program is not in the trusted store, but it is signed by a certificate authority whose certificate IS in the trusted store. You can also have chains of certificates: a certificate is signed by an authority whose certificate is signed by an authority whose certificate is in the trusted store. So long as the second program can find a certificate in the chain which is in his trusted store, he can trust the certificate owner. This chaining ability was designed to allow systems to verify certificates from many arbitrary parties, without having to import each of their certificates individually (which would involve manually verifying their fingerprints).

SSL uses X.509 certificates to identify the server (in our case, the SSM Data Server) to the client (in our case, the **hpssadm** utility). Then the client generates a secret piece of data and passes it to the server using the server's public key. The client and server use this initial secret piece of data to negotiate a new shared symmetric key, and then use the symmetric key for the remainder of the session.

Java stores keys and certificates in files called keystores. The keystore file is password-protected; even if you have read-access to the file, you can't get the key out unless you know this password. SSL requires that the server be able to obtain his private key, so he must know this password. This means that the server must be started manually and allow a user to type in this password, or that the password must be stored online somewhere. The SSM Data Server may be executed either way; in Low Security mode, the password to the keystore is stored on a file on the Data Server's machine and read by the Data Server at startup time. In Normal Security mode, the Data Server prompts for the password at the beginning of execution.

The Data Server and the **hpssadm** utility are connected by two RMI (Java Remote Method Invocation) sessions. In the first session, the Data Server acts as the server and **hpssadm** acts as the client. This is the session on which all **hpssadm** requests, such as to start or stop servers, force migrations, or lock devices are issued. It is also the session over which the **hpssadm** user's password is transferred to the Data Server. This session uses SSL underneath RMI to encrypt the entire connection, not just the password.

In the second session, the **hpssadm** utility acts as the server and the Data Server acts as the client. This is the session used by the Data Server to send asynchronous data change notifications to the

hpssadm program, such as new alarms or changes in HPSS server statuses. This session does not pass any private data such as passwords, does not use SSL, and is not encrypted.

For security reasons, an application can bind or unbind only to an RMI registry running on the same host. This prevents a client from removing or overwriting any of the entries in a server's remote registry. A lookup, however, can be done from any host.

3.8.9.2 References

For a description of public key encryption, digital signatures, and certificates, see:

Introduction to Public-Key Cryptography
http://developer.netscape.com/docs/manuals/security/pkin/index.htm

For a description of the Secure Sockets Layer (SSL) protocol, see:

Introduction to SSL
http://developer.netscape.com/docs/manuals/security/sslin/contents.htm

For a description of the Java policy file, see:

Default Policy Implementation and Policy File Syntax
http://java.sun.com/products/jdk/1.2/docs/guide/security/PolicyFiles.html

For the man page for the Java keytool utility for managing keys and trusted stores, see:

keytool - Key and Certificate Management Tool which for Unix platforms is at:
http://java.sun.com/products/jdk/1.2/docs/tooldocs/solaris/keytool.html

and for Windows platforms is at:
http://java.sun.com/products/jdk/1.2/docs/tooldocs/win32/keytool.html

3.9 Setup Linux Environment for XFS

After installing the appropriate versions of Linux and XFS software on your system, use the following steps to prepare the OS for an XFS / HPSS filesystem:

1. Apply the HPSS Linux XFS Patch (not required for RAIDZONE systems)

2. Create the XFS DMAPI Device

3.9.1 Apply the HPSS Linux XFS Patch

Use the following procedure to download, install, and configure your Linux environment for an XFS HDM (not required for RAIDZONE systems):

1. Download the patch, **xfs-2.4.18-1**, from the HPSS website (`http://www4.clearlake.ibm.com/hpss/support/patches/xfs-2.4.18-1.tar`).

2. Untar the downloaded file.

```
% tar -xvf xfs-2.4.18-1.tar
```

3. Copy **xfs-2.4.18-1** (the patch file) to **/usr/src**.

```
% cp xfs-2.4.18-1 /usr/src
```

4. Change directory to **/usr/src/linux-2.4.18** (or the root of your 2.4.18 kernel tree).

```
% cd linux-2.4.18
```

5. Apply the patch

```
% patch -p1 < ../xfs-2.4.18-1
```

6. Run **make config** and answer **yes** when questioned about XFS filesystem and DMAPI support.

7. Remake the dependencies

```
% make dep
```

8. Build a new kernel

```
% make bzImage
```

9. Copy the **bzImage** to **/boot**, add a new stanza to the **/etc/lilo.conf** file, and run **/sbin/lilo**.

```
% cp arch/i386/boot/bzImage /boot/vmlinuz-2.4.18-xfs
% vi /etc/lilo.conf
% /sbin/lilo
```

3.9.2 Create the DMAPI Device

XFS utilizes a special device to allow communication between the userspace library linked by the HDM and the XFS portion of the kernel. This device must be created before any DMAPI applications will be able to operate successfully. To create the device, run the following command as **root**:

```
% mknod /proc/fs/xfs_dmapi c 10 140
```

3.10 Setup Linux Environment for Non-DCE Mover

Use the following procedure to download, install, and configure your Linux environment for a Non-DCE Mover (note necessary for RAIDZONE systems):

1. Download the patch, **kaio-2.4.18-1.tar**, from the HPSS website (http://www4.clearlake.ibm.com/hpss/support/patches/kaio-2.4.18-1.tar).

2. Untar the downloaded file.

   ```
   % tar -xvf kaio-2.4.18-1.tar
   ```

3. Copy **kaio-2.4.18-1** (the patch file) to **/usr/src**.

   ```
   % cp kaio-2.4.18-1 /usr/src
   ```

4. Change directory to **/usr/src/linux-2.4.18** (or the root of your 2.4.18 kernel tree).

   ```
   % cd linux-2.4.18
   ```

5. Apply the patch

   ```
   % patch -p1 < ../kaio-2.4.18-1
   ```

6. Run **make config** and answer **yes** when questioned about AIO support. The default value of 4096 should be sufficient for the number of system-wide AIO requests. At this time, you should also configure the kernel to support your disk or tape devices. If tape device access is required, be sure to also enable the kernel for SCSI tape support.

7. Remake the dependencies

   ```
   % make dep
   ```

8. Build a new kernel

   ```
   % make bzImage
   ```

9. Copy the **bzImage** to **/boot**, add a new stanza to the **/etc/lilo.conf** file, and run **/sbin/lilo**.

   ```
   % cp arch/i386/boot/bzImage /boot/vmlinuz-2.4.18-kaio
   % vi /etc/lilo.conf
   % /sbin/lilo
   ```

10. If you need to rebuild the HPSS Mover, make a link from **/usr/src/linux/include/linux/aio.h** to **/usr/include/linux/aio.h**.

    ```
    % cd /usr/include/linux
    % ln -s /usr/src/linux/include/linux/aio.h
    ```

3.11 Verify System Readiness

The customer is responsible for providing the following material for the *Test Readiness Review*:

- Configuration diagram(s) showing the type and layout of hardware (nodes, robots, devices, networks, etc.) and the allocation of HPSS servers and clients.

- A detailed description of the anticipated production usage of HPSS, including distribution of file sizes, data acquisition and access methods, key projects, user expectations, admin/ operations expectations, availability requirements, and performance and loading requirements.

- Detailed HPSS configuration information (Note: The lshpss.ksh script is from the deployment tools package).

  To generate the HPSS configuration information locally, issue the following on the core HPSS server node:

  ```
  % dce_login hpss_ssm -k /krb5/hpss.keytabs
  % lshpss -s sfsServer -cos -hier -sc -migp -purgep -dev -drv \
                 -svr -mvr -acct -logp -locp -ffam -bfs -ns \
                 -pvr -logd -logc -nfsd -listmeta
  ```

  To generate the HPSS configuration information, store it locally in **/var/hpss/stats/ lshpss.out**, and have it automatically emailed to IBM Houston, issue the following on the core HPSS server node:

  ```
  % lshpss.ksh
  ```

  See Section 12.2.33: *lshpss — List Information About HPSS* on page 397 of the *HPSS Management Guide* for detailed usage and sample output information for **lshpss.ksh**

  The following is a list of **lshpss** review considerations:

  ➢ Class of Service List

  ❖ File sizes - Will COS be selected by file size? If so, do the sizes presented make sense (is there overlap amongst the COS's that can be selected by file size)? Are some of the COS's so close that, all other things being equal, it would make sense to combine two or more COS's – or does one COS have such a large file size range that it would make sense to split it into two or more COS's?

  ❖ Stage Code - For the standard disk-tape hierarchy, is the stage code set to anything other than "stage on open"? If so, what are the reasons for doing so (e.g., if "no stage" a good reason is that large files will only be read back once, but don't want those to compete for new files in the disk cache)? Especially need to make sure all understand the ramifications of background or async staging.

  ❖ Flags - Normally RWA are on – if not, why? If 'F' is on, means that this COS can only be selected by explicitly naming it (not via other create-time hints – like file size) – is that desired? Verify the "enforce maximum file size" setting matches desired intent.

  ➢ Hierarchy List

  ❖ Hierarchy Layout - Verify that the hierarchy levels make sense – tape or disk only hierarchies are what is really desired. Any hierarchies with multiple levels are correctly set up (need to check migration policy to ensure that the copy count stuff is set correctly).

- ❖ Storage Class Sharing - Are any storage classes shared amongst more than one hierarchy? If so, is that intentional and if so, discuss ramification of sharing vs. not sharing (usually boils down to avoiding "pockets" of unused storage – one SC full while another is nearly empty – vs. not being able to separate one set of users or accesses from another).

➢ Disk Storage Class List

- ❖ Minimum & Maximum Storage Segment Sizes / Average Number of Segments - Verify that the values specified, reasonably balance the trade-off of causing HPSS to track too many segments (segment sizes too small or average number of segments too large) vs. wasting too much space on average per file (segment sizes too large or average number of segments too small). If a good value can not be found for the expected file size distribution to balance these two extremes, maybe we need more storage classes?

- ❖ Virtual Volume Block Size & Stripe Width - When striping is used, you need to be a little careful that these values fit well with those of any other storage class which is in the same hierarchy, as we will potentially be moving data between the levels and need to be sure that when the two sets of Movers are chatting that we don't wind up with excessive overhead, nor do we wind up with extra tape segments.

  A general rule of thumb is to have the Virtual Volume Block Size values be the same, so that with good power of two stripe widths, we wind up with a decent set of Mover to Mover connections (i.e., each Mover on one side does not have to connect to every Mover on the other). This also usually guarantees that you don't wind up with more than one Mover on one side waiting for one on the other.

  The other consideration is to be sure that the minimum disk storage segment size times 32 (this is the maximum number of source / sink descriptors per IOD – if this maximum changes, please change these guidelines) is a multiple of the tape Virtual Volume Block Size times the stripe width. If this is not true, we could wind up with extra tape storage segments for a file due to the fact that the disk side can only send 32 segments in an IOD.

- ❖ Thresholds - Verify that the threshold values seem reasonable. Compare them to the thresholds for the purge policy – will the SC thresholds be reached before purging is kicked off? If so, then at least the warning level will likely be ignored after a period of time.

- ❖ Migration & Purge Policy Ids - One thing to consider is how many tape drives will be allocated should more than one of these storage classes be migrating at once. If a large percentage of the drives – or even more drives than are physically available – are reserved for migration at a given point in time, will the retrieval response be acceptable?

➢ Tape Storage Class List

- ❖ Media block size

- ❖ Blocks Between Tape Marks - Verify that these two match the admin guide (which should actually be the default values for 4.2). If not, why not?

    - ❖ Virtual Volume Block Size and Stripe Width - See the discussion for disk Virtual Volume Block Size & Stripe Width.

    - ❖ Thresholds - Again, make sure that these seem reasonable. Do they match whatever repack or tape migration scheme (if any) of the site

    - ❖ Max VVs to Write - Does this value make sense given the number of drives available and the migration policy?

    - ❖ Migration Policy & Purge Policy Ids - Should be zero unless they are really do tape migration. Just check actual policy later when we get to it.

- ➢ Mover Device List

    - ❖ Name - For disk devices, verify that the raw/character device is being used.

      For tape devices, verify that the correct file name is being used. For AIX, the base name (e.g., "/dev/rmt1") is usually sufficient, but on Solaris and Irix care must be taken to select the correct options that are built into the name (e.g., 'v' = variable block size, 'c' = use compression, etc.).

    - ❖ Flags - Based on the admin guide, make sure the correct flags are on. Location support should be on for most supported drives (other than StorageTek 9840 on Solaris). For Ampex, "Write TM(0) to Sync" must be on. If different than admin guide, why?

- ➢ PVL Drive List - Not much here – just usually verify that the addresses seem reasonable and that the drives are unlocked (although locked usually just means they were being used for something else at the time lshpss was run…).

- ➢ Server List - Overall, just make sure there aren't extra "test" entries lying around that don't need to be. The one server whose default thread count is usually too low is the tape storage server. If it is less than the maximum number of possible requests (client I/O and/or migrations & stages depending on the COS/hierarchy configurations – need to look at the BFS server-specific configuration to get the maximum threads allowed for these activities in the BFS), then there will be the possibility of a deadlock – as all the tape SS threads could be tied up waiting for tape mounts and there is no thread to handle a tape mount notification callback from the PVL.

- ➢ Mover List

    - ❖ Buffer Size - Typically this is left at the 1MB default value. In some cases a smaller value is justified (I don't recall a site running in production with a value of less than 512KB, but it is certainly possible) if memory on the Mover nodes is tight for the anticipated workload. A larger value may be justified if required for performance – usually on the device side (I believe the Ampex drives run best with a 4 or 8MB buffer size).

    - ❖ Port Range - This is typically unused, so the values will be zero. This feature allows a site to define the TCP/IP port that the Mover will bind to on its machine for all its client communication. This is required at some sites to allow Mover traffic to be sent through a firewall, as only a set range of ports are allowed.

- ❖ Control Hostname & Data Hostname - The control hostname should reference the network interface over which the requests are sent from the SS and PVL to the Mover. It is typically an ethernet address (although this is certainly not required), since that is a reasonably low-latency network and this interface does not require very much bandwidth.

  The data hostname is used by the passive side Mover (the disk side for an intra-HPSS transfer between disk and tape, and the read side for disk-disk or tape-tape), when it creates a listen port for the actual data. This is typically a higher bandwidth interface than the control hostname – usually something like gigabit ethernet, HiPPI or the SP switch. Note that this setting has no bearing on Mover to client transfers, as the network interface on the Mover used for that transfer is solely determined by the operating system's routing table based on the client address.

- ❖ Executable Name - Verify that the named executable supports the specific options that are required for this Mover. On the device end, make sure that the correct device driver interface is enabled (e.g., _ssd for the IBM Atape driver for 3490E / 3590 / 3580, _omi for the Greshem / OMI driver for StorageTek drives, _dd2 for Ampex DST drives, _bmux for the IBM BMUX attached 3490s). On the client side, make sure that the correct client access mechanism is enabled (e.g., _gpfs for local file support).

- ❖ Encryption Key Set - This is not critical – but the encryption key should be set. This should ensure that there is proper security placed on the TCP / IP connection to the Mover (the one used by the SS and PVL).

➢ Migration Policy List

- ❖ Runtime interval - Verify that this is a sane value. In most cases, for disk migration you don't want this to be going off 5 minutes after the last run completed – nor do you likely want it not going off for 5 days. For tape migration, a larger value is likely more appropriate (but we don't have much experience here at a production site…).

- ❖ Last Read Interval (tape) - This should ideally be based on the workload – how long after a file has been read is it quite unlikely to be accessed again? Since we don't really get this level of understanding very often and since we don't have any real experience – I would hazard a guess that a much larger value than the "Last Update Interval" for disk would be appropriate.

- ❖ Last Update Interval (disk) - This should ideally be based on the workload – how long after a file has been written is it quite unlikely to be written again? Since we don't really get this level of understanding very often, some other rule of thumb is typically used – I think 30 or 60 minutes is usually reasonable. A lower value might be used if a site is concerned about getting a tape copy of the data sooner after the data is written (worst case is that we might need to re-migrate the data later – if it is subsequently updated).

- ❖ Target Free Space - If this is not 100%, why not? For a disk migration, one goal is to get the data backed up to tape so in general we should migrate every file that meets the other migration criteria.

- ❖ Copy Count & Skip Factor - These control multiple copies. Typically these are set for 1 or 2 copies on tape (with a disk level at the top of the hierarchy). If not one of these, what is the rationale?
- ❖ Request Count - Verify that the request counts (remember that more than one disk SC can use the same migration policy) do not appear to make unreasonable demands on the available tape drives – e.g., if all migrations run at once are there enough drives to service all the migrations? Given the workload is that scenario likely? In that case, what are the requirements for retrieving files that must come from tape?
- ❖ Flags - If all flags ("FWC") are not on, what is the rationale?

➢ Purge Policy List

- ❖ Last Access Interval - Again this is ideally defined by the workload – for a file on disk, how long after it has been accessed (read, most likely) is it unlikely that it will be read again anytime soon. Typically something like 60 minutes seems to be reasonable.

➢ Start & Stop Space Used Thresholds

➢ Accounting Policy - None

➢ Log Policy List - Verify logging levels are reasonable. For every server other than the Mover, that currently means "AE------ AE-". For the Mover, add 'D' to enable DEBUG logging.

➢ Location Policy List - None

➢ File Family List - None

➢ Name Server - None

➢ Bitfile Server

- ❖ Max I/O & Max Copy Requests - Verify that these values are reasonable given the anticipated workload and available resources (core server power, storage devices, etc.). As noted before, these values must be compared the thread pool size in the tape Storage Server to eliminate the possibility of a deadlock during heavy load.
- ❖ Default COS - Verify that this is a reasonable value (some sights actually use a bogus value if they elect to not allow files to be created without specifying an explicit COS).
- ❖ COS Copy to Disk - For change of COS operations, should the files be copied to the disk level of the hierarchy or the tape level? Verify the desired setting is correct.
- ❖ PVR List

        - ❖ Deferred Dismount - Is deferred dismount disabled? If so, what is the rationale? In a future release, lshpss will print the deferred dismount time – is it reasonable?
        - ❖ Device Specific A & B - Verify that these are correct for the specific PVR type (of course, it probably wouldn't run if they weren't, but…).
    - ➢ Log Daemon
        - ❖ Logfile Max Size - Many sites are now using a larger value (e.g., 50MB), which aids in running delogs over a greater time window without having to retrieve a log file that has been archived into HPSS. Using a larger size also has the benefit of archiving fewer (albeit larger) files into HPSS.
        - ❖ Archive Log File - If not set, we will not be able to recover HPSS logs for activities that happen once the online log files wrap – is there a good reason not to archive?
    - ➢ Archive COS - The COS to which the archives logs will be written. Is it reasonable? Probably do not want a tape-only COS, since that will result in a tape mount for every archived log file (most likely).
    - ➢ Log Client List
        - ❖ Logfile Max Size - A larger log file allows for easier debugging – too small a log file forces the investigator to run delogs which includes info from all nodes, which may or may not be what is desired.
        - ❖ Log To - Normally this is set to logging to a local log file and the log daemon. If different, what is the rationale?
    - ➢ NFS Daemon List - None
    - ➢ SFS Files - None

- Detailed node configuration information for all HPSS server, DCE server, and SFS server nodes (Note: The lsnode and lsnode.ksh scripts are from the deployment tools package)

  To generate the node configuration information locally, issue the following on each HPSS server, DCE server, and SFS server node:

  ```
  root% lsnode
  ```

  To generate the node configuration information, store it locally in /var/hpss/stats/lsnode.out, and have it automatically emailed to IBM Houston, issue the following on each HPSS server, DCE server, and SFS server node:

  ```
  root% lsnode.ksh
  root% more /var/hpss/stats/lsnode.out
  ```

  The following is a list of **lshpss** review considerations:

  - ◆ "no" Options - Check that they aren't opened up ridiculously large – in particular for "tcp_sendspace" and "tcp_recvspace", as these are the defaults inherited by every application that creates a internet domain socket. If there is a need for the Mover to use larger socket buffers, a better solution is typically to use the HPSS network options configuration file, which provides better granularity of control and only affects HPSS subsystems.

- **RPC_UNSUPPORTED_NETIFS -** Verify that only the desired network interfaces are being utilized by DCE.
- Tape Devices - Verify that the tape devices appear to be correct (e.g., if a 3590 is attached to an AIX machine, it should show up as a 3590, not “Other SCSI tape”). Verify that all are configured to use variable block size (if they are to be controlled by an HPSS Mover in any case).
- Filesystems - Verify the requisite file systems are mounted and have enough space - **/var/dce** on all nodes and **/var/hpss** and the filesystems for the TRB and LA files on the core server. I’ve lost track – are there others (**/var/hpss** on Mover nodes?)?
- Crontab Entries - Verify that rmxcred (all nodes) and the sfs backups (core server only) are covered. If not in crontab, then how?
- Device Location / Configuration Information - Verify that the busses are reasonably well used – e.g., one SCSI adapter isn’t connected to two tape drives while another adapters is unused.
- Tape Device Info - Verify that variable block sizes and device buffering are enabled. Compression is also usually enabled.

- All tables documenting disk and tape performance results.
- SFS configuration information (Note: The **lsencina** and **lsencina.ksh** scripts are from the deployment tools package).

  To generate the SFS configuration information locally, issue the following on each SFS server node:

  ```
  root% dce_login hpss_ssm -k /krb5/hpss.keytabs
  root% cd /opt/hpss/tools/deploy/bin
  root% lsencina > /var/hpss/stats/lsencina.out
  root% more /var/hpss/stats/lsencina.out
  ```

  To generate the SFS configuration information, store it locally in **/var/hpss/stats/lsencina.out**, and have it automatically emailed to IBM Houston, issue the following:

  ```
  root% lsencina.ksh
  root% more /var/hpss/stats/lsencina.out
  ```

- Description of SFS backup procedures, including printouts of all SFS backup configuration files and local / custom scripts. Include information such as TRB file sizes, frequency of backups, number of TRB files generated per day, number of backups retained, etc.
- Description of DCE backup procedures (including frequency)
- Disaster Recovery Plan, including:

    - ◆ Disaster recovery requirements
    - ◆ Disaster recovery test plan
- Pre-Production Test Plan, including:
    - ◆ Customer's requirements (users, admin, management, and operations staff) for the production storage system considering all involved hardware and software, which may include detailed requirements in one or more of the following areas depending on customer requirements:
        - ➢ Functionality
        - ➢ Single-transfer performance
        - ➢ Aggregate performance
        - ➢ Maximum number of concurrent requests
        - ➢ System reliability given a specific loading profile over a specific period of time
        - ➢ Interoperability with other software
    - ◆ Test procedures to verify that, when successfully executed, will prove that the above expectations will be satisfied once the entire mass storage system (software and hardware) is put into production.
- Hardware and prerequisite software support arrangements, including vendor and customer id numbers and procedures for obtaining hardware, AIX, DCE, Encina, and Sammi support.

The material can be placed on the web for easy accessibility by all review participants. The material should be made available to all reviewers at least 2 days prior to the review.

Chapter 4 HPSS Installation

4.1 Overview

This chapter provides instructions and supporting information for installing the HPSS software from the HPSS distribution media.

To install this system, we recommend that the administrator be familiar with UNIX commands and configuration, be familiar with a UNIX text editor, and have some experience with the C language and shell scripts.

Note: For information on upgrading from a previous version of HPSS, please see Chapter 14: Upgrading to HPSS Release 4.5 (page 553) in the HPSS Management Guide.

4.1.1 Distribution Media

For the current release, the HPSS software is available on one of the following distribution media:

- CD-ROM
- image file (network)

4.1.2 Installation Packages

The HPSS software is packaged into the following packages:

- Core Package - Contains all HPSS binaries
- HDM / DFS Package - Contains HPSS HDM / DFS binaries
- HDM / XFS Package - Contains HPSS HDM / XFS binaries
- Mover Package - Contains HPSS Mover binaries
- Storage Subsystem Package - Contains the HPSS Storage Subsystem (NS, BFS, SS, MPS, MM, DMG, GK, LOGC, and PFTPD) binaries
- Client API Package - Contains HPSS Client API include files and libraries

- Non-DCE Package - Contains HPSS Non-DCE Client API include files and libraries and the HPSS Non-DCE Mover binaries.
- Source Code Package - Contains the HPSS Source Code

The HPSS software package names and sizes for the supported platforms are as follows:

Table 4-1 Installation Package Sizes and Disk Requirements

Platform	HPSS Package Name	Package Size	/opt/hpss Space Requirements	Package Description
AIX	**hpss_runtime-4.5.0.0.lpp**	410 MB	**hpss.core** 210 MB	All HPSS Components
			hpss.hdm 18 MB	HDM/DFS Component
			hpss.mvr 145 MB	Mover Component
			hpss.subsy 155 MB	Storage Subsystem
			hpss.clapi 36 MB	Client API Package
	hpss_source-4.5.0.0.lpp	36 MB	**hpss.source** 61 MB	HPSS Source Code
	hpss_ndapi-4.5.0.0.lpp	11 MB	**hpss.ndapi** 16 MB	Non-DCE Package
Solaris	**HPSScore-4.5.0.0.pkg**	210 MB	**HPSScore** 210 MB	All HPSS Components
	HPSShdm-4.5.0.0.pkg	36 MB	**HPSShdm** 36 MB	HDM/DFS Component
	HPSSmvr-4.5.0.0.pkg	160 MB	**HPSSmvr** 160 MB	Mover Component
	HPSSsubsy-4.5.0.0.pkg	170 MB	**HPSSsubsy** 170 MB	Storage Subsystem
	HPSSclapi-4.5.0.0.pkg	85 MB	**HPSSclapi** 85MB	Client API Package
	HPSSsrc-4.5.0.0.pkg	60 MB	**HPSSsrc** 61 MB	HPSS Source Code
	HPSSndapi-4.5.0.0.pkg	36 MB	**HPSSndapi** 36 MB	Non-DCE Package
IRIX	**HPSSndapi-4.5.0.0.tar.Z**	6 MB	**HPSSndapi** 27MB	Non-DCE Package
Linux	**HPSShdm-4.5.0.0.rpm**	6 MB	**HPSShdm** 23 MB	HDM/XFS Package
	HPSSndapi-4.5.0.0.rpm	8 MB	**HPSSndapi** 29 MB	Non-DCE Package

4.1.3 Installation Roadmap

The steps required to install the HPSS system are listed below. Each step is discussed in more detail in the section referenced.

1. Create owner account for HPSS files (Section 4.2).
2. Installation target directory preparation (Section 4.3).
3. Install HPSS package on a node (Section 4.4).

4. Verify HPSS installed files (Section 4.5.1).

4.2 Create Owner Account for HPSS Files

The HPSS software must be installed by a **root** user. In addition, a UNIX User ID of **hpss** and Group ID of **hpss** is required for the HPSS installation process to assign the appropriate ownership for the HPSS files. If the **hpss** User ID does not exist, the installation process will fail. If the **hpss** User ID and **hpss** Group ID do not exist, create them using the normal UNIX system administration procedures.

It is very important that the HPSS files' permissions are set up properly. If they are not, HPSS may not work properly after it is configured. We recommend that the HPSS files' ownerships and permissions set by the installation process be preserved. If they must be changed, care must be taken to ensure they are changed correctly. Refer to 4.5.1 for more information on the HPSS file ownerships and permissions.

4.3 Installation Target Directory Preparation

The HPSS software is installed in the **/opt/hpss** directory. For AIX, **/usr/lpp/hpss** is also used by the SMIT utility to store the HPSS package deinstallation information. Before installing the HPSS software, make sure that the installation target directory satisfies the following conditions:

- The **/opt/hpss** and the **/usr/lpp/hpss** directories are not being used.
- The disk, where the installation target directory resides, has enough space to hold all the HPSS packages to be installed on this node.

4.4 Install HPSS Package on a Node

Installation shall be performed on each HPSS node. The appropriate HPSS software package(s) shall be installed on each HPSS node. For example, the Mover Package shall be installed on an HPSS mover node.

The option "Install HPSS Subsystem on Remote Node" in the HPSS infrastructure configuration utility, mkhpss, is no longer supported in this release.

4.4.1 AIX Installation

The HPSS software is installed in the same manner as other AIX optional or licensed program products. A detailed description can be found in the *AIX Version 4.x for RISC System/6000 Installation Guide* (SC23-2550-03), "Installing Optional Software and Service Updates" document.

4.4.1.1 Install an HPSS Package Using the installp Command

Log on to the node as **root** user and issue the **installp** command as follows to install an HPSS package (e.g., HPSS core package):

```
% installp -acgNQqwX -d \
<input device/directory>/hpss_runtime.4.5.0.0.lppimage \
-f hpss.core 2>&1
```

4.4.1.2 Install an HPSS Package Using the AIX SMIT Utility

Perform the following steps to install an HPSS package (e.g., HPSS core package) from the HPSS distribution media:

1. Log on to the node as **root** user. Enter **smitty install** and select the following options in the order shown:

   ```
   Install and Update Software
   Install and Update from LATEST Available Software
   ```

2. Enter the INPUT device / directory for software information.

3. Set the following parameter values on the screen:

   ```
   INPUT device / directory for software <data entered from step 2>
   SOFTWARE to install                                  [hpss.core]
   PREVIEW only? (install operation will NOT occur)     no
   COMMIT software updates?                             no
   SAVE replaced files?                                 yes
   AUTOMATICALLY install requisite software?            no
   EXTEND file systems if space needed?                 yes
   OVERWRITE same or newer versions?                    no
   VERIFY install and check file sizes                  no
   Include corresponding LANGUAGE filesets?             yes
   DETAILED output?                                     no
   Process multiple volumes?                            yes
   ```

4. Press **Enter** on the dialog box to begin installing the software. (Press **F3** to return to Step 3.)

4.4.2 Solaris Installation

4.4.2.1 Install an HPSS Package Using the pkgadd Command

Log on to the node as **root** user and issue the **pkgadd** command as follows to install an HPSS package (e.g., HPSS core package):

```
% pkgadd -d <input device/directory>/HPSScore-4.5.0.0.pkg HPSScore
```

4.4.3 IRIX Installation

4.4.3.1 Install the HPSSndapi Package

The HPSSndapi package is in a compressed tar file format. Perform the following steps to install this package:

1. Log on to the node as **root** user

2. Place the compressed tar file in a working directory (e.g., **/var/spool/pkg/HPSSndapi-4.5.0.0.tar.Z**)

3. Untar the package by issuing the command as follows:

   ```
   % zcat HPSSndapi-4.5.0.0.tar.Z | tar -xvf -
   ```

4. Issue the **inst** command to initiate the installation process:

   ```
   % inst -f /var/spool/pkg/HPSSndapi -r /opt/hpss
   ```

5. Issue the following commands from the **Inst Menu**:

   ```
   Inst> list HPSSndapi
   Inst> go
   Inst> quit
   ```

4.4.4 Linux Installation

4.4.4.1 Install an HPSS Package Using the rpm Command

The Linux HPSS packages are in RPM format. Log into the node as **root** and issue the **rpm** command as follows to install a package (**HPSSndapi** in this example):

```
% rpm -Uvh <input device/directory> HPSSndapi.4.5.0.0.rpm
```

4.5 Post Installation Procedures

4.5.1 Verify HPSS Installed Files

1. When the HPSS installation has completed successfully, the following HPSS file structures may be created depend on the HPSS package installed:

   /opt/hpss/bin/<HPSS binaries>

   /opt/hpss/lib/<HPSS libraries>

/opt/hpss/include/<HPSS include, idl, and tidl files>

/opt/hpss/msg/<HPSS message catalog>

/opt/hpss/tools/<HPSS tools>

/opt/hpss/man/<HPSS man pages>

/opt/hpss/config/<HPSS configuration scripts>

/opt/hpss/stk/<STK files>

/opt/hpss/sammi/hpss_ssm/<HPSS Sammi files>

/opt/hpss/src/<HPSS source files>

2. In addition, verify that the HPSS file ownerships and file permissions set by the installation process are preserved as follows:

Executable files:	`rwxr-xr-x bin  bin`
Include files:	`r--r--r-- bin  bin`
Library files:	`r--r--r-- bin  bin`
Sammi files:	`r--r----- bin  bin`
Source files:	`r--r----- hpss hpss`
Make files:	`rw-rw-r-- hpss hpss`

After the HPSS Infrastructure Configuration is performed (as described in Chapter 5: *HPSS Infrastructure Configuration* (page 215)), the permissions of the files in the following directory are changed by the **mkhpss** script as follows:

```
/opt/hpss/sammi/hpss_ssm: rw-rw---- hpss  hpss
```

The HPSS file permissions can be further restricted or relaxed according to the site's policy. However, this should be done carefully so that the new permissions will not cause problems for HPSS. Special care should be exercised when changing the permission for the Sammi files. A number of Sammi data files require read / write access by the SSM user in order for Sammi to run correctly, even though it may not be obvious that a user needs write access to a particular file. The Sammi files as installed by the HPSS installation procedure should allow read / write access to all configured SSM users. When changing the permissions set by the installation process, ensure that the new permissions will not disable part or all of the configured SSM sessions.

4.5.2 Remake HPSS

The HPSS software package includes the binaries required by HPSS and no additional "make" is needed. However, due to special circumstances, it may be necessary for a site to rebuild the binaries. If rebuilding the HPSS binaries is required, the following should be performed:

1. Construct the HPSS source tree
2. Compile the HPSS binaries

4.5.2.1 Construct the HPSS Source Tree

4.5.2.1.1 Construct the HPSS Base Source Tree

The HPSS base source tree contains the source code for all the HPSS components except the STK PVR proprietary code. To construct the HPSS base source tree, the following steps must be performed:

1. Log on as **root**.
2. Install the HPSS source code package (code is installed in the **/opt/hpss** directory).
3. Change directory to **/opt/hpss** directory.
4. Review the **/opt/hpss/Makefile.macros** file. This file defines the "make" environments and options for the HPSS software compilation. Ensure that the environments and options are specified properly before running the "make" command.
5. Issue `make idl-tidl` command.

4.5.2.1.2 Construct the HPSS HDM Component Source Tree

The HPSS HDM component source tree can be extracted from the HPSS base source tree. To construct the HPSS HDM component source tree, the following steps must be performed:

1. Log on as **root**.
2. Change directory to the HPSS base source tree (the default location is **/opt/hpss**).
3. Review the **Makefile.macros** file. Make sure that the **HDM_SUPPORT** flag is set to **on**.
   - For DFS HDMs, make sure that the **HDM_SUPPORT_TYPE** flag is set to **HDM_DFS**.
   - For XFS HDMs, make sure that the **HDM_SUPPORT_TYPE** flag is set to **HDM_XFS**.
4. Ensure that the target directory tree (where the source tree will be constructed) is empty.
5. Issue the following command:

```
% make BUILD_ROOT=<HDM source tree directory> build-hdm
```

4.5.2.1.3 Construct the HPSS Non-DCE Component Source Tree

The HPSS Non-DCE component source tree can be extracted from the HPSS base source tree. To construct the HPSS Non-DCE component source tree, the following steps must be performed:

1. Log on as **root**.
2. Change directory to the HPSS base source tree (the default location is **/opt/hpss**).
3. Review the **Makefile.macros** file.
4. Ensure that the target directory tree (where the source tree will be constructed) is empty.
5. Issue the following command:

```
% make BUILD_ROOT=<Non-DCE source tree directory> build-nodce
```

4.5.2.2 Compile the HPSS Binaries

4.5.2.2.1 Define the HPSS Shared Libraries Location

The SSM Data Server and Gatekeeper Server each links with shared libraries. This allows the shared libraries to be recompiled without causing the dynamic executables to be relinked. The pathname to the shared library that each server links with is stored in its dynamic executable file. The dynamic executable will fail to load if the pathname of the shared library has been changed since the dynamic executable was linked. The default location for all HPSS shared libraries is **/opt/hpss/lib**. Sites which need to install their shared libraries in another location will need to modify the location of the shared library stored in the dynamic executable. This can be achieved by modifying the **RUNLIBS_PATH** macro in **Makefile.macros** and then relinking the dynamic executable.

The **RUNLIBS_PATH** macro specifies the run-time top level directory of the HPSS tree. It defaults to "**/opt/hpss**". The Gatekeeper Server and SSM Data Server define their shared library locations in terms of this macro, as **$(RUNLIBS_PATH)/lib**. Sites which install their run-time HPSS tree in a location other than **/opt/hpss** should change **RUNLIBS_PATH** to reflect the name of their tree.

4.5.2.2.2 Configure the HPSS MPI-IO Component

The HPSS MPI-IO component source tree is included in the HPSS base source tree, but since this component is dependent upon a host MPI that is not platform-dependent, the default setting for **MPIO_SUPPORT** in the **Makefile.macros** file is **off**. In order to enable MPI-IO, the following steps must be performed:

Review the **Makefile.macros** file. Change **MPIO_SUPPORT** to **on** and select the appropriate **MPIO_MPI** host (**MPICH**, **IBM_POE**, or **SUN_HPC**). If no Fortran or C++ support is required, the **MPIO_FORTRAN_SUPPORT** and **MPIO_CPLUSPLUS_SUPPORT** flags should be changed to **off**.

The setting in **Makefile.macros** for **DCE_SUPPORT** affects how MPI-IO is built; the MPI-IO library must be compatible with the HPSS API, with respect to DCE support. MPI-IO is extracted with the other non-DCE sources (see Section 4.5.2.1.3) and can be built with the other non-DCE components, as desired.

4.5.2.2.3 Perform the Compilation

To compile the HPSS binaries, the following steps must be performed:

1. Log on as **root**.

2. Place the constructed HPSS source tree (i.e., HPSS base source tree, HPSS HDM source tree, or HPSS Non-DCE source tree) in the desired build directory (the default location is **/opt/hpss**).

3. Review the **Makefile.macros** file which is in the root of the source tree. This file defines the "**make**" environments and options for the HPSS software compilation. Before running the "**make**" command, ensure that the **BUILD_TOP_ROOT** macro specifies the run-time tree in which the HPSS executables and libraries will be placed.

4. Issue **make clean** command.

5. Issue **make** command.

6. If the HPSS binaries and libraries are to be installed in a location different from their build location, copy them to the install tree.

4.5.3 Set Up Sammi License Key

If Sammi is to be executed from an HPSS node, the Sammi software must be installed. The Sammi license key provided by Kinesix must be set up before Sammi can be invoked. The Sammi license key can be set up as follows:

1. Change directory to **<Sammi installation directory>/bin**:

   ```
   % cd <Sammi installation directory>/bin
   ```

2. Execute the sammi_license utility:

   ```
   % sammi_license -w
   ```

The user will be prompted for the host name where the SSM session will run and the Sammi license key. Verify that the **<Sammi installation directory>/bin/.s2_license.<hostname>** file is created with the read / write permissions for all SSM users.

Chapter 5

HPSS Infrastructure Configuration

5.1 Overview

This chapter provides instructions and supporting information for the infrastructure configuration of HPSS.

Before configuring the HPSS infrastructure, we recommend that the administrator be familiar with UNIX commands and configuration, be familiar with a UNIX text editor, and have some experience with the C language, shell scripts, DCE, and Encina.

The procedures described in this chapter assume that DCE, Encina, Sammi, Java, and HPSS have been successfully installed. In addition, a DCE / DFS cell has been fully configured.

5.1.1 Infrastructure Road Map

The steps required to configure the HPSS infrastructure are listed below. Each step is discussed in more detail in the section referenced.

1. Verify the prerequisite software (Section 5.2 below).
2. Define the HPSS environment variables (Section 5.3: *Define the HPSS Environment Variables* on page 216).
3. Configure the HPSS infrastructure on a node (Section 5.4: *Configure the HPSS Infrastructure on a Node* on page 240).

5.2 Verify the Prerequisite Software

Verify that the appropriate versions of the following prerequisites have been installed and configured in preparation for the HPSS infrastructure configuration (For versions, see Section 2.3: *Prerequisite Software Considerations* on page 46):

- DCE / DFS

- WebSphere (Encina TxSeries)
- Sammi
- Java
- DCE / DFS server / client machine(s)
- HPSS software

Refer to the *DCE Version 3.1 Administration Guide* and the *Encina Server Administration: System Administrator's Guide and Reference* for more information on installing and configuring DCE and Encina.

Refer to Section 3.8: *Install and Configure Java and hpssadm* on page 175 for more information on installing Java 1.3.

5.3 Define the HPSS Environment Variables

The HPSS environment variables are defined in two files: The **/opt/hpss/config/hpss_env.default** file defines environment variables that may need to be changed by the administrator. The **/opt/hpss/include/hpss_env_defs.h** file defines environment variables that most likely do not need to be modified. Before running the **mkhpss** script on a node, copy the **hpss_env.default** file to **hpss_env.** Review and edit it to ensure that the variables reflect the local environment. To override an environment variable defined in the **hpss_env_defs.h** file, redefine the environment variable in the **hpss_env** file.

Section 5.3.1: *hpss_env.default* on page 216 and Section 5.3.2: *hpss_env_defs.h* on page 221 list the verbatim contents of these two files.

The utility **/opt/hpss/config/hpss_set_env** is provided to help managing the HPSS environment variables.

Usage:

```
   hpss_set_env [-all] [-def] [-set] | ENVNAME
```

Where:

```
      -all        Show current HPSS environment values
      -def        Show all HPSS default environment values
      -set        Set HPSS default environment values
      ENVNAME     Display current value for ENVNAME
```

5.3.1 hpss_env.default

The following is a verbatim listing of the **hpss_env.default** file:

```
#!/bin/ksh
# static char SccsId[] = “ @(#)34   3.42   config/hpss_env.default, gen, 4.5   8/21/01
14:36:40”;
```

```
#===============================================================================
#
#  Name: hpss_env - Site defined HPSS global variables/environment shell script
#
#  Synopsis: hpss_env
#
#  Arguments: none
#
#  Outputs:
#                    - HPSS global variable definitions and environment parameters
#
#  Description: This script defines the HPSS global variables and environment
#               parameters which override the values specified in the
#               ./include/hpss_env_defs.h file.
#
#  Language: Korn shell
#
#  Interfaces:
#                    - All HPSS subsystems
#
#  Resources used:
#                    N/A
#
#  Limitations:
#                    - Environment variables in this shell script must be
#           modified to reflect any deviation.
#
#  Assumptions:
#                    none
#
#  Traceability:
#                    Version   Date       Description
#                    _______   ________   _______________________________________________
#                      1.1     08/30/93   Initial version
#                    1.2-1.5    03/08/95   DCE 1.3 changes
#         1.8 -     04/20/95   IT Cleaned up and
#         1.12      07/06/95   updated descriptions
#         3.1       08/08/95   multiple platform support
#         3.2       08/30/95   multiple hpss support
#         3.3       10/23/95   gen0038
#         3.4       11/21/95   r3 pftpd changes
#         3.5       02/16/96   r3 clean up
#         3.6       02/19/96   r3 clean up
#         3.7       03/06/96   move subsys tar file generation to remote install
#         3.8-3.11  06/19/96   Add SSM and other HPSS variables
#         3.12      06/27/96   Change HPSSLOG_SHMKEY to match default of 2801
#         3.13      07/02/96   Use HPSS_ROOT environ. variable if set
#         3.14      07/25/96   Set null value to be ""
#         3.15      08/07/96   Add LANG variable
#         3.16      07/17/97   Add HPSS_PATH_ETC variable
#         3.17      07/23/97   Release 3.2 updates (gen0097)
#         3.18      12/17/97   Mod sfs server default
#                    3.19     02/26/98   Added HPSS_CDS_LS and HPSS_LS_NAME. (ls0002)
#                    3.20     02/27/98   Added HPSS_PRINCIPAL_{DMG,LS} (ls0002)
#                    3.21     03/19/98   Rename $HPSS_CDS_CNS to $HPSS_CDS_NS
#                    3.22      04/27/98   Added numerous variables for storage servers
#         3.23      07/20/98   Added HPSS_PATH_HDM (dfs0010)
#         3.24      10/09/98   Added other HDM variables (dfs0029)
#         3.25      11/13/98   4v1 default variable changes
#         3.26      11/18/98   4v1 default encina variable changes
#         3.27      11/18/98   4v1 default keytab variable changes
#         3.28      11/29/98   Move most of the default var's to
#                              ./include/hpss_env_defs.h
#         3.29      11/30/98   Restore HPSSLOG variable
#         3.30      07/01/99   Added conditions for SunOS
#         3.31      03/03/00   Added variables for SSM command line
#         3.32      03/17/00   Corrections to variables for SSM command line
#         3.33      03/24/00   Comments for SSM command line, LD_LIBRARY_PATH
#         3.34      06/30/00   Move SCCS line to the second line (1923)
#         3.35      09/14/00   Java 1.3.0 mods (1990)
#         3.36      09/15/00   Use /opt for install path
#         3.37      10/15/00   Remove HPSS_SFS_VOL
#         3.38      10/30/00   Put in JAVA_ROOT, JAVA_BIN (2198)
#         3.39      11/22/00   Put in JAVA_SUPPORT (2058)
#         3.40      11/29/00   Set JAVA_ROOT based on platform (2261)
#         3.41      07/11/01   Remove JAVA_SUPPORT flag (2442)
#         3.42      08/21/01   Update HPSS_PATH_SAMMI_INSTALL (2458)
#
#  Notes:
#        Licensed Materials
#
#        Copyright (C) 1992-2001 International Business Machines Corporation,
#        The Regents of the University of California, Sandia Corporation, and
#        UT-Battelle.
#
#        All rights reserved.
#
#        Portions of this work were produced by the University of California,
#        Lawrence Livermore National Laboratory (LLNL) under Contract No.
#        W-7405-ENG-48 with the U.S. Department of Energy (DOE), by the
#        University of California, Lawrence Berkeley National Laboratory (LBNL)
#        under Contract No. DEAC03776SF00098 with DOE, by the University of
#        California, Los Alamos National Laboratory (LANL) under Contract No.
#        W-7405-ENG-36 with DOE, by Sandia Corporation, Sandia National
#        Laboratories (SNL) under Contract No. DEAC0494AL85000 with DOE, and
#        UT-Battelle, Oak Ridge National Laboratory (ORNL) under Contract No.
#        DE-AC05-96OR22464 with DOE.  The U.S. Government has certain reserved
#        rights under its prime contracts with the Laboratories.
#
#                                DISCLAIMER
#        Portions of this software were sponsored by an agency of the United
#        States Government.  Neither the United States, DOE, The Regents of the
#        University of California, Sandia Corporation, UT-Battelle, nor any of
#        their employees, makes any warranty, express or implied, or assumes
#        any 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.
#
#        High Performance Storage System is a registered trademark of
#        International Business Machines Corporation.
#
#================================================================================

#================================================================================
# Describe Global/Environment Variables
#================================================================================
#
#  Variable                Description
#  ============================================================================
#
```

```
#  D i r e c t o r i e s
#
#  HPSS_PATH                      Pathname where HPSS top level is
#
#  S A M M I   D i r e c t o r i e s
#
#  HPSS_PATH_SAMMI_INSTALLPathname where SAMMI is installed
#
#  S y s t e m / U s e r   I n f o r m a t i o n
#
#  HPSS_SYSTEM                    System platform name
#  HPSS_SYSTEM_VERSION     System version
#  HPSS_HOST                      Host name
#  HPSS_HOST_FULL_NAME     Fully qualified Host domain name
#
#  D C E   V a r i a b l e s
#
#  HPSS_CELL_ADMIN         Principal name for administrating hpss in a cell
#  HPSS_CDS_PREFIX         CDS prefix for HPSS servers
#  HPSS_CDS_HOST           CDS host name
#
#  E n c i n a   V a r i a b l e s
#
#  HPSS_SFS_ADMIN          Principal name for administrating hpss
#                                 in Encina SFS
#  HPSS_SFS_SERVER         Encina SFS server name with CDS prefix
#
#  J A V A   V a r i a b l e s
#
#  JAVA_ROOT                      Top level directory where Java is installed.
#  JAVA_HOME                      Top level directory of the Java Runtime.  If
#                                        only the JRE is installed, then JAVA_HOME is
#                                        the same as JAVA_ROOT.  If the entire SDK (Java
#                                        Software Development Kit) is installed, then
#                                        the JRE is a subset of this and JAVA_HOME is
#                                        ${JAVA_ROOT}/jre.
#
#  M i s c e l l a n e o u s
#
#  HPSSLOG                        HPSS logging output destination [stdout | NULL]
#
#================================================================================
# Set HPSS Global/Environment Variables
#================================================================================

#  D i r e c t o r i e s . . .
#
export HPSS_PATH=${HPSS_ROOT:-/opt/hpss}


#  S A M M I   D i r e c t o r i e s
#
export HPSS_PATH_SAMMI_INSTALL=”/usr/local/sammi”


#  S y s t e m / U s e r   I n f o r m a t i o n . . .
#
export HPSS_SYSTEM=$(uname)
export HPSS_HOST=$(hostname)
if [[ $HPSS_SYSTEM = SunOS ]]; then
```

```
   export HPSS_SYSTEM_VERSION=$(uname -r)
   export HPSS_HOST_FULL_NAME=$(hostname)
else
   export HPSS_SYSTEM_VERSION=$(oslevel)
   export HPSS_HOST_FULL_NAME=`host $HPSS_HOST |cut -f1 -d' '`
fi



#  D C E   V a r i a b l e s . . .
#
export HPSS_CELL_ADMIN="cell_admin"
export HPSS_CDS_PREFIX=/.:/hpss
export HPSS_CDS_HOST=$HPSS_HOST


#  E n c i n a   V a r i a b l e s . . .
#
export HPSS_SFS_ADMIN="encina_admin"
export HPSS_SFS_SERVER=/.:/encina/sfs/hpss

#  J A V A   V a r i a b l e s . . .
#
if [[ $HPSS_SYSTEM = SunOS ]]; then
    export JAVA_ROOT="/usr/j2se"
    export JAVA_HOME="${JAVA_ROOT}/jre"
else
    export JAVA_ROOT="/usr/java130"
    export JAVA_HOME="${JAVA_ROOT}/jre"
fi

#  M i s c e l l a n e o u s . . .
#
export HPSSLOG=""
###export HPSSLOG="stdout"

#===============================================================================
# Override HPSS Global/Environment Variables Defined in
# ./include/hpss_env_defs.h
#
# For example:  To assign an UID to the HPSS BFS principal
#
#              export HPSS_PRINCIPAL_BFS_UID="-uid 1234"
#
#===============================================================================




#===============================================================================
# Pickup HPSS Default Variables defined from ./include/hpss_env_defs.h
#===============================================================================

$HPSS_PATH/config/hpss_set_env -set > /tmp/tmp.$$; . /tmp/tmp.$$; rm /tmp/tmp.$$
```

5.3.2 *hpss_env_defs.h*

The following is a verbatim listing of the **hpss_env_defs.h** file:

```
/* static char SccsId[] = " @(#)71   1.44   include/hpss_env_defs.h, gen, 4.5   4/29/02
12:28:46"; */
/*==============================================================================
 *
 * Include Name:  hpss_env_defs.h
 *
 * Description:   Contains default definitions for HPSS environment variables.
 *                           These are used by the mm_GetEnv function to set defaults
 *                           when the environment varibale does not exist.
 *
 * Traceability:
 *      Vers         Author Date          Description
 *      ----         ------ ---------------------------------------------
 *                   1.1    dus    01/07/98mm0055: Initial R4.1 version
 *                   1.2    dus    01/14/98mm0055: Added $
 *                   1.3    dus    02/05/98mm0055: Add (${HPSS_HOST}) to DescNames
 *                                                     for logc, hpssd and mover
 *                   1.4    dus    02/06/98mm0055: Add HPSS_ALARMS_SSMDS
 *                   1.5    bartp  02/26/98ls0002: Renamed HPSS_LS_GROUP to
 *                                                      HPSS_LS_NAME.
 *                   1.6    dus    03/04/98ssm0280: Add remaining env for SM
 *                   1.7    dus    03/18/98mm0055: Update to include old names
 *                   1.8    dus    04/06/98hpss0128: Update NS parameters
 *                   1.9    dus    04/28/98hpss0137: Add File Family
 *      1.10    kmt     08/24/98        ssm0309: Add support for AML PVR,
 *                                               Shelf Tape & NDCG
 *                   1.11   dus    11/02/98gen0111: Add HPSS_PRINCIPAL_CLIENT_API
 *                                                      update copyright
 *                   1.12    pal     11/30/98        mps0113: Change HPSS_UNIX_MPS_REPORT
 *                   1.13   dus    12/03/98gen0111: Add HPSS_CONFIG_ACCOUNTNG
 *                   1.14   pal    01/07/99gen01??: Replicate 4v1 changes to 4va
 *      1.15    JZL     09/07/99        sud0024:
 *                                        Add HPSS_PATH_ADM,HPSS_PATH_CORE
 *      1.16    kwe     10/05/99        1696: Move core directory under adm.
 *      1.17    guidryg 11/09/99        4.2 changes
 *      1.18    guidryg 11/09/99        Typo in global config specification.
 *      1.19    debbiem 12/06/99        Add GK Unix file.
 *              guidryg                 Add storage class thresholds
 *                                      Fix HPSS server prefix
 *      1.20    guidryg 01/27/00        Remove mountd config variables
 *                   1.21    bartp03/02/00Added acct validation var
 *                   1.22    vyw03/03/00Initial SSM command line checkin
 *                   1.23    guidryg03/06/00Reorg to show which are subsystem-
 *                                              specific
 *                   1.24    vyw03/07/00ssm1847: JAVA_CONFIG, closed quotes
 *                   1.25    vyw03/07/00ssm1847: JAVA_CONFIG to new default
 *                   1.26    guidryg03/07/00Fixed quoting problem in data server
 *                                              section
 *                   1.27    vyw03/17/00ssm1847: additional SSMDS variables
 *                   1.28    ctnguyen03/22/00ssm1847: add jar files
 *                   1.29    vyw     03/24/00ssm1847: add HPSS_SSMDS_RMI_HOST
 *                   1.30    ctnguyen04/04/00ssm1847: add JAVA_SUPPORT
 *      1.31    ctnguyen04/12/00ssm1843: set rmi default port to 1066
 *      1.32    shreyas 04/26/00        Added NDAPI env defaults
 *      1.33    vyw     09/14/00        SSMDS Java 1.3 and security (1990);
 *                                              also changed HPSS_ROOT to /opt/hpss
 *      1.34    guidryg 09/15/00        Use /opt for install.
 *      1.35    ctnguyen10/09/00        Add ENCINA_LOCAL and ENCINA_MIRROR.
 *      1.36    ctnguyen11/13/00        Add HPSS_PATH_SSM.
 *      1.37    shreyas 11/27/00        change HPSS_NDCG_KRB5_SERVICENAME
 *      1.39    JAD     01/25/01        1971 - Additions to support RAIT.
 *                  1.40    HDJ/WHR 02/16/01        2304: Check in NFS V3 code
 *      1.41    shreyas 03/02/01        2313 - LTO mods
 *      1.42    vyw     07/11/01        2442: remove JAVA_SUPPORT flag
 *      1.43    pnc     08/21/01        2458: Remove remote install parms
 *      1.44    jls     04/15/02        2653: Add HPSS_NFS_DISABLE_JUNCTIONS &
 *                                                     HPSS_MNT_DISABLE_JUNCTIONS
 *
 *  Notes:
 *
 *       Licensed Materials
 *
 *       Copyright (C) 1992-2001 International Business Machines Corporation,
 *       The Regents of the University of California, Sandia Corporation, and
 *       UT-Battelle.
 *
 *       All rights reserved.
 *
 *       Portions of this work were produced by the University of California,
 *       Lawrence Livermore National Laboratory (LLNL) under Contract No.
 *       W-7405-ENG-48 with the U.S. Department of Energy (DOE), by the
 *       University of California, Lawrence Berkeley National Laboratory (LBNL)
 *       under Contract No. DEAC03776SF00098 with DOE, by the University of
 *       California, Los Alamos National Laboratory (LANL) under Contract No.
 *       W-7405-ENG-36 with DOE, by Sandia Corporation, Sandia National
 *       Laboratories (SNL) under Contract No. DEAC0494AL85000 with DOE, and
 *       UT-Battelle, Oak Ridge National Laboratory (ORNL) under Contract No.
 *       DE-AC05-96OR22464 with DOE.  The U.S. Government has certain reserved
 *       rights under its prime contracts with the Laboratories.
 *
 *                                 DISCLAIMER
 *       Portions of this software were sponsored by an agency of the United
 *       States Government.  Neither the United States, DOE, The Regents of the
 *       University of California, Sandia Corporation, UT-Battelle, nor any of
 *       their employees, makes any warranty, express or implied, or assumes
 *       any 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.
 *
 *       High Performance Storage System is a registered trademark of
 *       International Business Machines Corporation.
 *
 *---------------------------------------------------------------------------*/

#ifndef mm_config_defs_h
#define mm_config_defs_h
#define HPSS_ENV_FILE

/*
 *************************************************************************
 * The env_t structure contains 2 fields;
 *                  name   - name of environment variable
 *                  def    - default for when name is not defined in the environment
 *                  value  - the expanded value for the environment variable
 ************************************************************************
 */

typedef struct env {
        char *name;
                    char *def;
                    char *value;
} env_t;

static env_t   hpss_env_defs[] = {

/*
 ************************************************************************
 *                  HPSS_ROOT     - Root pathname for HPSS Unix top level
 *                  HPSS_HOST     - Machine host name
 *                  HPSS_KEYTAB_FILE_SERVER- Fully qualified DCE HPSS server keytab file
 *      HPSS_KEYTAB_FILE_CLIENT - Fully qualified DCE HPSS client keytab file
 *                  HPSS_PATH     - Pathname for HPSS Unix top level
 *                  HPSS_PATH_BIN - Pathname for HPSS executables
 *                  HPSS_PATH_SLASH_BIN- Pathname for /bin
 *      HPSS_PATH_SLASH_ETC     - Pathname for /etc
 *      HPSS_PATH_USR_BIN       - Pathname for /usr/bin
 *      HPSS_PATH_USR_SBIN      - Pathname for /usr/sbin
 *                  HPSS_PATH_VAR - Pathname for HPSS var directory
 *                  HPSS_USER     - HPSS user name
 *                  HPSS_USERROOT - Root user id
 *      HPSS_PATH_SAMMI_BIN     - Pathname for SAMMI bin
 *      HPSS_PATH_SAMMI_DATA    - Pathname for SAMMI data
 ************************************************************************
 */
                    { “HPSS_ROOT”,      “/opt/hpss”                    },
                    { “HPSS_HOST”,      “%H”                    },
                    { “HPSS_KEYTAB_FILE_SERVER”,“/krb5/hpss.keytabs”},
                    { “HPSS_KEYTAB_FILE_CLIENT”,“/krb5/hpssclient.keytab”},
                    { “HPSS_PATH”,      “${HPSS_ROOT}”      },
                    { “HPSS_PATH_BIN”,   “${HPSS_PATH}/bin”},
                    { “HPSS_PATH_SLASH_BIN”,“/bin”    },
                    { “HPSS_PATH_SLASH_ETC”,        “/etc”                     },
                    { “HPSS_PATH_USR_BIN”,          “/usr/bin”                 },
                    { “HPSS_PATH_USR_SBIN”,         “/usr/sbin”                },
                    { “HPSS_PATH_VAR”,“/var/hpss”     },
                    { “HPSS_USER”,      “hpss”                  },
                    { “HPSS_USERROOT”,“root”               },
                    { “HPSS_PATH_SAMMI_BIN”,        “$HPSS_PATH_SAMMI_INSTALL/bin”  },
                    { “HPSS_PATH_SAMMI_DATA”,       “$HPSS_PATH_SAMMI_INSTALL/data” },
/*
 ************************************************************************
 * HPSS DCE Group names
 *      HPSS_GRP_NAME             - HPSS group name
 *      HPSS_GRP_NAME_SERVER      - HPSS Server group name
 *      HPSS_GRP_NAME_CLIENT      - HPSS Client group name
 ************************************************************************
 */
                    { “HPSS_GRP_NAME”,              “hpss”                     },
                    { “HPSS_GRP_NAME_SERVER”,       “hpss_server”              },
                    { “HPSS_GRP_NAME_CLIENT”,       “hpss_client”              },
/*
 ************************************************************************
 * HPSS Server DCE Principal names
 *
 *                   HPSS_PRINCIPAL- DCE Principal name for HSEC Server
 *                   HPSS_PRINCIPAL_BFS- DCE Principal name for Bitfile Server
 *                   HPSS_PRINCIPAL_CLIENT_API
 *                                       - DCE Principal name for Client API
 *                   HPSS_PRINCIPAL_DMG- DCE Principal name for DMAP Gateway
 *                   HPSS_PRINCIPAL_FTPD- DCE Principal name for FTP Daemon
 *                   HPSS_PRINCIPAL_GK- DCE Principal name for Gatekeeper Server
 *                   HPSS_PRINCIPAL_HPSSD- DCE Principal name for Startup Daemon
 *                   HPSS_PRINCIPAL_LOG- DCE Principal name for Log Client and Daemon
 *                   HPSS_PRINCIPAL_LS- DCE Principal name for Location Server
 *                   HPSS_PRINCIPAL_MM- DCE Principal name for Metadata Monitor
 *                   HPSS_PRINCIPAL_MOUNTD- DCE Principal name for Mount Daemon
 *                   HPSS_PRINCIPAL_MPS- DCE Principal name for Migration/Purge Server
 *                   HPSS_PRINCIPAL_MVR- DCE Principal name for Mover
 *                   HPSS_PRINCIPAL_NDCG- DCE Principal name for Non-DCE Gateway
 *                   HPSS_PRINCIPAL_NFSD- DCE Principal name for NFS Daemon
 *                   HPSS_PRINCIPAL_NS- DCE Principal name for Name Server
 *                   HPSS_PRINCIPAL_PFSD- DCE Principal name for PFS Daemon
 *                   HPSS_PRINCIPAL_PVL- DCE Principal name for PVL
 *                   HPSS_PRINCIPAL_PVR- DCE Principal name for PVR
 *                   HPSS_PRINCIPAL_SS- DCE Principal name for Storage Server
 *                   HPSS_PRINCIPAL_SSM- DCE Principal name for SSM
 ***************************************************************************
 */
                     { “HPSS_PRINCIPAL”,NULL,                 },
                     { “HPSS_PRINCIPAL_BFS”,“hpss_bfs”,},
                     { “HPSS_PRINCIPAL_CLIENT_API”,“hpss_client_api”,},
                     { “HPSS_PRINCIPAL_DMG”,“hpss_dmg”,},
                     { “HPSS_PRINCIPAL_FTPD”,“hpss_ftp”,},
                     { “HPSS_PRINCIPAL_GK”,“hpss_gk”,  },
                     { “HPSS_PRINCIPAL_HPSSD”,“hpss_hpssd”,},
                     { “HPSS_PRINCIPAL_LOG”,“hpss_log”,},
                     { “HPSS_PRINCIPAL_LS”,“hpss_ls”,  },
                     { “HPSS_PRINCIPAL_MM”,“hpss_mmon”,},
                     { “HPSS_PRINCIPAL_MOUNTD”,“hpss_mountd”,},
                     { “HPSS_PRINCIPAL_MPS”,“hpss_mps”,},
                     { “HPSS_PRINCIPAL_MVR”,“hpss_mvr”,},
                     { “HPSS_PRINCIPAL_NDCG”,“hpss_ndcg”,},
                     { “HPSS_PRINCIPAL_NFSD”,“hpss_nfs”,},
                     { “HPSS_PRINCIPAL_NS”,“hpss_cns”, },
                     { “HPSS_PRINCIPAL_PFSD”,“hpss_piofsie”,},
                     { “HPSS_PRINCIPAL_PVL”,“hpss_pvl”,},
                     { “HPSS_PRINCIPAL_PVR”,“hpss_pvr”,},
                     { “HPSS_PRINCIPAL_SS”,“hpss_ss”,  },
                     { “HPSS_PRINCIPAL_SSM”,“hpss_ssm”,},
/*
 ***************************************************************************
 * HPSS Server DCE Principal UID’s
 *
 *      HPSS_PRINCIPAL_BFS_UID      - DCE Principal UID for Bitfile Server
 *      HPSS_PRINCIPAL_CLIENT_API_UID
 *                                  - DCE Principal UID for Client API
 *      HPSS_PRINCIPAL_DMG_UID      - DCE Principal UID for DMAP Gateway
 *      HPSS_PRINCIPAL_FTPD_UID     - DCE Principal UID for FTP Daemon
 *      HPSS_PRINCIPAL_GK_UID       - DCE Principal UID for Gatekeeper Server
 *      HPSS_PRINCIPAL_HPSSD_UID    - DCE Principal UID for Startup Daemon
 *      HPSS_PRINCIPAL_LOG_UID      - DCE Principal UID for Log Client and Daemon
 *      HPSS_PRINCIPAL_LS_UID       - DCE Principal UID for Location Server
 *      HPSS_PRINCIPAL_MM_UID       - DCE Principal UID for Metadata Monitor
 *      HPSS_PRINCIPAL_MOUNTD_UID   - DCE Principal UID for Mount Daemon
 *      HPSS_PRINCIPAL_MPS_UID      - DCE Principal UID for Migration/Purge Server
 *      HPSS_PRINCIPAL_NDCG_UID     - DCE Principal UID for Non-DCE Gateway
 *      HPSS_PRINCIPAL_MVR_UID      - DCE Principal UID for Mover
 *      HPSS_PRINCIPAL_NFSD_UID     - DCE Principal UID for NFS Daemon
 *      HPSS_PRINCIPAL_NS_UID       - DCE Principal UID for Name Server
 *      HPSS_PRINCIPAL_PFSD_UID     - DCE Principal UID for PFS Daemon
 *      HPSS_PRINCIPAL_PVL_UID      - DCE Principal UID for PVL
 *      HPSS_PRINCIPAL_PVR_UID      - DCE Principal UID for PVR
 *      HPSS_PRINCIPAL_SS_UID       - DCE Principal UID for Storage Server
 *      HPSS_PRINCIPAL_SSM_UID      - DCE Principal UID for SSM
 *
 * NOTE: Principal UID must be in the format of “-uid <number of uid>”
 *       For example:
 *      { “HPSS_PRINCIPAL_BFS_UID”          “-uid 1234”,             },
 *
 ***************************************************************************
 */
                   { “HPSS_PRINCIPAL_BFS_UID”,         “”,                 },
                   { “HPSS_PRINCIPAL_CLIENT_API_UID”,  “”,                 },
                   { “HPSS_PRINCIPAL_DMG_UID”,         “”,                 },
                   { “HPSS_PRINCIPAL_FTPD_UID”,        “”,                 },
                   { “HPSS_PRINCIPAL_GK_UID”,          “”,                 },
                   { “HPSS_PRINCIPAL_HPSSD_UID”,       “”,                 },
                   { “HPSS_PRINCIPAL_LOG_UID”,         “”,                 },
                   { “HPSS_PRINCIPAL_LS_UID”,          “”,                 },
                   { “HPSS_PRINCIPAL_MM_UID”,          “”,                 },
                   { “HPSS_PRINCIPAL_MOUNTD_UID”,      “”,                 },
                   { “HPSS_PRINCIPAL_MPS_UID”,         “”,                 },
                   { “HPSS_PRINCIPAL_MVR_UID”,         “”,                 },
                   { “HPSS_PRINCIPAL_NDCG_UID”,        “”,                 },
                   { “HPSS_PRINCIPAL_NFSD_UID”,        “”,                 },
                   { “HPSS_PRINCIPAL_NS_UID”,          “”,                 },
                   { “HPSS_PRINCIPAL_PFSD_UID”,        “”,                 },
                   { “HPSS_PRINCIPAL_PVL_UID”,         “”,                 },
                   { “HPSS_PRINCIPAL_PVR_UID”,         “”,                 },
                   { “HPSS_PRINCIPAL_SS_UID”,          “”,                 },
                   { “HPSS_PRINCIPAL_SSM_UID”,         “”,                 },
/*
 ***************************************************************************
 * HPSS Server Executable Names
 *
 *                  HPSS_EXEC_ACCT- executable name for Accounting
 *                  HPSS_EXEC_BFS - executable name for Bitfile Server
 *                  HPSS_EXEC_DMG - executable name for DMAP Gateway
 *                  HPSS_EXEC_FTPD- executable name for FTPD
 *                  HPSS_EXEC_GK  - executable name for Gatekeeper Server
 *                  HPSS_EXEC_HPSSD- executable name for Start Daemon
 *                  HPSS_EXEC_LOGC- executable name for Log Client
 *                  HPSS_EXEC_LOGD- executable name for Log Daemon
 *                  HPSS_EXEC_LS  - executable name for Location Server
 *                  HPSS_EXEC_MM  - executable name for Metadata Monitor
 *                  HPSS_EXEC_MOUNTD- executable name for Mount Daemon
 *                  HPSS_EXEC_MPS - executable name for Migration/Purge Server
 *                  HPSS_EXEC_MVR - executable name for Mover
 *                  HPSS_EXEC_MVR_TCP- executable name for Mover TCP
 *                  HPSS_EXEC_NDCG- executable name for Non-DCE Gateway
 *                  HPSS_EXEC_NFSD- executable name for NFS Daemon
 *                  HPSS_EXEC_NS  - executable name for Name Server
 *                  HPSS_EXEC_PFSD- executable name for PFS Daemon
 *                  HPSS_EXEC_PVL - executable name for PVL
 *                  HPSS_EXEC_PVR_AMPEX- executable name for PVR Ampex
 *                  HPSS_EXEC_PVR_OPER- executable name for PVR Operator
 *                  HPSS_EXEC_PVR_STK- executable name for PVR STK
 *                  HPSS_EXEC_PVR_3494- executable name for PVR 3494
 *                  HPSS_EXEC_PVR_3495- executable name for PVR 3495
 *      HPSS_EXEC_PVR_LTO- executable name for PVR LTO
 *      HPSS_EXEC_PVR_AML       - executable name for PVR AML
 *                  HPSS_EXEC_SSDISK- executable name for Storage Server - Disk
 *                  HPSS_EXEC_SSTAPE- executable name for Storage Server - Tape
 *                  HPSS_EXEC_SSMDS- executable name for SSM Data Server
 *                  HPSS_EXEC_SSMSM- executable name for SSM Storage Manager
 *
 ***************************************************************************
 */
                    { "HPSS_EXEC_ACCT","${HPSS_PATH_BIN}/hpss_acct"},
                    { "HPSS_EXEC_BFS","${HPSS_PATH_BIN}/hpss_bfs"},
                    { "HPSS_EXEC_DMG","${HPSS_PATH_BIN}/hpss_dmg"},
                    { "HPSS_EXEC_FTPD","${HPSS_PATH_BIN}/hpss_ftp"},
                    { "HPSS_EXEC_GK","${HPSS_PATH_BIN}/hpss_gk"},
                    { "HPSS_EXEC_HPSSD","${HPSS_PATH_BIN}/hpssd"},
                    { "HPSS_EXEC_LOGC","${HPSS_PATH_BIN}/hpss_logc"},
                    { "HPSS_EXEC_LOGD","${HPSS_PATH_BIN}/hpss_logd"},
                    { "HPSS_EXEC_LS","${HPSS_PATH_BIN}/hpss_ls"},
                    { "HPSS_EXEC_MM","${HPSS_PATH_BIN}/hpss_mmon"},
                    { "HPSS_EXEC_MOUNTD","${HPSS_PATH_BIN}/hpss_mnt"},
                    { "HPSS_EXEC_MPS","${HPSS_PATH_BIN}/hpss_mps"},
                    { "HPSS_EXEC_MVR","${HPSS_PATH_BIN}/hpss_mvr"},
                    { "HPSS_EXEC_MVR_TCP","${HPSS_PATH_BIN}/hpss_mvr_tcp"},
                    { "HPSS_EXEC_NDCG","${HPSS_PATH_BIN}/hpss_ndcg"},
                    { "HPSS_EXEC_NFSD","${HPSS_PATH_BIN}/hpss_nfs"},
                    { "HPSS_EXEC_NS","${HPSS_PATH_BIN}/hpss_cns"},
                    { "HPSS_EXEC_PFSD","${HPSS_PATH_BIN}/hpss_piofsie"},
                    { "HPSS_EXEC_PVL","${HPSS_PATH_BIN}/hpss_pvl"},
                    { "HPSS_EXEC_PVR_AMPEX","${HPSS_PATH_BIN}/hpss_pvr_ampex" },
                    { "HPSS_EXEC_PVR_OPER","${HPSS_PATH_BIN}/hpss_pvr_operator" },
                    { "HPSS_EXEC_PVR_STK","${HPSS_PATH_BIN}/hpss_pvr_stk"},
                    { "HPSS_EXEC_PVR_3494","${HPSS_PATH_BIN}/hpss_pvr_3494"},
                    { "HPSS_EXEC_PVR_3495","${HPSS_PATH_BIN}/hpss_pvr_3495"},
                    { "HPSS_EXEC_PVR_LTO","${HPSS_PATH_BIN}/hpss_pvr_lto"},
                    { "HPSS_EXEC_PVR_AML","${HPSS_PATH_BIN}/hpss_pvr_aml"},
                    { "HPSS_EXEC_SSDISK","${HPSS_PATH_BIN}/hpss_ss_disk"},
                    { "HPSS_EXEC_SSTAPE","${HPSS_PATH_BIN}/hpss_ss_tape"},
                    { "HPSS_EXEC_SSMDS","${HPSS_PATH_BIN}/hpss_ssmds"},
                    { "HPSS_EXEC_SSMSM","${HPSS_PATH_BIN}/hpss_ssmsm"},
/*
 ***************************************************************************
 * Utilities need by SSM
 *
 *                  HPSS_EXEC_ACL_EDIT- Pathname for the acl_edit utility
 *                  HPSS_EXEC_CDSCP      - Pathname for the cdscp utility
 *                  HPSS_EXEC_DELOG      - Pathname for the delog utility
 *                  HPSS_EXEC_RECLAIM- Pathname for the reclaim utility
 *                  HPSS_EXEC_REPACK- Pathname for the repack utility
 *
 ***************************************************************************
 */
                    { "HPSS_EXEC_ACL_EDIT","${HPSS_PATH_SLASH_BIN}/acl_edit" },
                    { "HPSS_EXEC_CDSCP","${HPSS_PATH_SLASH_BIN}/cdscp" },
                    { "HPSS_EXEC_DELOG","${HPSS_PATH_BIN}/hpss_delog" },
                    { "HPSS_EXEC_RECLAIM","${HPSS_PATH_BIN}/reclaim.ksh" },
                    { “HPSS_EXEC_REPACK”,“${HPSS_PATH_BIN}/repack” },
/*
 ***************************************************************************
 * Logging Unix files
 *
 *                  HPSS_PATH_LOG       - unix path name for logging files
 *                  HPSS_UNIX_LOCAL_LOG- local log file
 ***************************************************************************
 */
                    { “HPSS_PATH_LOG”,“${HPSS_PATH_VAR}/log”},
                    { “HPSS_UNIX_LOCAL_LOG”,“${HPSS_PATH_LOG}/local.log”},
/*
 ***************************************************************************
 * Accounting Unix files
 *
 *                  HPSS_PATH_ACCT      - unix path name for accounting files
 *                  HPSS_UNIX_ACCT_CHECKPOINT - checkpoint file
 *                  HPSS_UNIX_ACCT_REPORT- report file
 *                  HPSS_UNIX_ACCT_COMMENTARY- commentary file
 ***************************************************************************
 */
                    { “HPSS_PATH_ACCT”,“${HPSS_PATH_VAR}/acct”    },
                    { “HPSS_UNIX_ACCT_CHECKPOINT”,“${HPSS_PATH_ACCT}/acct_checkpoint” },
                    { “HPSS_UNIX_ACCT_REPORT”, “${HPSS_PATH_ACCT}/acct_report”    },
                    { “HPSS_UNIX_ACCT_COMMENTARY”, “${HPSS_PATH_ACCT}/acct_commentary”
},

/*
 ***************************************************************************
 * NFS Unix files
 *
 *                  HPSS_PATH_NFS       - unix path name for NFS files
 *                  HPSS_UNIX_NFS_EXPORTS- exports file
 *                  HPSS_UNIX_NFS_CREDMAP- credmap file
 *                  HPSS_UNIX_NFS_CACHEFILE- cache file
 *                  HPSS_UNIX_NFS_CHECKPOINT- checkpoint
 *                  HPSS_NFS_DISABLE_JUNCTIONS - disable traversal of junctions
 *                  HPSS_MNT_DISABLE_JUNCTIONS - disable mounting junctions
 **************************************************************************
 */
                    { “HPSS_PATH_NFS”,“${HPSS_PATH_VAR}/nfs”   },
                    { “HPSS_UNIX_NFS_EXPORTS”,“${HPSS_PATH_NFS}/exports”   },
                    { “HPSS_UNIX_NFS_CREDMAP”,“${HPSS_PATH_NFS}/credmap.nfs”   },
                    { “HPSS_UNIX_NFS_CACHEFILE”,“${HPSS_PATH_NFS}/cachefile.nfs”  },
                    { “HPSS_UNIX_NFS_CHECKPOINT”,“${HPSS_PATH_NFS}/checkpoint.nfs” },
                    { “HPSS_NFS_DISABLE_JUNCTIONS”,“TRUE” },
                    { “HPSS_MNT_DISABLE_JUNCTIONS”,“TRUE” },
/*
 ***************************************************************************
 * MPS Unix files
 *
 *                  HPSS_PATH_MPS       - unix path name for MPS files
 *                  HPSS_UNIX_MPS_REPORT- report file
 ***************************************************************************
 */
                    { “HPSS_PATH_MPS”,“${HPSS_PATH_VAR}/mps”},
                    { “HPSS_UNIX_MPS_REPORT”,“”       },
/*
 ***************************************************************************
 * Gatekeeper Unix files
 *
 *                  HPSS_PATH_GK        - unix path name for Gatekeeping files
 *                  HPSS_UNIX_GK_SITE_POLICY  - site policy file
 ***************************************************************************
 */
                    { "HPSS_PATH_GK","${HPSS_PATH_VAR}/gk"    },
                    { "HPSS_UNIX_GK_SITE_POLICY","${HPSS_PATH_GK}/gksitepolicy"      },
/*
 ***************************************************************************
 * SFS Files
 *
 *                  HPSS_SFS      - Encina SFS server name without CDS prefix
 *                  HPSS_SFS_SERVER- Encina SFS server name with CDS prefix
 *                  HPSS_SFS_SUFFIX- Suffix to add to end of SFS file names
 *                  ENCINA_LOCAL  - Encina local directory
 *                  ENCINA_MIRROR - Encina mirror directory
 ***************************************************************************
 */
                    { "HPSS_SFS",       "hpss"                    },
                    { "HPSS_SFS_SERVER","/.:/encina/sfs/${HPSS_SFS}"},
                    { "HPSS_SFS_SUFFIX",            ""                              },
                    { "ENCINA_LOCAL",               "/opt/encinalocal"              },
                    { "ENCINA_MIRROR",              "/opt/encinamirror"             },
/*
 ***************************************************************************
 * Common SFS Files - These SFS files are common to the system as a whole -
 *                             i.e. not specific to a storage subsystem or to a
 *                             particular server.
 *
 *                  HPSS_CONFIG_GLOBAL- Global config file
 *                  HPSS_CONFIG_STORSUBSYS- Storage subsystem configurations
 *                  HPSS_CONFIG_COS     - Class of service
 *                  HPSS_CONFIG_HIERARCHY- Hierarchy
 *                  HPSS_CONFIG_STORAGECLASS- Storage class
 *                  HPSS_CONFIG_SCLASS- Old name for Storage class
 *                  HPSS_CONFIG_SCLASSTHRESHOLD- Storage Class Thresholds
 *                  HPSS_CONFIG_NSGLOBALFILESETS- Global Filesets file
 *                  HPSS_CONFIG_FILE_FAMILY- File Family
 *                  HPSS_CONFIG_ACCOUNTING- Accounting Policy
 *                  HPSS_CONFIG_ACCT- Old name for Accounting Policy
 *                  HPSS_CONFIG_ACCTSUM- Accounting summary
 *                  HPSS_CONFIG_LOGPOLICY- Log policy
 *                  HPSS_CONFIG_LOGP- Old name for Log policy
 *                  HPSS_CONFIG_LSPOLICY- Location Server policy
 *                  HPSS_CONFIG_LS      - Old name for Location Server policy
 *                  HPSS_CONFIG_MIGPOLICY- Migration policy
 *                  HPSS_CONFIG_MIGRP- Old name for migration policy
 *                  HPSS_CONFIG_MOVERDEVICE- Mover device
 *                  HPSS_CONFIG_PURGEPOLICY- Purge policy
 *                  HPSS_CONFIG_PURGP- Old name for purge policy
 *                  HPSS_CONFIG_SERVER- Generic server configurations
 ***************************************************************************
 */
                    { "HPSS_CONFIG_GLOBAL",
                    "${HPSS_SFS_SERVER}/globalconfig${HPSS_SFS_SUFFIX}"},
                    { "HPSS_CONFIG_STORSUBSYS",
                          "${HPSS_SFS_SERVER}/storsubsysconfig${HPSS_SFS_SUFFIX}"},
                    { "HPSS_CONFIG_COS",
                          "${HPSS_SFS_SERVER}/cos${HPSS_SFS_SUFFIX}"},
                    { "HPSS_CONFIG_HIERARCHY",
                          “${HPSS_SFS_SERVER}/hierarchy${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_STORAGECLASS”,
                          “${HPSS_SFS_SERVER}/storageclass${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_SCLASS”,   “${HPSS_CONFIG_STORAGECLASS}”},
                    { “HPSS_CONFIG_SCLASSTHRESHOLD”,
                          “${HPSS_SFS_SERVER}/sclassthreshold${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_NSGLOBALFILESETS”,
                          “${HPSS_SFS_SERVER}/nsglobalfilesets${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_FILE_FAMILY”,
                          “${HPSS_SFS_SERVER}/filefamily${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_ACCOUNTING”,
                          “${HPSS_SFS_SERVER}/accounting${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_ACCT”,“${HPSS_CONFIG_ACCOUNTING}”},
                    { “HPSS_CONFIG_ACCTSUM”,
                          “${HPSS_SFS_SERVER}/acctsum${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_LOGPOLICY”,
                          “${HPSS_SFS_SERVER}/logpolicy${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_LOGP”, “${HPSS_CONFIG_LOGPOLICY}”},
                    { “HPSS_CONFIG_LSPOLICY”,
                          “${HPSS_SFS_SERVER}/lspolicy${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_LS”, “${HPSS_CONFIG_LSPOLICY}”},
                    { “HPSS_CONFIG_MIGPOLICY”,
                          “${HPSS_SFS_SERVER}/migpolicy${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_MIGRP”, “${HPSS_CONFIG_MIGPOLICY}”},
                    { “HPSS_CONFIG_MOVERDEVICE”,
                          “${HPSS_SFS_SERVER}/moverdevice${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_PURGEPOLICY”,
                          “${HPSS_SFS_SERVER}/purgepolicy${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_PURGP”,“${HPSS_CONFIG_PURGEPOLICY}”},
                    { “HPSS_CONFIG_SERVER”,
                          “${HPSS_SFS_SERVER}/serverconfig${HPSS_SFS_SUFFIX}”},
/*
 ***************************************************************************
 * Storage Subsystem SFS Files - These SFS files are specific to a
 *                              particular storage subsystem.  Note that the values
 *                              defined here represent the default “stem” of the
 *                              actual filename.  In order to get the full filename
 *                              append “.<subsys_id>” to the value defined here.
 *
 *                  HPSS_CONFIG_BFMIGRREC- Bitfile migration records
 *                  HPSS_CONFIG_BFPURGEREC- Bitfile purge records
 *                  HPSS_CONFIG_ACCTLOG- Accounting log
 *                  HPSS_CONFIG_BITFILE- BFS bitfile descriptors
 *                  HPSS_CONFIG_BFCOSCHANGE- BFS COS changes
 *                  HPSS_CONFIG_BFDISKSEGMENT- BFS bitfile segments - disk
 *                  HPSS_CONFIG_BFTAPESEGMENT- BFS bitfile segments - tape
 *                  HPSS_CONFIG_BFDISKALLOCREC- BFS disk allocation records
 *                  HPSS_CONFIG_BFSSSEGCHKPT- BFS segment checkpoints
 *                  HPSS_CONFIG_BFSSUNLINK- BFS unlink records
 *                  HPSS_CONFIG_NSACLS- NS ACLs
 *                  HPSS_CONFIG_NSFILESETATTRS- NS Fileset attributes
 *                  HPSS_CONFIG_NSOBJECTS- NS objects
 *                  HPSS_CONFIG_NSTEXT- NS text
 *                  HPSS_CONFIG_MPCHKPT- MPS checkpoint
 *                  HPSS_CONFIG_SSPVDISK- SS physical volume - disk
 *                  HPSS_CONFIG_SSPVTAPE- SS physical volume - tape
 *                  HPSS_CONFIG_STORAGEMAPDISK- SS storage map - disk
 *                  HPSS_CONFIG_STORAGEMAPTAPE- SS storage map - tape
 *                  HPSS_CONFIG_STORAGESEGDISK- SS storage segment - disk
 *                  HPSS_CONFIG_STORAGESEGTAPE- SS storage segment - tape
 *                   HPSS_CONFIG_VVDISK- SS virtual volume - disk
 *                   HPSS_CONFIG_VVTAPE- SS virtual volume - tape
 ***************************************************************************
 */
                     { “HPSS_CONFIG_BFMIGRREC”,
                            “${HPSS_SFS_SERVER}/bfmigrrec${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_BFPURGEREC”,
                            “${HPSS_SFS_SERVER}/bfpurgerec${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_ACCTLOG”,
                            “${HPSS_SFS_SERVER}/acctlog${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_BITFILE”,
                            “${HPSS_SFS_SERVER}/bitfile${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_BFCOSCHANGE”,
                            “${HPSS_SFS_SERVER}/bfcoschange${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_BFDISKSEGMENT”,
                            “${HPSS_SFS_SERVER}/bfdisksegment${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_BFTAPESEGMENT”,
                            “${HPSS_SFS_SERVER}/bftapesegment${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_BFDISKALLOCREC”,
                            “${HPSS_SFS_SERVER}/bfdiskallocrec${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_BFSSSEGCHKPT”,
                            “${HPSS_SFS_SERVER}/bfsssegchkpt${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_BFSSUNLINK”,
                            “${HPSS_SFS_SERVER}/bfssunlink${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_NSACLS”,
                            “${HPSS_SFS_SERVER}/nsacls${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_NSFILESETATTRS”,
                            “${HPSS_SFS_SERVER}/nsfilesetattrs${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_NSOBJECTS”,
                            “${HPSS_SFS_SERVER}/nsobjects${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_NSTEXT”,
                            “${HPSS_SFS_SERVER}/nstext${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_MPCHKPT”,
                            “${HPSS_SFS_SERVER}/mpchkpt${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_SSPVDISK”,
                            “${HPSS_SFS_SERVER}/sspvdisk${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_SSPVTAPE”,
                            “${HPSS_SFS_SERVER}/sspvtape${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_STORAGEMAPDISK”,
                            “${HPSS_SFS_SERVER}/storagemapdisk${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_STORAGEMAPTAPE”,
                            “${HPSS_SFS_SERVER}/storagemaptape${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_STORAGESEGDISK”,
                            “${HPSS_SFS_SERVER}/storagesegdisk${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_STORAGESEGTAPE”,
                            “${HPSS_SFS_SERVER}/storagesegtape${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_VVDISK”,
                            “${HPSS_SFS_SERVER}/vvdisk${HPSS_SFS_SUFFIX}”},
                     { “HPSS_CONFIG_VVTAPE”,
                            “${HPSS_SFS_SERVER}/vvtape${HPSS_SFS_SUFFIX}”},
/*
 ***************************************************************************
 * Accounting SFS Files
 *
 *                   HPSS_CONFIG_ACCTSNAP- Accounting snap shot
 *                   HPSS_CONFIG_ACCTVALIDATE- Account validation records
 ***************************************************************************
 */
                     { “HPSS_CONFIG_ACCTSNAP”,
                            “${HPSS_SFS_SERVER}/acctsnap${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_ACCTVALIDATE”,
                           “${HPSS_SFS_SERVER}/acctvalidate${HPSS_SFS_SUFFIX}”},
/*
 ***************************************************************************
 * BFS SFS Files
 *
 *                  HPSS_CONFIG_BFS     - BFS type specific
 ***************************************************************************
 */
                    { “HPSS_CONFIG_BFS”,
                           “${HPSS_SFS_SERVER}/bfs${HPSS_SFS_SUFFIX}”},
/*
 ***************************************************************************
 * NameServer SFS Files
 *
 *                  HPSS_CONFIG_NS      - NS type specific
 ***************************************************************************
 */
                    { “HPSS_CONFIG_NS”,
                           “${HPSS_SFS_SERVER}/nsconfig${HPSS_SFS_SUFFIX}”},
/*
 ***************************************************************************
 * DMAP Gateway SFS Files
 *
 *                  HPSS_CONFIG_DMG     - DMG type specific
 *                  HPSS_CONFIG_DMGFILESET- DMG filesets
 ***************************************************************************
 */
                    { “HPSS_CONFIG_DMG”,
                           “${HPSS_SFS_SERVER}/dmg${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_DMGFILESET”,
                           “${HPSS_SFS_SERVER}/dmgfileset${HPSS_SFS_SUFFIX}”},
/*
 ***************************************************************************
 * Gatekeeper Server SFS Files
 *
 *      HPSS_CONFIG_GK                  - GK type specific
 ***************************************************************************
 */
        { “HPSS_CONFIG_GK”,
                “${HPSS_SFS_SERVER}/gkconfig${HPSS_SFS_SUFFIX}”         },
/*
 ***************************************************************************
 * Logging SFS Files
 *
 *                  HPSS_CONFIG_LOGC- Log client type specific
 *                  HPSS_CONFIG_LOGD- Log daemon type specific
 ***************************************************************************
 */
                    { “HPSS_CONFIG_LOGC”,
                           “${HPSS_SFS_SERVER}/logclient${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_LOGD”,
                           “${HPSS_SFS_SERVER}/logdaemon${HPSS_SFS_SUFFIX}”},
/*
 ***************************************************************************
 * Location Server SFS Files
 *
 *                  HPSS_CONFIG_SITE- Location Server site file
 ***************************************************************************
 */
```

```
                    { “HPSS_CONFIG_SITE”,
                           “${HPSS_SFS_SERVER}/site${HPSS_SFS_SUFFIX}”},
/*
 ***************************************************************************
 * Metadata Monitor SFS Files
 *
 *                  HPSS_CONFIG_MM      - Metadata Monitor type specific
 ***************************************************************************
 */
                    { “HPSS_CONFIG_MM”,
                           “${HPSS_SFS_SERVER}/mmonitor${HPSS_SFS_SUFFIX}”},
/*
 ***************************************************************************
 * Migration/Purge Server SFS Files
 *
 *                  HPSS_CONFIG_MPS     - MPS type specific
 ***************************************************************************
 */
                    { “HPSS_CONFIG_MPS”,
                           “${HPSS_SFS_SERVER}/mps${HPSS_SFS_SUFFIX}”},
/*
 ***************************************************************************
 * Mover SFS Files
 *
 *                  HPSS_CONFIG_MVR     - Mover type specific
 ***************************************************************************
 */
                    { “HPSS_CONFIG_MVR”,
                           “${HPSS_SFS_SERVER}/mover${HPSS_SFS_SUFFIX}”},
/*
 ***************************************************************************
 * Non-DCE SFS Files
 *
 *                  HPSS_CONFIG_NDCG- Gateway type specific
 ***************************************************************************
 */
                    { “HPSS_CONFIG_NDCG”,
                           “${HPSS_SFS_SERVER}/ndcg${HPSS_SFS_SUFFIX}”},
/*
 ***************************************************************************
 * NFS Daemon SFS Files
 *
 *                  HPSS_CONFIG_NFSD- NFSD type specific
 ***************************************************************************
 */
                    { “HPSS_CONFIG_NFSD”,
                           “${HPSS_SFS_SERVER}/nfs${HPSS_SFS_SUFFIX}” },
/*
 ***************************************************************************
 * PVL SFS Files
 *
 *                  HPSS_CONFIG_PVL     - PVL type specific
 *                  HPSS_CONFIG_PVLACTIVITY- PVL activity
 *                  HPSS_CONFIG_PVLDRIVE- PVL drive
 *                  HPSS_CONFIG_PVLJOB- PVL job
 *                  HPSS_CONFIG_PVLPV- PVL physical volume
 ***************************************************************************
 */
                    { “HPSS_CONFIG_PVL”,
                           “${HPSS_SFS_SERVER}/pvl${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_PVLACTIVITY”,
                           “${HPSS_SFS_SERVER}/pvlactivity${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_PVLDRIVE”,
                           “${HPSS_SFS_SERVER}/pvldrive${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_PVLJOB”,
                           “${HPSS_SFS_SERVER}/pvljob${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_PVLPV”,
                           “${HPSS_SFS_SERVER}/pvlpv${HPSS_SFS_SUFFIX}”},
/*
 ***************************************************************************
 * PVR SFS Files
 *
 *                  HPSS_CONFIG_PVR      - PVR type specific
 *                  HPSS_CONFIG_CART_AMPEX- PVR ampex cartridges
 *                  HPSS_CONFIG_CART_OPER- PVR operator cartridges
 *                  HPSS_CONFIG_CART_STK- PVR STK cartridges
 *                  HPSS_CONFIG_CART_STK_RAIT       - PVR STK RAIT cartridges
 *                  HPSS_CONFIG_CART_3494- PVR 3494 cartridges
 *                  HPSS_CONFIG_CART_3495- PVR 3495 cartridges
 *      HPSS_CONFIG_CART_LTO- PVR LTO cartridges
 *                  HPSS_CONFIG_CART_AML- PVR AML cartridges
 ***************************************************************************
 */
                    { “HPSS_CONFIG_PVR”,
                           “${HPSS_SFS_SERVER}/pvr${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_CART_AMPEX”,
                           “${HPSS_SFS_SERVER}/cartridge_ampex${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_CART_OPER”,
                           “${HPSS_SFS_SERVER}/cartridge_operator${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_CART_STK”,
                           “${HPSS_SFS_SERVER}/cartridge_stk${HPSS_SFS_SUFFIX}”},
        { “HPSS_CONFIG_CART_STK_RAIT”,
                “${HPSS_SFS_SERVER}/cartridge_stk_rait${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_CART_3494”,
                           “${HPSS_SFS_SERVER}/cartridge_3494${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_CART_3495”,
                           “${HPSS_SFS_SERVER}/cartridge_3495${HPSS_SFS_SUFFIX}”},
                    { “HPSS_CONFIG_CART_LTO”,
                           “${HPSS_SFS_SERVER}/cartridge_lto${HPSS_SFS_SUFFIX}”   },
                    { “HPSS_CONFIG_CART_AML”,
                “${HPSS_SFS_SERVER}/cartridge_aml${HPSS_SFS_SUFFIX}”    },
/*
 ***************************************************************************
 * Storage Server SFS Files
 *
 *                  HPSS_CONFIG_SS       - SS type specific
 ***************************************************************************
 */
                    { “HPSS_CONFIG_SS”,
                           “${HPSS_SFS_SERVER}/ss${HPSS_SFS_SUFFIX}”},
/*
 ***************************************************************************
 * DCE CDS Names - used as defaults for ServerName in hpss_server_config_t
 *
 *                  HPSS_CDS_PREFIX      - CDS prefix for HPSS servers
 *
 *                  HPSS_CDS_BFS         - CDS name - Bitfile Server
 *                  HPSS_CDS_DMG         - CDS name - DMAP Gateway
 *                  HPSS_CDS_GK          - CDS name - Gatekeeper Server
 *                  HPSS_CDS_HPSSD       - CDS name - Startup Daemon
 *                   HPSS_CDS_LOGC       - CDS name - Log Client
 *                   HPSS_CDS_LOGD       - CDS name - Log Daemon
 *                   HPSS_CDS_LS         - CDS name - Location Server
 *                   HPSS_CDS_MM         - CDS name - Metadata Monitor
 *                   HPSS_CDS_MOUNTD     - CDS name - Mount Daemon
 *                   HPSS_CDS_MPS        - CDS name - Migration/Purge Server
 *                   HPSS_CDS_MVR        - CDS name - Mover
 *                   HPSS_CDS_NDCG       - CDS name - Non-DCE Gateway
 *                   HPSS_CDS_NFSD       - CDS name - NFS Daemon
 *                   HPSS_CDS_NS         - CDS name - Name Server
 *                   HPSS_CDS_PFSD       - CDS name - PFSD
 *                   HPSS_CDS_PVL        - CDS name - PVL
 *                   HPSS_CDS_PVR_AMPEX- CDS name - PVR - Ampex
 *                   HPSS_CDS_PVR_OPER- CDS name - PVR - Operator
 *                   HPSS_CDS_PVR_STK- CDS name - PVR - STK
 *                   HPSS_CDS_PVR_STK_RAIT- CDS name - PVR - STK RAIT
 *                   HPSS_CDS_PVR_3494- CDS name - PVR - 3494
 *                   HPSS_CDS_PVR_3495- CDS name - PVR - 3495
 *       HPSS_CDS_PVR_LTO    - CDS name - PVR - LTO
 *                   HPSS_CDS_PVR_AML- CDS name - PVR - AML
 *                   HPSS_CDS_SSDISK     - CDS name - Storage Server - Disk
 *                   HPSS_CDS_SSTAPE     - CDS name - Storage Server - Tape
 *                   HPSS_CDS_SSMDS      - CDS name - SSM - Data Server
 *                   HPSS_CDS_SSMSM      - CDS name - SSM - System Manager
 ***************************************************************************
 */

                     { “HPSS_CDS_PREFIX”,“/.:/hpss”    },

                     { “HPSS_CDS_BFS”,“${HPSS_CDS_PREFIX}/bfs”},
                     { “HPSS_CDS_DMG”,“${HPSS_CDS_PREFIX}/dmg”},
                     { “HPSS_CDS_GK”,“${HPSS_CDS_PREFIX}/gk”},
                     { “HPSS_CDS_HPSSD”,“${HPSS_CDS_PREFIX}/hpssd”},
                     { “HPSS_CDS_LOGC”,“${HPSS_CDS_PREFIX}/logc”},
                     { “HPSS_CDS_LOGD”,“${HPSS_CDS_PREFIX}/logd”},
                     { “HPSS_CDS_LS”,“${HPSS_CDS_PREFIX}/ls”},
                     { “HPSS_CDS_MM”,“${HPSS_CDS_PREFIX}/mmonitor”},
                     { “HPSS_CDS_MOUNTD”,“${HPSS_CDS_PREFIX}/mountd”},
                     { “HPSS_CDS_MPS”,“${HPSS_CDS_PREFIX}/mps”},
                     { “HPSS_CDS_MVR”,“${HPSS_CDS_PREFIX}/mvr”},
                     { “HPSS_CDS_NDCG”,“${HPSS_CDS_PREFIX}/ndcg”},
                     { “HPSS_CDS_NFSD”,“${HPSS_CDS_PREFIX}/nfs”},
                     { “HPSS_CDS_NS”,“${HPSS_CDS_PREFIX}/cns”},
                     { “HPSS_CDS_PFSD”,“${HPSS_CDS_PREFIX}/piofsie”},
                     { “HPSS_CDS_PVL”,“${HPSS_CDS_PREFIX}/pvl”},
                     { “HPSS_CDS_PVR_AMPEX”,“${HPSS_CDS_PREFIX}/pvr_ampex”},
                     { “HPSS_CDS_PVR_OPER”,“${HPSS_CDS_PREFIX}/pvr_operator”},
                     { “HPSS_CDS_PVR_STK”,“${HPSS_CDS_PREFIX}/pvr_stk”},
                     { “HPSS_CDS_PVR_STK_RAIT”,“${HPSS_CDS_PREFIX}/pvr_stk_rait”},
                     { “HPSS_CDS_PVR_3494”,“${HPSS_CDS_PREFIX}/pvr_3494”},
                     { “HPSS_CDS_PVR_3495”,“${HPSS_CDS_PREFIX}/pvr_3495”},
                     { “HPSS_CDS_PVR_LTO”,          “${HPSS_CDS_PREFIX}/pvr_lto”},
                     { “HPSS_CDS_PVR_AML”,“${HPSS_CDS_PREFIX}/pvr_aml”},
                     { “HPSS_CDS_SSDISK”,“${HPSS_CDS_PREFIX}/ssdisk”},
                     { “HPSS_CDS_SSTAPE”,“${HPSS_CDS_PREFIX}/sstape”},
                     { “HPSS_CDS_SSMDS”,“${HPSS_CDS_PREFIX}/ssmds”},
                     { “HPSS_CDS_SSMSM”,“${HPSS_CDS_PREFIX}/ssmsm”},
/*
 ***************************************************************************
 * Descriptive Names
 *
 *                   HPSS_DESC_BFS       - Descriptive name - Bitfile Server
 *                   HPSS_DESC_DMG       - Descriptive name - DMAP Gateway
 *                   HPSS_DESC_FTPD      - Descriptive name - FTP Daemon
 *                   HPSS_DESC_GK        - Descriptive name - Gatekeeper Server
 *                   HPSS_DESC_HPSSD     - Descriptive name - Startup Daemon
 *                   HPSS_DESC_LOGC      - Descriptive name - Log Client
 *                   HPSS_DESC_LOGD      - Descriptive name - Log Daemon
 *                   HPSS_DESC_LS        - Descriptive name - Location Server
 *                   HPSS_DESC_MM        - Descriptive name - Metadata Monitor
 *                   HPSS_DESC_MOUNTD- Descriptive name - Mount Daemon
 *                   HPSS_DESC_MPS       - Descriptive name - MPS
 *                   HPSS_DESC_MVR       - Descriptive name - Mover
 *                   HPSS_DESC_NDCG      - Descriptive name - Non-DCE Gateway
 *                   HPSS_DESC_NFSD      - Descriptive name - NFS Daemon
 *                   HPSS_DESC_NS        - Descriptive name - Name Server
 *                   HPSS_DESC_PFSD      - Descriptive name - PFSD
 *                   HPSS_DESC_PVL       - Descriptive name - PVL
 *                   HPSS_DESC_PVR_AMPEX- Descriptive name - PVR - Ampex
 *                   HPSS_DESC_PVR_OPER- Descriptive name - PVR - Operator
 *                   HPSS_DESC_PVR_STK- Descriptive name - PVR - STK
 *      HPSS_DESC_PVR_STK_RAIT          - Descriptive name - PVR - STK RAIT
 *                   HPSS_DESC_PVR_3494- Descriptive name - PVR - 3494
 *                   HPSS_DESC_PVR_3495- Descriptive name - PVR - 3495
 *      HPSS_DESC_PVR_LTO  - Descriptive name - PVR - LTO
 *                   HPSS_DESC_PVR_AML- Descriptive name - PVR - AML
 *                   HPSS_DESC_SSDISK- Descriptive name - SS - Disk
 *                   HPSS_DESC_SSTAPE- Descriptive name - SS - Tape
 *                   HPSS_DESC_SSMSM     - Descriptive name - SSM System Manager
 ***************************************************************************
 */
                   { “HPSS_DESC_BFS”,“BFS”                  },
                   { “HPSS_DESC_DMG”,“DMAP Gateway”  },
                   { “HPSS_DESC_FTPD”,“FTP Daemon”   },
                   { “HPSS_DESC_GK”,“Gatekeeper Server”},
                   { “HPSS_DESC_HPSSD”,“Startup Daemon (${HPSS_HOST})”},
                   { “HPSS_DESC_LOGC”,“Log Client (${HPSS_HOST})”},
                   { “HPSS_DESC_LOGD”,“Log Daemon”   },
                   { “HPSS_DESC_LS”,“Location Server”},
                   { “HPSS_DESC_MM”,“Metadata Monitor”},
                   { “HPSS_DESC_MOUNTD”,“Mount Daemon”},
                   { “HPSS_DESC_MPS”,“Migration/Purge Server”},
                   { “HPSS_DESC_MVR”,“Mover (${HPSS_HOST})”},
                   { “HPSS_DESC_NDCG”,“Non-DCE Gateway”},
                   { “HPSS_DESC_NFSD”,“NFS Daemon”   },
                   { “HPSS_DESC_NS”,“Name Server”    },
                   { “HPSS_DESC_PFSD”,“PIOFS Import/Export”},
                   { “HPSS_DESC_PVL”, “PVL”                 },
                   { “HPSS_DESC_PVR_AMPEX”,“Ampex PVR”},
                   { “HPSS_DESC_PVR_OPER”,“Operator PVR”},
                   { “HPSS_DESC_PVR_STK”,“STK PVR”   },
       { “HPSS_DESC_PVR_STK_RAIT”,     “STK RAIT PVR”                  },
                   { “HPSS_DESC_PVR_3494”,“3494 PVR” },
                   { “HPSS_DESC_PVR_3495”,“3495 PVR” },
                   { “HPSS_DESC_PVR_LTO”,“3584 LTO PVR”},
                   { “HPSS_DESC_PVR_AML”,“AML PVR”   },
                   { “HPSS_DESC_SSDISK”, “Disk Storage Server”},
                   { “HPSS_DESC_SSTAPE”, “Tape Storage Server”},
                   { “HPSS_DESC_SSMSM”,“SSM System Manager”},
/*
```

```
 ***************************************************************************
 * System Manager Specific
 *
 *   The SM attempts to throttle the connection attempts to other servers. It
 *   will attempt to reconnect to each server every
 *   HPSS_SM_SRV_CONNECT_INTERVAL_MIN seconds until the number of failures for
 *   that server has reached HPSS_SM_SRV_CONNECT_FAIL_COUNT. After the failure
 *   count has been reached the SM will only try to reconnect to the server
 *   every HPSS_SM_SRV_CONNECT_INTERVAL_MAX seconds until a successful i
 *   connection is made at which time the connection interval for the server
 *   will be set back to HPSS_SM_SRV_CONNECT_INTERVAL_MIN.
 *
 *                 HPSS_SM_SRV_CONNECT_FAIL_COUNT- Number of connection failures to a
 *                                             server before the
 *                                             HPSS_SM_SRV_CONNECT_INTERVAL_MAX
 *                                             interval takes affect
 *                 HPSS_SM_SRV_CONNECT_INTERVAL_MIN - Interval between attempting server
 *                                             connections when
 *                                             HPSS_SM_SERVER_CONNECT_FAIL_COUNT
 *                                             has not yet been reached (seconds)
 *                 HPSS_SM_SRV_CONNECT_INTERVAL_MAX - Interval between server
connections
 *                                             when HPSS_SM_SERVER_CONNECT_FAIL_COUNT
 *                                             has been reached without a successful
 *                                             connection (seconds)
 *
 *                 HPSS_SM_SRV_LIST_UPDATE_INTERVAL - Frequency at which the SM sends
 *                                             updated server lists to the Data
 *                                             Server so that we don’t flood the
 *                                             Data Server with updates (seconds)
 *                 HPSS_SM_SRV_MONITOR_THREADS- Number of threads created to monitor
 *                                             server connections
 *      ENCINA_TPOOL_SIZE              - Thread Pool Size used by the System
 *                                       Manager
 ***************************************************************************
 */
                   { “HPSS_SM_SRV_CONNECT_FAIL_COUNT”,“3”},
                   { “HPSS_SM_SRV_CONNECT_INTERVAL_MIN”,“20”},
                   { “HPSS_SM_SRV_CONNECT_INTERVAL_MAX”,“60”},
                   { “HPSS_SM_SRV_LIST_UPDATE_INTERVAL”,“5”},
                   { “HPSS_SM_SRV_MONITOR_THREADS”,“10”},
                   { “ENCINA_TPOOL_SIZE”,                 “100”                  },
/*
 ***************************************************************************
 * The HPSS_NOTIFY_Q_*_THREADS variables define how many threads the SSM
 * System Manager will create per client to process that queue.  Each thread
 * will take notifications one at a time from the queue and forward them to
 * its client.  The recommended number of threads per client per queue is 1.
 * Too many threads per client per queue will cause unnecessary memory growth
 * in the SSM System Manager.
 *
 *                 HPSS_NOTIFY_Q_DATA_THREADS - Number of threads to create per client
 *                                         to process the queue of notifications of
 *                                         managed object attribute changes
 *                 HPSS_NOTIFY_Q_LIST_THREADS - Number of threads to create per client
 *                                         to process the queue of notifications of
 *                                         changes to the server list, drive list,
 *                                         class of service list, storage class
 *                                         list, hierarchy list, migration policy
 *                                         list, or purge policy list
 *                  HPSS_NOTIFY_Q_LOG_THREADS -  Number of threads to create per client
 *                                           to process the queue of notifications
 *                                           of alarms, events, and status messages
 *                  HPSS_NOTIFY_Q_TAPE_THREADS - Number of threads to create per client
 *                                           to process the queue of notifications
 *                                           of tape mounts and unmounts
 *                  HPSS_NOTIFY_Q_TAPE_CHECKIN_THREADS - Number of threads to create per
 *                                           client to process the queue of
 *                                           notifications of tape check-in and
 *                                  check-out requests
 ***************************************************************************
 */
                    { "HPSS_NOTIFY_Q_DATA_THREADS","1"},
                    { "HPSS_NOTIFY_Q_LIST_THREADS","1"},
                    { "HPSS_NOTIFY_Q_LOG_THREADS","1" },
                    { "HPSS_NOTIFY_Q_TAPE_THREADS","1"},
                    { "HPSS_NOTIFY_Q_TAPE_CHECKIN_THREADS","1"},
/*
 ***************************************************************************
 * The HPSS_NOTIFY_Q_*_SIZE_MAX variables define the maximum size to which
 * each queue is allowed to grow.  Once the queue reaches that limit, incoming
 * notifications will be stalled until the queue size shrinks.
 *
 *                  HPSS_NOTIFY_Q_DATA_SIZE_MAX - Maximum queue size for notifications
 *                                            of changes in managed object attributes
 *                  HPSS_NOTIFY_Q_LIST_SIZE_MAX - Maximum queue size for notifications
 *                                            of changes to the server list, drive
 *                                            list, class of service list, storage
 *                                            class list, hierarchy list, migration
 *                                            policy list, or purge policy list
 *                  HPSS_NOTIFY_Q_LOG_SIZE_MAX -  Maximum queue size for notifications
 *                                            of alarms, events, or status messages
 *                  HPSS_NOTIFY_Q_TAPE_SIZE_MAX - Maximum queue size for notifications
 *                                            of tape mounts and unmounts
 *                  HPSS_NOTIFY_Q_TAPE_CHECKIN_SIZE_MAX - Maximum queue size for
 *                                            notifications of tape check-in and
 *                                   check-out requests
 ***************************************************************************
 */
                    { "HPSS_NOTIFY_Q_DATA_SIZE_MAX","500"},
                    { "HPSS_NOTIFY_Q_LIST_SIZE_MAX","500"},
                    { "HPSS_NOTIFY_Q_LOG_SIZE_MAX","500"},
                    { "HPSS_NOTIFY_Q_TAPE_SIZE_MAX","500"},
                    { "HPSS_NOTIFY_Q_TAPE_SIZE_CHECKIN_MAX","500"},
/*
 ***************************************************************************
 * Data Server Specific
 *
 * These variables are required for the Data Server regardless of whether
 * it supports hpssadm or not:
 *
 *                  HPSS_ALARMS_SSMDS- File to store Sammi Alarms/Events
 *                                        NULL -> DS will store internally
 *                  HPSS_PORT_SSMDS- Sun RPC address for the Data Server
 *
 * These variables are required for the Java DS and the hpssadm utility:
 *
 *      HPSS_PATH_SSM           - unix path name for data server files
 *                  HPSS_SSMDS_COUNTRY- Country for Java internationalization
 *                  HPSS_SSMDS_LANGUAGE- Language for Java internationalization
 *                   HPSS_SSMDS_INTERVAL- Interval at which DS checks idle RMI clients
 *                   HPSS_SSMDS_RMI_HOST- Java RMI host for DS
 *                   HPSS_SSMDS_RMI_NAME- Java RMI base name for DS
 *                   HPSS_SSMDS_RMI_PORT- Port for Java RMI (Remote Method Invocation)
 *                   HPSS_SSMDS_KEYSTORE- Data Server keystore file (for SSL)
 *                   HPSS_SSMDS_KEYSTORE_PW- File holding password to Data Server keystore
 *                                         file, or the string “PROMPT” if the sysadm
 *                                         wishes to be prompted for the password at
 *                                         Data Server startup time
 *                   HPSS_SSMDS_JAVA_CONFIG- Pathname of DS Java config file
 *                   HPSS_SSMDS_JAVA_POLICY- Java policy file for DS
 *                   HPSS_HPSSADM_JAVA_POLICY- Java policy file for hpssadm utility
 *                   JAVA_ROOT     - top level where Java is installed
 *                   JAVA_HOME     - top level for Java Runtime Environment
 *                   JAVA_BIN      - location of tools like the Java interpreter
 *                   JAVA_LIB1     - main Java library directory
 *                   JAVA_LIB2     - Java native threads library directories
 *                   JAVA_LIB3     - HPSS library directory
 *                   HPSS_SSMDS_LIBPATH- library search path for the Java VM
 *                                         created by the DS
 *                   JAVA_CLS1     - location of standard Java classes
 *                   JAVA_CLS2     - location of hpssadm Java classes
 *                   JAVA_CLS3     - location of mobjects Java classes
 *                   HPSS_SSMDS_CLASSPATH- SSMDS search path for Java classes
 *                   AIXTHREAD_MNRATIO- required for JNI under Java 1.3.0 on AIX
 *                   AIXTHREAD_SCOPE- required for JNI under Java 1.3.0 on AIX
 *                   AIXTHREAD_MUTEX_DEBUG- required for JNI under Java 1.3.0 on AIX
 *                   AIXTHREAD_RWLOCK_DEBUG- required for JNI under Java 1.3.0 on AIX
 *                   AIXTHREAD_COND_DEBUG- required for JNI under Java 1.3.0 on AIX
 *
 * The DS and the hpssadm utility each set the default HPSS_SSMDS_RMI_HOST
 * at runtime as the host upon which they find themselves running.  Therefore,
 * if the DS and all hpssadm clients are not always running on the same host,
 * this variable must be overridden in hpss_env so that the hpssadm clients
 * will know where to find the DS.
 *
 * When the HPSS_SSMDS_RMI_PORT is left NULL as defined here, the Java RMI
 * Registry and the DS and hpssadm utilities will use port 1099.
 *
 * The JAVA_HOME variable is used by several HPSS scripts to set the command
 * execution PATH and other variables.
 *
 ***************************************************************************
 */
       { “HPSS_PATH_SSM”,        “${HPSS_PATH_VAR}/ssm”                    },
                   { “HPSS_ALARMS_SSMDS”,   NULL                                 },
                   { “HPSS_PORT_SSMDS”,   “0x20000069”                           },

                   { “HPSS_SSMDS_COUNTRY”,   “US”                                },
                   { “HPSS_SSMDS_LANGUAGE”,   “en”                               },
                   { “HPSS_SSMDS_INTERVAL”,   “60000000”                         },
                   { “HPSS_SSMDS_RMI_HOST”,   NULL                               },
                   { “HPSS_SSMDS_RMI_NAME”,   “HPSS_DataServer”                  },
                   { “HPSS_SSMDS_RMI_PORT”,   “1066”                             },
                   { “HPSS_SSMDS_KEYSTORE”,   “${HPSS_PATH_VAR}/ssm/keystore.ds”
},
                   { “HPSS_SSMDS_KEYSTORE_PW”,”${HPSS_PATH_VAR}/ssm/keystore.ds.pw”
},
                   { “HPSS_SSMDS_JAVA_CONFIG”,”${HPSS_PATH_VAR}/ssm/hpssadm.config”
},
                   { “HPSS_SSMDS_JAVA_POLICY”,”${HPSS_PATH_VAR}/ssm/java.policy.ds”
},
                   { “HPSS_HPSSADM_JAVA_POLICY”,
                                          “${HPSS_PATH_VAR}/ssm/java.policy.hpssadm” },
                   { “JAVA_ROOT”,   “/usr/java130”                      },
                   { “JAVA_HOME”,   “${JAVA_ROOT}/jre”                      },
                   { “JAVA_BIN”,    “${JAVA_ROOT}/bin”                      },
                   { “JAVA_LIB1”,   “${JAVA_HOME}/lib”                      },
                   { “JAVA_LIB2”,   “${JAVA_ROOT}/bin:${JAVA_ROOT}/bin/classic”},
                   { “JAVA_LIB3”,   “${HPSS_ROOT}/lib”                      },
                   { “HPSS_SSMDS_LIBPATH”,  “${JAVA_LIB2}:${JAVA_LIB3}”             },
                   { “JAVA_CLS1”,   “${JAVA_LIB1}/rt.jar”                  },
                   { “JAVA_CLS2”,   “${HPSS_ROOT}/bin/hpssadm.jar”      },
                   { “JAVA_CLS3”,   “${HPSS_ROOT}/bin/mobjects.jar”      },
                   { “HPSS_SSMDS_CLASSPATH”, “${JAVA_CLS1}:${JAVA_CLS2}:${JAVA_CLS3}”
},
                   { “AIXTHREAD_MNRATIO”,   “1:1”                                  },
                   { “AIXTHREAD_SCOPE”,   “S”                     },
                   { “AIXTHREAD_MUTEX_DEBUG”, “OFF”      },
                   { “AIXTHREAD_RWLOCK_DEBUG”,”OFF”      },
                   { “AIXTHREAD_COND_DEBUG”,  “OFF”      },

/*
 ***************************************************************************
 * HDM Specific
 *
 *      HPSS_HDM_SHMEM_KEY - The HDM’s shared memory key (default 3789)
 *      HPSS_HDM_SERVER_ID - The HDM’s Server ID (default 1)
 ***************************************************************************
 */
                   { “HPSS_HDM_SERVER_ID”,         “1”                           },
                   { “HPSS_HDM_SHMEM_KEY”,         “3789”                        },
                   { “HPSS_PATH_HDM”,              “${HPSS_PATH_VAR}/hdm”        },
/*
 ***************************************************************************
 * LOG Specific
 *
 *      HPSSLOG_SHMKEY - The HPSS logging shared memory key (default 2801)
 *      HPSSLOGGER     - HPSS logger output destination [ stdout | syslog ]
 ***************************************************************************
 */
                   { “HPSSLOG_SHMKEY”,             “2801”                        },
                   { “HPSSLOGGER”,                 ““                            },
/*
 ***************************************************************************
 * FTPD Specific
 *
 *      HPSS_PATH_FTP          - FTP daemon ./etc pathname
 *      HPSS_FTPD_CONTROL_PORT - FTP daemon control default port ID
 *      HPSS_FTP_RESERVED      - FTP reserved port IDs
 *      HPSS_FTP_BLOCK_SIZE    - FTP block size
 ***************************************************************************
 */
                   { “HPSS_PATH_FTP”,              “${HPSS_PATH_VAR}/ftp”        },
                   { “HPSS_FTPD_CONTROL_PORT”,     “4021”                        },
                   { “HPSS_FTP_RESERVED”,          “1025”                        },
                   { “HPSS_FTP_BLOCK_SIZE”,        “4”                           },
/*
 ***************************************************************************
 * LS Specific
 *
 *      HPSS_LS_NAME                    - Location Server rpc group
 ***************************************************************************
 */
                   { “HPSS_LS_NAME”,       “${HPSS_CDS_LS}/group”                  },

/*
 ***************************************************************************
 * NDAPI Specific
 *
 *      HPSS_NDCG_KRB5_SERVICENAME - Non DCE Gateway kerberos servicename
 *      HPSS_KRB_TO_DCE_FILE       - File to translate krb5 realm names
 *                                   into DCE cellnames
 *      HPSS_KCHILD_PATH           - Pathname for the ndcg_kchild executable
 *      HPSS_KCHILD_KEYTAB         - kerberos Keytab filename
 *      HPSS_NDCL_KEY_CONFIG_FILE  - File containing encryption key (DCE auth)
 ***************************************************************************
 */
                   { “HPSS_NDCG_KRB5_SERVICENAME”, “ndcg” },
                   { “HPSS_KRB_TO_DCE_FILE”,       “${HPSS_PATH_ETC}/
krb5_Realms_to_DCE_Cells” },
                   { “HPSS_KCHILD_PATH”,           “${HPSS_PATH_BIN}/ndcg_kchild” },
                   { “HPSS_KCHILD_KEYTAB”,         “/krb5/v5srvtab” },
                   { “HPSS_NDCL_KEY_CONFIG_FILE”,  “${HPSS_PATH_ETC}/ndcl.keyconfig” },
/*
 ***************************************************************************
 * Installation & Miscellaneous
 *
 *      RPC_SUPPORTED_PROTSEQS - The RPC transport protocol used
 *      HPSS_PATH_ADM          - Pathname where HPSS administrative files are placed
 *      HPSS_PATH_CORE         - Pathname where HPSS subsystem core files are placed
 *      HPSS_PATH_TMP          - Pathname where HPSS temporary files are placed
 *      HPSS_PATH_ETC          - Pathname where HPSS runtime config files are placed
 *      HPSS_AUTH_LEVEL        - he security authorization level
 ***************************************************************************
 */
                   { “RPC_SUPPORTED_PROTSEQS”,     “ncadg_ip_udp”                  },
                   { “HPSS_PATH_ADM”,              “${HPSS_PATH_VAR}/adm”          },
                   { “HPSS_PATH_CORE”,             “${HPSS_PATH_ADM}/core”         },
                   { “HPSS_PATH_TMP”,              “${HPSS_PATH_VAR}/tmp”          },
                   { “HPSS_PATH_ETC”,              “${HPSS_PATH_VAR}/etc”          },
                   { “HPSS_AUTH_LEVEL”,            “2”                             },
                   { NULL,                         NULL,                           }
};
#endif
```

5.4 Configure the HPSS Infrastructure on a Node

The HPSS Infrastructure Configuration must be performed on each node after the HPSS system is successfully installed. The following steps should be performed:

1. Configure HPSS with DCE (only on the first core server node in the cell).
2. Configure Encina SFS (only if Encina SFS will run on this node).
3. Create and Manage SFS Files (only if Encina SFS will run on this node).
4. Set up FTP Daemon (only if FTP Daemon will run on this node).
5. Set up Startup Daemon.
6. Add SSM User (only if SSM will run on this node).
7. Start SSM Servers / Session (only if SSM will run on this node).

The HPSS Infrastructure Configuration utility, **mkhpss**, is used to configure the above steps for the Encina SFS configuration. Refer to Section 5.5: *Using the mkhpss Utility* on page 241 for more information on the options supported by the utility. Refer to Section Chapter 5: *HPSS Infrastructure Configuration* on page 215 for more information on how to configure Encina SFS.

For example outputs from a sample configuration session, refer to either:

- Section E.1: *AIX Infrastructure Configuration Example* on page 497
- Section E.3: *Solaris Infrastructure Configuration Example* on page 511

To start the HPSS infrastructure configuration on the installation node:

1. Log in as **root** user to the node.
2. Change directory to **/opt/hpss/config**:

   ```
   % cd /opt/hpss/config
   ```

3. Execute the shell script **mkhpss**:

   ```
   % ./mkhpss
   ```

4. Select the proper menu option for the desired procedure.

5.5 *Using the* mkhpss *Utility*

The **mkhpss** utility is an interactive, menu-driven HPSS process. It provides menu options that allow the user to perform the needed infrastructure configurations (as described in Section 5.5.1 through 5.5.9) as well as an Exit option to quit the utility. Section 6.8.4: *Log Client Specific Configuration* (page 333) describes additional options not on the Menu.

```
<mkhpss>                Infrastructure Configuration Menu
                        =================================
<mkhpss> Prompt ==>   Select Infrastructure Configuration Option:
<mkhpss>              [1] Configure HPSS with DCE
<mkhpss>              [2] Configure SFS Server
```

```
<mkhpss>              [3] Create and Manage SFS Files
<mkhpss>              [4] Set Up FTP Daemon
<mkhpss>              [5] Set Up Startup Daemon
<mkhpss>              [6] Add SSM Administrative User
<mkhpss>              [7] Start SSM Servers/User Session

<mkhpss>              [E] Re-run hpss_env()
<mkhpss>              [U] Un-configure HPSS
<mkhpss>              [X] Exit
<mkhpss> Reply ===> (Select Option [1-7, E, U, X]):
```

Messages will be provided to indicate the status of the HPSS infrastructure configuration process at each stage. The HPSS infrastructure configuration results can be obtained from the messages displayed during the HPSS infrastructure configuration process or can be reviewed by reading the **/opt/hpss/config/mkhpss.data** file. Do not alter this file.

The **hpss_unmake** file contains information needed for the Unconfigure HPSS option. This file is in the **/opt/hpss/config** directory. Do not alter this file.

The **sfs_servers** file contains information needed for the Unconfigure Encina option. Each entry in this file has a name and the disks used for each SFS server. This file is also in the **/opt/hpss/config** directory. Do not alter this file.

Although it is recommended that the mkhpss script be used only once per node to configure the HPSS infrastructure, it can be run again to correct or modify the infrastructure configuration as needed. However, after the HPSS servers are configured, mkhpss should not be used to modify DCE, Encina, or any other data that may affect the configuration of the HPSS system.

The following steps are automatically performed at the beginning of each **mkhpss** execution:

1. Load in the environment variables defined in the **hpss_env** file and **hpss_env_defs.h** file.
2. Verify that the user has root authority.
3. Create and initialize directories and files required for running HPSS.
4. Edit the **mkhpss.data** file to record status messages and processed states.

5.5.1 Configure HPSS with DCE

This option can only be performed from a node with the HPSS All Servers package installed. This option should only be performed once in the cell. The System Administrator should propagate the generated HPSS server and client keytab files to all other HPSS nodes. Refer to Section 5.6: *Customize DCE Keytabs Files* on page 246 for more information. The following steps are automatically performed:

1. Remove the HPSS keytab files, if any exist.
2. Create DCE groups for the HPSS servers and clients.
3. Create DCE principals and accounts for HPSS servers.

4. Create keytab files for HPSS servers and clients.

5. Randomize the keys in the created server and client keytabs.

6. Set up the DCE CDS directory for HPSS servers.

7. Set up HPSS-related Extended Registry Attributes, and create **hpss_cross_cell_members** group.

5.5.2 Configure Encina SFS Server

This option configures and starts an SFS server. By performing this step (Configure Encina SFS Server) multiple times, multiple SFS servers can be configured. Multiple data volumes can be configured on a single SFS server, and logical volumes can be mirrored to increase the availability of the data they store.

The following steps are performed during the interactive process of configuring the server:

1. Creation of **encina_admin_group** and **encina_servers_group** DCE groups as well as principals, account, and the keytab file for this SFS server.

2. CDS directories for this SFS server are created

3. The ACL for the clearinghouse is updated

4. The SFS log volume is created

5. The **logarchive** directory is created

6. The SFS log volume is enabled

7. Media archiving is enabled (if desired)

8. SFS data volumes are created, and SFS data volumes are enabled

9. ACLs for this SFS server are set

10. The SFS server is started

If the server machine is running AIX, **mkhpss** will prompt the administrator as in the following:

```
<mkhpss> Prompt ==> Enter number of copies (1,2):
```

or

```
<mkhpss> Prompt ==> Enter number of copies (1,2,3):
```

depending on the number of disks available for the chosen volume group.

Use **2** or **3** to mirror the logical volume.

If the server machine is running Solaris, **mkhpss** will prompt the administrator to enter disk name to add mirror to the logical volume.

For Solaris server machines, the disk name should start with **/dev/rdsk**.

5.5.3 Manage SFS Files

The **mkhpss** script invokes the **managesfs** utility, a front-end to the **sfsadmin** commands provided by Encina, to allow the user to create and manage the SFS files created for HPSS. The Manage SFS Files option should not be selected until the SFS has been configured as discussed in Section 5.5.2: *Configure Encina SFS Server* on page 243.

The script will present a menu to allow the user to create, delete, empty, query, and list SFS files. Refer to Section 12.2.37: *managesfs — Manage HPSS Metadata Files Utility* on page 403 of the *HPSS Management Guide* for more information on the **managesfs** utility.

5.5.4 Setup FTP Daemon

The following steps are performed when the Setup FTP Daemon option is selected:

1. Prompt the user for the port number to be used by the HPSS FTP.
2. Add hpssftp and hpssftp-data entries to the **/etc/services** file.
3. Add the hpssftp entry to the **/etc/inetd.conf** file.
4. Set up HPSS FTP Daemon directories and template files.

5.5.5 Setup Startup Daemon

The following steps are performed if the Setup Startup Daemon option is selected:

1. Add the hpssd entry to the **/etc/inittab** file.
2. Copy the rc.hpss file to the **/etc** directory.

5.5.6 Add SSM Administrative User

The HPSS system is ready to be configured using SSM once the HPSS software is installed on the node, and the HPSS infrastructure components are configured. The **mkhpss** script will optionally allow the user to configure an SSM user with authority level of “administrator” to use SSM to configure the HPSS system.

The **mkhpss** script will invoke a subset of the HPSS User Management Utility (**hpssuser**) to create the SSM User ID and password, the required Sammi configuration data, and the required UNIX and DCE accounts (if they do not already exist) for the SSM user. Due to the system limit, the length of the user ID must not exceed eight characters. If this step is not desired at this point, refer to

Section 6.2.2: *SSM User Session Configuration and Startup* on page 253 for instructions to create an SSM user session at a later time.

Even though all SSM User IDs and their prerequisite accounts can now be created, we recommend that only an administrative SSM ID be created at this time. The hpssuser utility should be invoked to create a new HPSS account after the HPSS system is operational so that all desired accounts (UNIX, DCE, SSM, or FTP) for new HPSS users can be created at the same time.

5.5.7 Start Up SSM Servers/User Session

If requested, the **mkhpss** script will start up the SSM servers and the SSM administration session so that the user can configure the HPSS system. If this step is not desired at this point, refer to Section 6.2.1: *SSM Server Configuration and Startup* on page 253 and Section 6.2.2: *SSM User Session Configuration and Startup* on page 253 for instructions on starting up the SSM servers and the SSM user sessions at a later time.

5.5.8 Re-run hpss_env()

If requested, the **mkhpss** script will re-read the environment variable values defined in the **hpss_env** file and the **hpss_env_defs.h** file.

5.5.9 Un-configure HPSS

This option allows administrator to delete the existing HPSS configuration from the node.

If no SFS server is configured, this option will issue a warning message, and the user will have option to abort or continue with the unconfiguration.

If the user chooses to continue, **mkhpss** deletes the existing HPSS configuration and releases any resources obtained by the HPSS Infrastructure Configuration.

If SFS server is configured, this option will display the following unconfigure menu:

```
<mkhpss> Prompt ==>     Select Unconfiguration Option:

<mkhpss>                [1] Unconfigure encina
<mkhpss>                [2] Unconfigure an installation node

<mkhpss>                [M] Return to the Main Menu

<mkhpss> Reply ===> (Select Option [1-2, M]):
```

Option 1 on the Unconfiguration Menu allows the user to unconfigure SFS servers. If somehow an SFS server is partially configured this option should be used to unconfigure the SFS server so it can be configured again if desired. Option 1 will display the list of all SFS servers configured, and the administrator can pick one on the list to unconfigure:

```
<mkhpss> Prompt ==> Password for cell_admin:
Password must be changed!
```

```
<mkhpss> Prompt ==>        Select the sfs server to be unconfigured

                           1) encina/sfs/hpss
                           2) encina/sfs/hpss1

<mkhpss>                   [M] Return to the Main Menu
<mkhpss>                   [U] Return to the Unconfigure Menu

<mkhpss> Reply ===> (Select Option [1-2, M, U]):
```

Option 2 on the Unconfiguration Menu will issue the following warning message, and the user will have the option to abort or continue with the unconfiguration:

```
<mkhpss> WARNING => ABOUT TO UN-CONFIGURE HPSS! ! ! ! !
<mkhpss> WARNING => ABOUT TO UN-CONFIGURE HPSS! ! ! ! !
<mkhpss> WARNING => ABOUT TO UN-CONFIGURE HPSS! ! ! ! !
<mkhpss> WARNING => LOCAL HPSS CONFIGURATION METADATA WILL BE
DESTROYED! ! ! ! !
<mkhpss> WARNING => LOCAL HPSS CONFIGURATION METADATA WILL BE
DESTROYED! ! ! ! !
<mkhpss> WARNING => LOCAL HPSS CONFIGURATION METADATA WILL BE
DESTROYED! ! ! ! !
<mkhpss> WARNING => LOCAL HPSS CONFIGURATION METADATA WILL BE
DESTROYED! ! ! ! !
<mkhpss> WARNING => (i.e., Local Keytabs, Local SFS Files, ... etc.)
<mkhpss> Prompt ==> Do you wish to continue?
<mkhpss> Reply ===> (N)
```

If the user chooses to continue with the unconfiguration, **mkhpss** deletes the existing configuration (including Encina configuration) and releases any resources.

5.6 Customize DCE Keytabs Files

As part of the HPSS configuration with DCE, the **mkhpss** script created the following default DCE principals:

- hpss_bfs
- hpss_client_api
- hpss_cns
- hpss_dmg
- hpss_ftpd
- hpss_gk
- hpss_hpssd

- hpss_log
- hpss_ls
- hpss_mm
- hpss_mountd
- hpss_mps
- hpss_mvr
- hpss_ndcg
- hpss_nfs
- hpss_pvl
- hpss_pvr
- hpss_ssm
- hpss_ss

To adhere to the site local password policy, these principals were created with known keys and subsequently changed with randomized keys. The HPSS keytab files (**/krb5/hpss.keytabs** and **/ krb5/hpssclient.keytab**) hold both versions of the keys; however, the DCE Registry holds only the randomized keys.

Periodically, the HPSS administrator should change the passwords in the two HPSS keytab files.

The procedure to change the passwords is as follows:

1. List the contents of the two keytab files:

   ◆ Become root.

   ◆ **dce_login** as an entity allowed access to the DCE Registry for changing passwords, usually **cell_admin**.

   ◆ Issue the commands:

   ```
   % dcecp -c keytab list /.:/hosts/$HPSS_CDS_HOST/config/keytab/
   hpss.keytabs
   % dcecp -c keytab list /.:/hosts/$HPSS_CDS_HOST/config/keytab/
   hpssclient.keytab
   ```

2. Update the keytab files:

   ◆ For each entry in **/krb5/hpss.keytabs** do:

   ```
   % dcecp -c keytab \
      add /.:/hosts/$HPSS_CDS_HOST/config/keytab/hpss.keytabs \
      -member <entry_name> \
   ```

```
   -random \
   -registry
```

- For each entry in **/krb5/hpssclient.keytab** do:

```
% dcecp -c keytab add \
   /.:/hosts/$HPSS-CDS_HOST/config/keytab/hpssclient.keytab \
   -member <entry_name> \
   -random \
   -registry
```

  where **<entry_name>** refers to an entry in the keytab file; e.g., **hpss_ssm**, and **$HPSS_CDS_HOST** refers to the CDS machine host name; e.g., **hydra**.

3. See the discussion immediately following this step! Propagate the resulting keytab files to every HPSS server machine. Note that the most secure mechanism for performing this is "**footnet**". If FTP is used, be sure to specify the "**bin**" option. The keytab files on every HPSS system should have the following ownership and permissions set:

```
/krb5/hpss.keytabs          hpss hpss     rw- rw- ---
/krb5/hpssclient.keytab     hpss hpss     rw- rw- ---
```

It is strongly recommended that both keytab files be generated on a single HPSS server machine and securely propagated to every other HPSS server machine; however, a customer may prefer to create appropriate keytab files which contain only the entries required for a specific HPSS server machine. This, however, is strongly discouraged because it can create a "Catch 22" condition in which the encryption keys on one or more HPSS systems cannot be set to match the keys stored in the DCE Registry!

If a customized keytab file is used on every different HPSS server system, steps 1 and 2 above must be performed on each system.

If the key for a server on one machine is changed, do not change the key on another machine since this will de-synchronize the entry on the first system changed!

Chapter 6 HPSS Configuration

6.1 Overview

This chapter provides instructions for creating the configuration data to be used by the HPSS servers. This includes creating the server configuration, defining the storage policies, and defining the storage characteristics. The configuration data can be created, viewed, modified, or deleted using the HPSS SSM GUI windows. Refer to Appendix F: *Additional SSM Information* (page 525) for more information on how to use SSM.

To create or modify the HPSS configuration data, we recommend that the administrator be familiar with the information described for the HPSS planning process as documented in Chapter 2. The procedures described in this chapter assume that the HPSS installation and infrastructure configuration have been completed.

6.1.1 HPSS Configuration Roadmap

The following steps summarize the configuration for the HPSS system. It is very important that the steps be performed in the order listed. Each step is discussed in more detail in the referenced section.

1. Configure and start up SSM (Section 6.2: *SSM Configuration and Startup* on page 252)
2. Create global configuration (Section 6.3: *Global Configuration* on page 255)
3. Create storage subsystem configuration (Section 6.4: *Storage Subsystems Configuration* on page 259)
4. Create a basic configuration entry for each HPSS server (Section 6.5: *Basic Server Configuration* on page 262)
5. Configure HPSS storage policies (Section 6.6: *HPSS Storage Policy Configuration* on page 279)
6. Configure HPSS storage characteristics (Section 6.7: *HPSS Storage Characteristics Configuration* on page 305)
7. Update global and storage subsystem configurations (Section 6.3: *Global Configuration* on page 255 and Section 6.4: *Storage Subsystems Configuration* on page 259)

8. Create a specific configuration entry for each HPSS server (Section 6.8: *Specific Server Configuration* on page 323)

9. Configure MVR devices and PVL drives (Section 6.9: *Configure MVR Devices and PVL Drives* on page 401)

6.1.2 HPSS Configuration Limits

The following configuration limits are imposed by SSM and / or the HPSS servers:

6.1.2.1 Server

- Maximum number of HPSS servers: unlimited

6.1.2.2 Storage Policy

- Total Accounting Policies: 1
- Total Migration Policies: 64
- Total Purge Policies 64

6.1.2.3 Storage Characteristic

- Total Storage Classes: 384
- Total Storage Hierarchies: 64
- Total Classes Of Service: 64
- Maximum File Families: 1024

6.1.2.4 Mover Device/PVL Drive

- Maximum Devices / Drives: Unlimited
- Total Devices per Mover: 64

6.1.2.5 Storage Space

- Import: 10,000 cartridges per SSM import request
- Export: 10,000 cartridges per SSM export request
- Create: 10,000 physical volumes per SSM create request

- Delete: 10,000 physical volumes per SSM delete request

6.1.3 Using SSM for HPSS Configuration

The HPSS server and resource configuration data may be created, viewed, updated, or deleted using the SSM windows. The configuration data are kept in Encina SFS files.

When you submit a request to configure a new server, SSM displays all fields with the appropriate default data. If the default values were not used during the installation and the infrastructure configuration phases or are not desired, type over the displayed data with the desired values. Press the **Enter** key after entering the data so that the new value is read by SSM.

As each server general configuration entry is created, SSM keeps track of the configured server to aid the user in defining the remainder of the HPSS configuration and to allow the user to monitor and manage the servers during the HPSS operational phase.

This guide presents and describes the configuration data displayed in the SSM configuration windows in the form of tables. Each table lists the SSM window display field names and, for each field, gives a description of the variable, acceptable values for the variable, and the default value of the variable if one exists.

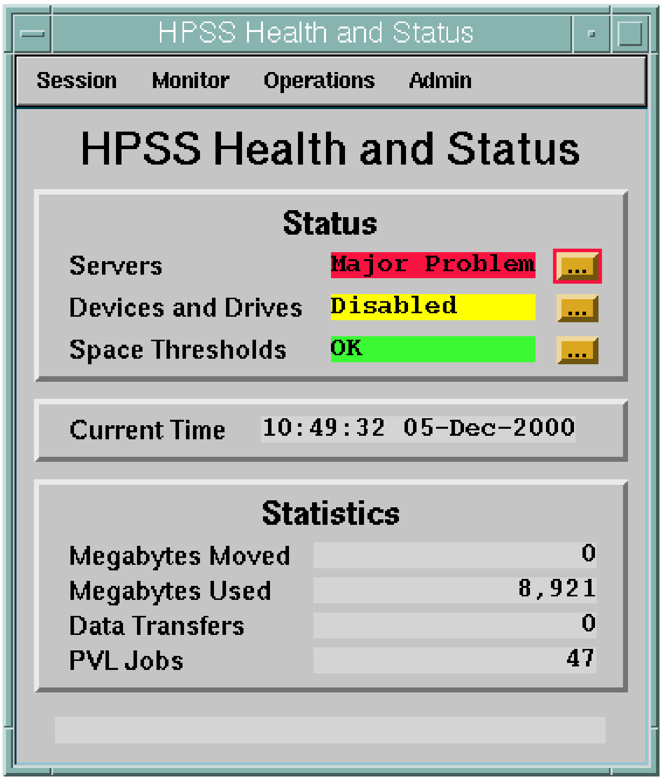
Many of the fields in the tables have an advice box that provides additional information about the field.

6.1.3.1 Using the Help Window

All of the SSM windows provide a pop-up help window that contains detailed information about the fields on the window. To display the help window, move the cursor to any blank space on the window, hold down the **Shift** key, and click the right-most mouse button.

6.1.3.2 Using the HPSS Configuration Windows

All HPSS configuration data can be created, viewed, updated, and deleted using the appropriate HPSS Configuration windows. You obtain these windows from the HPSS Health and Status window that appears when you log on to the system (as discussed in Section 6.2.2). When the HPSS Health and Status window (shown in Figure 6-1) appears, click on the **Admin** menu, select the **Configure HPSS** option and click on the appropriate configuration option. The desired configuration window will be displayed. Refer to the appropriate sections in this chapter for more information on how to use the HPSS configuration windows to configure an HPSS system.



HPSS Health and Status

Status

Servers	Major Problem
Devices and Drives	Disabled
Space Thresholds	OK

Current Time 10:49:32 05-Dec-2000

Statistics

Megabytes Moved	0
Megabytes Used	8,921
Data Transfers	0
PVL Jobs	47

Figure 6-1 HPSS Health and Status Window

6.1.4 *Server Reconfiguration and Reinitialization*

HPSS servers read their respective configuration file entries during startup to set their initial running conditions. Note that modifying a configuration file while a server is running does not immediately change the running condition of the server. Servers will need to perform a reinitialization or restart to read any newly modified configuration data. This reinitialization is usually accomplished by stopping and restarting the server. For servers that have the ability to reinitialize without restarting (i.e., the Metadata Monitor, Log Daemon, and Log Client) this can be accomplished by using the **Reinitialize Server** feature in the appropriate SSM windows.

6.2 *SSM Configuration and Startup*

If the SSM servers and the SSM administrative session were configured and started up during the HPSS infrastructure configuration phase, the administrator does not have to perform the steps described in this section. However, the information described in this section will be useful for the subsequent restart of the SSM servers or for the startup of other SSM sessions.

The SSM configuration consists of creating the configuration entry for the SSM System Manager, starting up the SSM servers, and bringing up one or more SSM user sessions. An SSM server startup script, **start_ssm**, is provided to bring up the SSM System Manager and the SSM Data Server. Another provided script, **start_ssm_session**, can be used to bring up an SSM user session.

6.2.1 SSM Server Configuration and Startup

The SSM System Manager will automatically create an SSM configuration entry, if one does not already exist, using the environment variables defined in the **/opt/hpss/config/hpss_env** file. This configuration entry may later be modified, if necessary, by the administrator. The SSM Data Server also uses the environment variables defined in the **/opt/hpss/config/hpss_env** file, but does not require a configuration entry. Refer to Section 5.3: *Define the HPSS Environment Variables* on page 216 for more information on the SSM variables.

The SSM server startup script, **start_ssm**, as supplied with HPSS can be used to invoke the SSM System Manager and the SSM Data Server. Both servers require a number of environment variables to be used as their command line arguments and to be used to configure and manage the HPSS servers. These variables are defined in the **/opt/hpss/config/hpss_env** file. They should have been edited to reflect the site's configuration during the HPSS infrastructure configuration phase.

To start up the SSM servers, invoke the **start_ssm** script. The System Manager will be brought up first, followed by the Data Server. The **start_ssm** script can also be invoked with the **-s** (start the System Manager only) option or the **-d** (start the Data Server only) option. Once the SSM servers are up and running, one or more SSM sessions can be started to manage HPSS.

Before starting up the SSM servers, ensure that DCE, Encina, and Sun RPC servers are up and running.

The start_ssm script will not start an SSM server if that server is already running on the same node. Ensure that only one copy of each configured SSM System Manager and Data Server is running at any one time. If there are multiple SSM servers running with the same configuration or environment data, there may be data loss for the SSM windows. Typically, there will be only one System Manager and one Data Server configured and both are run on the same node. If this setup is not desired, refer to Section F.6: *Non-standard SSM Configurations* on page 532 for more information on other possible configurations.

6.2.2 SSM User Session Configuration and Startup

An SSM administrator User ID should have been already created as part of the HPSS infrastructure configuration phase. If the User ID does not exist, refer to Section 8.1.1: *Adding HPSS Users* on page 215 of the *HPSS Management Guide* for instructions on adding users and creating an SSM account before proceeding with this section.

SSM supports the following SSM security levels:

1. **admin.** This security level is normally assigned to an HPSS administrator. This SSM user can view all SSM windows and perform all control functions provided by SSM.

2. **operator.** This security level is normally assigned to an HPSS operator. This SSM user can view all SSM windows and perform all SSM control functions except for changing the HPSS configuration.

3. **privileged.** This security level is normally assigned to a privileged user such as an HPSS system analyst. This SSM user can view most of the SSM windows but cannot perform any SSM control functions.

4. **user.** This security level is normally assigned to an user who may have a need to monitor some HPSS functions. This user can view a limited set of the SSM windows but cannot perform any of the SSM control functions.

Refer to Appendix F: *Additional SSM Information* (page 525) for more information on setting up an SSM account and the list of SSM windows accessible to each SSM user type.

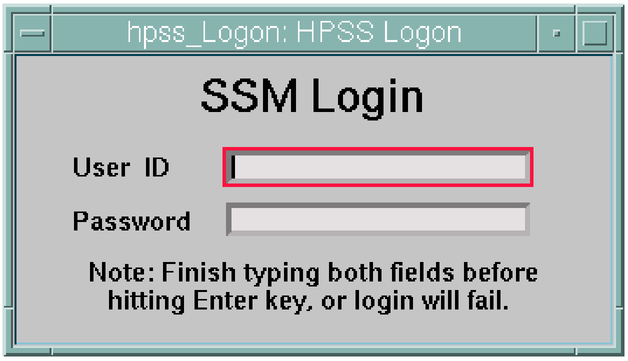
Before bringing up the SSM session, ensure that the Sammi license key provided by Kinesix has been properly set up. If it is not, refer to Section 4.5.3: *Set Up Sammi License Key* on page 213 for instructions on setting up the Sammi license.

In addition to the above, the SSM session may also need a Motif key bindings file to ensure that certain keyboard keys are defined in specific ways expected by Sammi. The key bindings file is named **.motifbind** and must be placed in the user's home directory on the host where the user will be displaying the SSM windows. Refer to Section F.4: *Customizing SSM and Sammi* on page 528 for more information on how to set up the Motif key bindings file.

The SSM user session **start_ssm_session** script as supplied with HPSS can be used to bring up any configured SSM session. To start up an SSM user session, invoke the startup script, **start_ssm_ session,** and enter the requested data when prompted. (The **start_ssm_session** script will not start an SSM session if that session is already up and running.) After a brief interval, the SSM Logon window (shown in Figure 6-2) will be displayed to allow the user to log on. Enter the valid DCE **User ID** and **Password** before pressing the **Enter** key. After the user is authenticated by DCE, the HPSS Health and Status window will be displayed. From this window, the user can proceed to perform the HPSS configuration.

Before starting up the SSM user session, ensure that the Sun RPC servers and the SSM servers are up and running.

There is no limit on the number of SSM sessions that can be brought up at the same time. However, depending on the system workload, too many running SSM sessions may degrade the overall system performance.


hpss_Logon: HPSS Logon

SSM Login

User ID

Password

Note: Finish typing both fields before hitting Enter key, or login will fail.

Figure 6-2 HPSS Logon Window

6.3 Global Configuration

The HPSS Global Configuration metadata record provides important information that is used by all HPSS servers. This is the first configuration that must be done through SSM.

6.3.1 Configure the Global Configuration Information

The global configuration information can be configured using the HPSS Global Configuration window. After the information has been created, it can be updated.

Due to the importance of the global configuration metadata, great care must be taken before updating it once the system is configured. If the global configuration is updated, all HPSS components, (including SSM) must be recycled.

It is possible to delete the global configuration metadata; however, this should never be done.

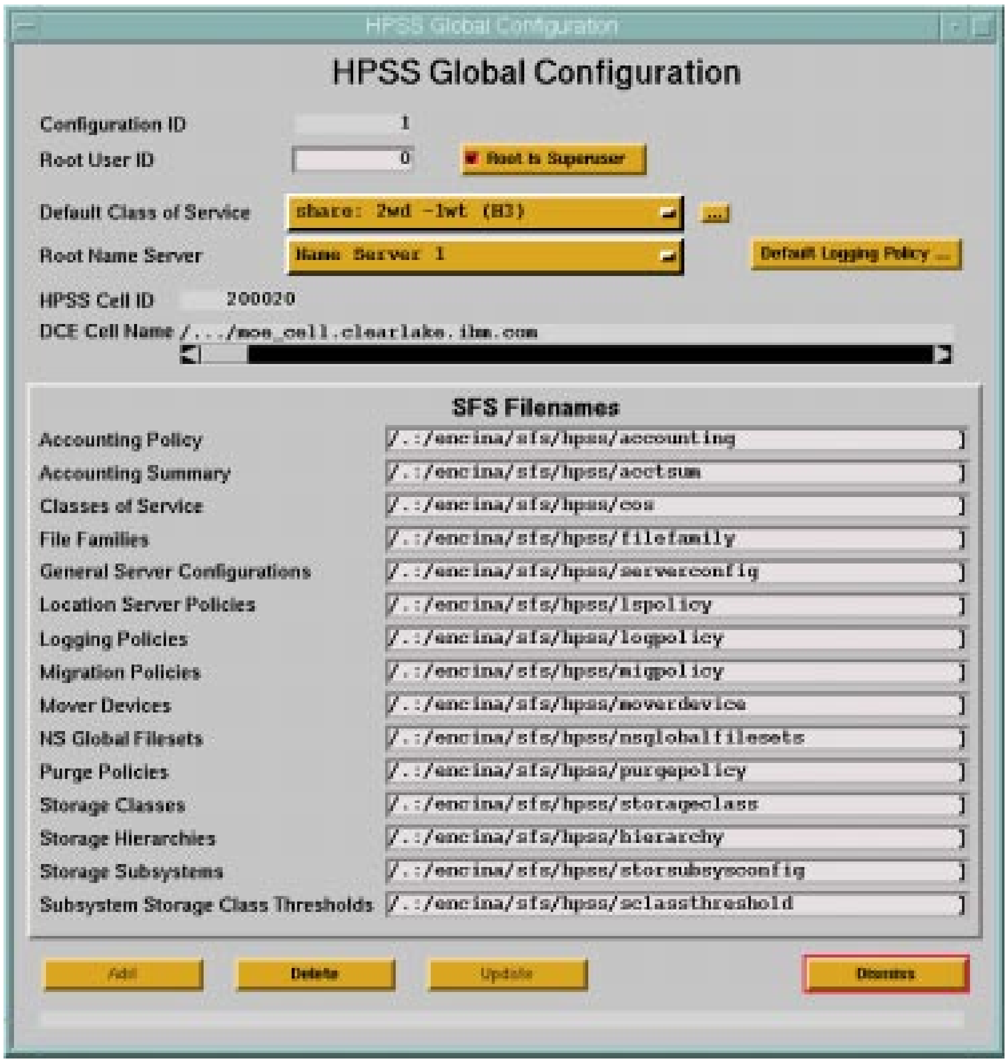
From the HPSS Health and Status window (shown in Figure 6-1 on page 252), click on the Admin menu, select the Configure HPSS option and click on the Global option. The HPSS Global Configuration window will be displayed as shown in Figure 6-3 on page 256.

If the "Add" button is sensitive (not grayed out) when this window opens, no global configuration exists yet, and you may create it. The fields contain default values. To create the configuration, modify the defaults if necessary, then click on the "Add" button. The result of the request will be displayed on the message line at the bottom of the window.

If the "Update" buttons is sensitive, the configuration already exists, and you may view or update it. To update the displayed configuration, modify the desired fields, then click on the "Update" button. The result of the request will be displayed on the message line.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

6.3.2 Global Configuration Variables



HPSS Global Configuration

Configuration ID	1
Root User ID	0
Default Class of Service	share: 2wd -1wt (B3)
Root Name Server	Name Server 1
HPSS Cell ID	200020
DCE Cell Name	/.../mos_cell.clearlake.ibm.com

SFS Filenames

Accounting Policy	/.:/encina/sfs/hpss/accounting
Accounting Summary	/.:/encina/sfs/hpss/acctsum
Classes of Service	/.:/encina/sfs/hpss/cos
File Families	/.:/encina/sfs/hpss/filefamily
General Server Configurations	/.:/encina/sfs/hpss/serverconfig
Location Server Policies	/.:/encina/sfs/hpss/lspolicy
Logging Policies	/.:/encina/sfs/hpss/logpolicy
Migration Policies	/.:/encina/sfs/hpss/migpolicy
Mover Devices	/.:/encina/sfs/hpss/moverdevice
NS Global Filesets	/.:/encina/sfs/hpss/nsglobalfilesets
Purge Policies	/.:/encina/sfs/hpss/purgepolicy
Storage Classes	/.:/encina/sfs/hpss/storageclass
Storage Hierarchies	/.:/encina/sfs/hpss/hierarchy
Storage Subsystems	/.:/encina/sfs/hpss/storsubsysconfig
Subsystem Storage Class Thresholds	/.:/encina/sfs/hpss/sclassthreshold

Figure 6-3 HPSS Global Configuration screen

Table 6-1 lists the Global Configuration variables and provides specific recommendations for the Global Configuration.

Table 6-1 Global Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
General Section			

Table 6-1 Global Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Root User ID	The UID of the user who has root access privileges to the NS database, if the **Root Is Superuser** flag is set to ON.	Valid root user ID	0
Root Is Superuser	A flag that indicates whether root privileges are enabled for the UID specified in the **Root User ID** field. Root access privileges grant the specified user the same access rights to a name space object as the owner of that object.	ON, OFF	ON
Default Class of Service	The name of the default Class of Service (COS). Bitfile Servers will store new files in this COS when the HPSS client does not specify a COS on the creation request. This default can be overridden for specific storage subsystems.	Any valid COS	None
	Advice: Since the Global Configuration record must be created before any servers are created, this field must initially be left blank. Once one or more Classes of Service have been created, the Global Configuration window can be reopened to Update the record, and a Default Class of Service can be selected. If no Default Class of Service is ever input into this field, the Bitfile Server will fail to start.		
Root Name Server	The Name Server which manages the root fileset (“/”) in the HPSS namespace.	Any Name Server	None
	Advice: Since the Global Configuration record must be created before any servers are created, this field must initially be left blank. Once one or more Name Servers have been created, the Global Configuration window can be reopened to Update the record, and a root Name Server can be selected. *The root Name Server should be selected with care. Once it is chosen, and the Update button is clicked, it cannot be changed as long as the chosen server exists. If the root Name Server MUST be changed, the current server's configuration will have to be deleted. SSM will then allow another root Name Server to be selected.*		

Table 6-1 Global Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Default Logging Policy	The logging policy that will be used by HPSS servers for which a specific logging policy has not been configured.	Any Logging policy or none	None
Accounting Policy	Name of the SFS file where the accounting policy is stored.	Any valid Encina filename	/.:/encina/sfs/hpss/accounting
Accounting Summary	Name of the SFS file where accounting summary information is stored.	Any valid Encina filename	/.:/encina/sfs/hpss/acctsum
Classes of Service	Name of the SFS file where class of service configurations are stored.	Any valid Encina filename	/.:/encina/sfs/hpss/cos
File Families	Name of the SFS file where file family information is stored.	Any valid Encina filename	/.:/encina/sfs/hpss/filefamily
General Server Configurations	Name of the SFS file where basic server configurations stored.	Any valid Encina filename	/.:/encina/sfs/hpss/serverconfig
Location Server Policies	Name of the SFS file where the location policy is stored.	Any valid Encina filename	/.:/encina/sfs/hpss/lspolicy
Logging Policies	Name of the SFS file where logging policies are stored.	Any valid Encina filename	/.:/encina/sfs/hpss/logpolicy
Migration Policies	Name of the SFS file where migration policies are stored.	Any valid Encina filename	/.:/encina/sfs/hpss/migpolicy
Mover Devices	Name of the SFS file where mover device information is stored.	Any valid Encina filename	/.:/encina/sfs/hpss/moverdevice
NS Global Filesets	Name of the SFS file containing the fileset ID, name, and server ID information for all filesets known to all local Name Servers.	Any valid Encina filename	/.:/encina/sfs/hpss/nsglobalfileset s
Purge Policies	Name of the SFS file where purge policy information is stored.	Any valid Encina filename	/.:/encina/sfs/hpss/purgepolicy

Table 6-1 Global Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Storage Classes	Name of the SFS file where storage class configurations are stored.	Any valid Encina filename	/.:/encina/sfs/ hpss/ storageclass
Storage Hierarchies	Name of the SFS file where storage hierarchy configurations are stored.	Any valid Encina filename	/.:/encina/sfs/ hpss/hierarchy
Storage Subsystems	Name of the SFS file where storage subsystem configurations are stored.	Any valid Encina filename	/.:/encina/sfs/ hpss/ storsubsysconfig
Subsystem Storage Class Thresholds	Name of the SFS file where storage subsystem-specific migration and purge thresholds are stored.	Any valid Encina filename	/.:/encina/sfs/ hpss/ sclassthreshold

6.4 Storage Subsystems Configuration

The storage subsystem configuration information can be configured using the **HPSS Storage Subsystem Configuration** window.

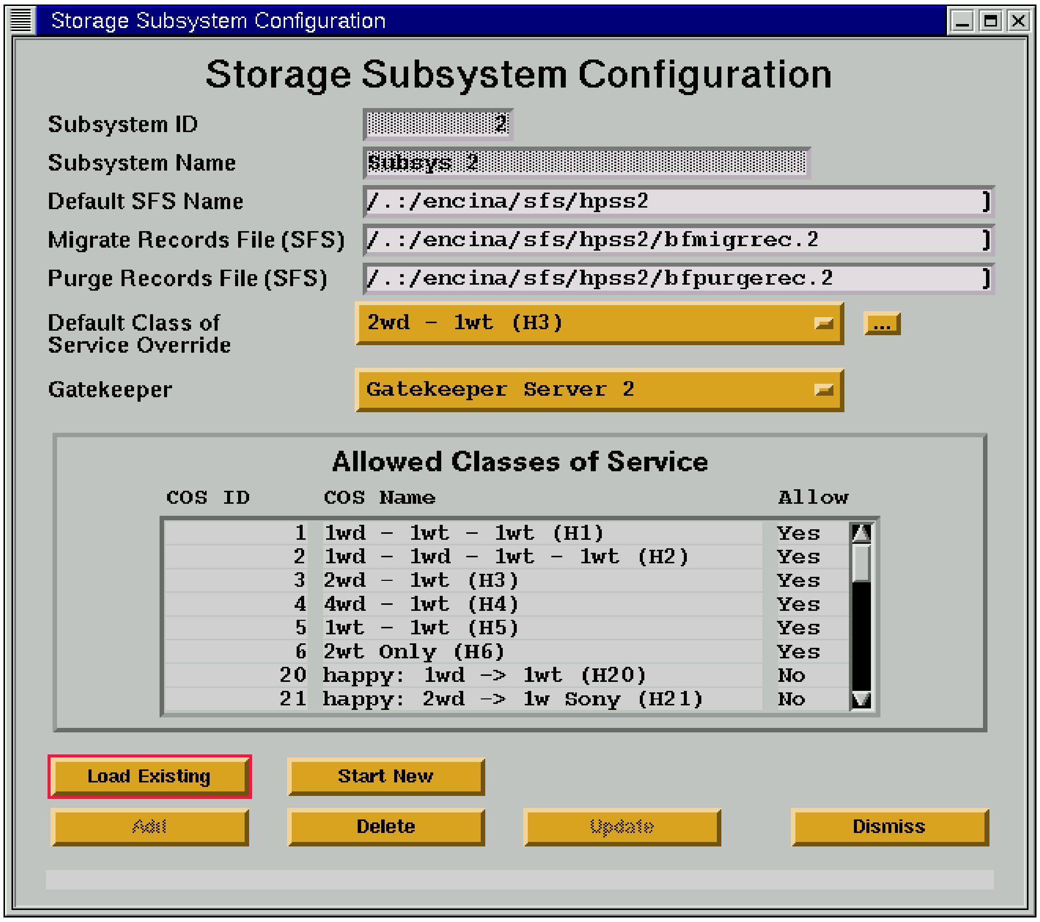
Due to the importance of the storage subsystem configuration metadata, great care must be taken before updating it once the system is configured. If the storage subsystem configuration is updated or deleted, all HPSS components, (including SSM) must be recycled.

From the **HPSS Health and Status** window (shown in Figure 6-1 on page 252), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Storage Subsystems** option. The **Storage Subsystem Configuration** window will be displayed as shown in Figure 6-4. The fields are displayed with default values for a new storage subsystem. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing storage subsystem, select the **Load Existing** button on the **Storage Subsystem Configuration** window and select the desired storage subsystem from the popup list. The window will be refreshed with the configuration data. After modifying the data, click the **Update** button to write the changes to the SFS file.

To delete an existing storage subsystem, select the **Load Existing** button on the **Storage Subsystem Configuration** window and select the desired storage subsystem from the popup list. The window will be refreshed with the configuration data. Click on the **Delete** button to delete the storage subsystem.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.



Storage Subsystem Configuration

Subsystem ID	2
Subsystem Name	Subsys 2
Default SFS Name	/.:/encina/sfs/hpss2
Migrate Records File (SFS)	/.:/encina/sfs/hpss2/bfmigrrec.2
Purge Records File (SFS)	/.:/encina/sfs/hpss2/bfpurgerec.2
Default Class of Service Override	2wd - 1wt (H3)
Gatekeeper	Gatekeeper Server 2

Allowed Classes of Service

```
COS ID  COS Name                        Allow
     1  1wd - 1wt - 1wt (H1)            Yes
     2  1wd - 1wd - 1wt - 1wt (H2)      Yes
     3  2wd - 1wt (H3)                  Yes
     4  4wd - 1wt (H4)                  Yes
     5  1wt - 1wt (H5)                  Yes
     6  2wt Only (H6)                   Yes
    20  happy: 1wd -> 1wt (H20)         No
    21  happy: 2wd -> 1w Sony (H21)     No
```

Figure 6-4 Storage Subsystem Configuration window

6.4.1 *Storage Subsystem Configuration Variables*

Table 6-2 lists the **Storage Subsystem Configuration Variables** and provides specific recommendations.

Table 6-2 Storage Subsystem Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Subsystem ID	A unique integer ID for the storage subsystem. This field may only be modified at create time for the storage subsystem.	Any non-zero, positive integer value.	Last configured subsystem ID + 1

Table 6-2 Storage Subsystem Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Subsystem Name	The descriptive name of the storage subsystem. This field may only be modified at create time for the storage subsystem.	A unique character string up to 31 bytes in length.	"Subsystem #N" where N is the subsystem ID
Migrate Records File (SFS)	The name of the SFS file where migration records are stored for this storage subsystem.	Any complete SFS file name.	/.:/encina/sfs/hpss/bfmigrrec.N where "N" is the storage subsystem id
Purge Records File (SFS)	The name of the SFS file where purge records are stored for this storage subsystem.	Any complete SFS file name.	/.:/encina/sfs/hpss/bfpurgerec.N where "N" is the storage subsystem id
Default Class of Service Override	A class of service id which overrides the default class of service specified on the global configuration screen for this storage subsystem only.	Any valid class of service id.	default: none (no override value)
Gatekeeper	The descriptive name of the Gatekeeper server for this subsystem.	The descriptive name of any valid gatekeeper.	default: none (no gatekeeper specified)
	Advice: It is very likely that you will configure a subsystem before configuring a Gatekeeper for it. In that case, simply leave this configuration entry blank initially, create the Gatekeeper configuration, then update the subsystem configuration with the new Gatekeeper.		
Allowed Classes of Service	A list of all of the classes of service in the system which allows individual classes of service to be enabled and disabled for the given subsystem.	Yes=COS Is Enabled, No=COS Is Disabled	Yes/No
	Advice: To toggle between "Yes" and "No" for a particular COS, simply left-click the existing "Yes"/"No" value. Note that a newly created COS will not appear in the selection list until the BFS, MPS, and SSs associated with the subsystem have been recycled. *Note: When new COSs are added, the initial "Yes"/"No" value for that COS is determined by the current setting for the other COSs. If all previous COSs were set to "Yes", the default for new COSs will be "Yes". Otherwise, the default will be "No".*		

6.5 Basic Server Configuration

All HPSS servers use a similar metadata structure for the basic server configuration. A basic server configuration entry must be created for each of the following server types:

- Bitfile Server
- DMAP Gateway
- Gatekeeper Server
- HPSS Startup Daemon
- Location Server
- Log Client
- Log Daemon
- Metadata Monitor
- Migration / Purge Server
- Mover
- Name Server
- NFS Daemon
- NFS Mount Daemon
- Non-DCE Client Gateway
- Physical Volume Library
- Physical Volume Repository
- Storage Server

Before configuring the HPSS servers, the planning guidelines for the HPSS servers as described in Section 2.6: *HPSS Server Considerations* on page 62 should be carefully considered.

The SSM System Manager basic configuration entry is usually created when the server is started up for the first time using the environment variables defined by the /opt/hpss/config/hpss_env script. If the default values are not appropriate for the site's configuration, the administrator should modify them using the Basic Server Configuration window described in the following section and restart the System Manager rather than re-create a new entry. It is recommended that only one SSM System Manager configuration entry be created to avoid possible loss of communication between the HPSS servers and SSM.

6.5.1 Configure the Basic Server Information

A basic server configuration entry can be created using the Basic Server Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through this window.

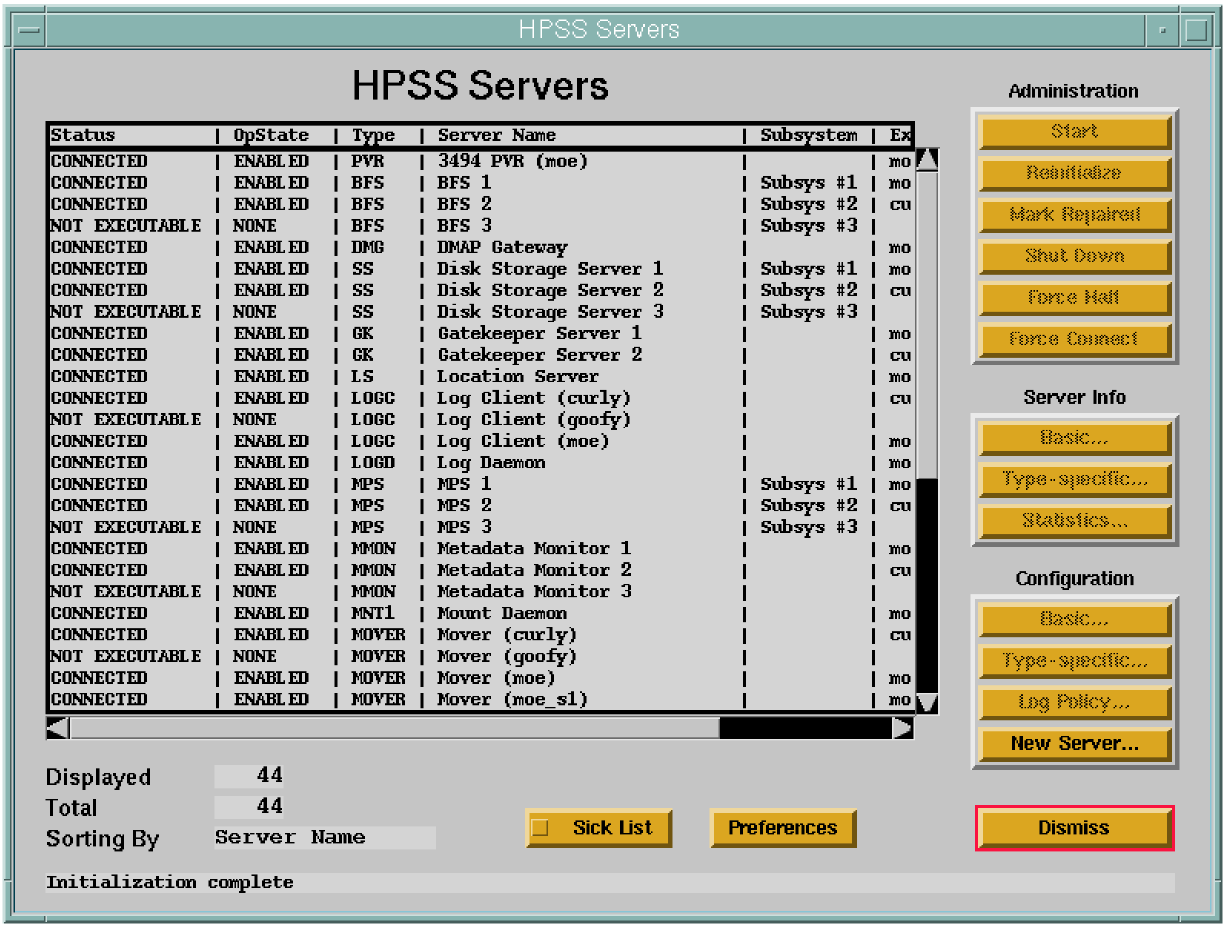
From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 6-5.

To configure a new server, click on the **New Server...** button on the HPSS Servers window. A blank Basic Server Configuration window is displayed along with the server-type popup list. Select the desired type of server and the Basic Server Configuration window will be filled in as shown in Figure 6-6 with default values. If the default data is not desired, change the field with the desired value. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the desired server entry on the HPSS Servers window and click on the **Basic...** button from the **Configuration** button group. The Basic Server Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the desired server displayed on the HPSS Servers window and click on the **Basic...** button from the **Configuration** button group. The Basic Server Configuration window will be displayed with the configured data. Verify that the associated server- specific configuration, if exists, is already deleted then click on the **Delete** button to delete the basic configuration entry.

Refer to the window’s help file for more information on the individual fields and buttons as well as the supported operations available from the window.



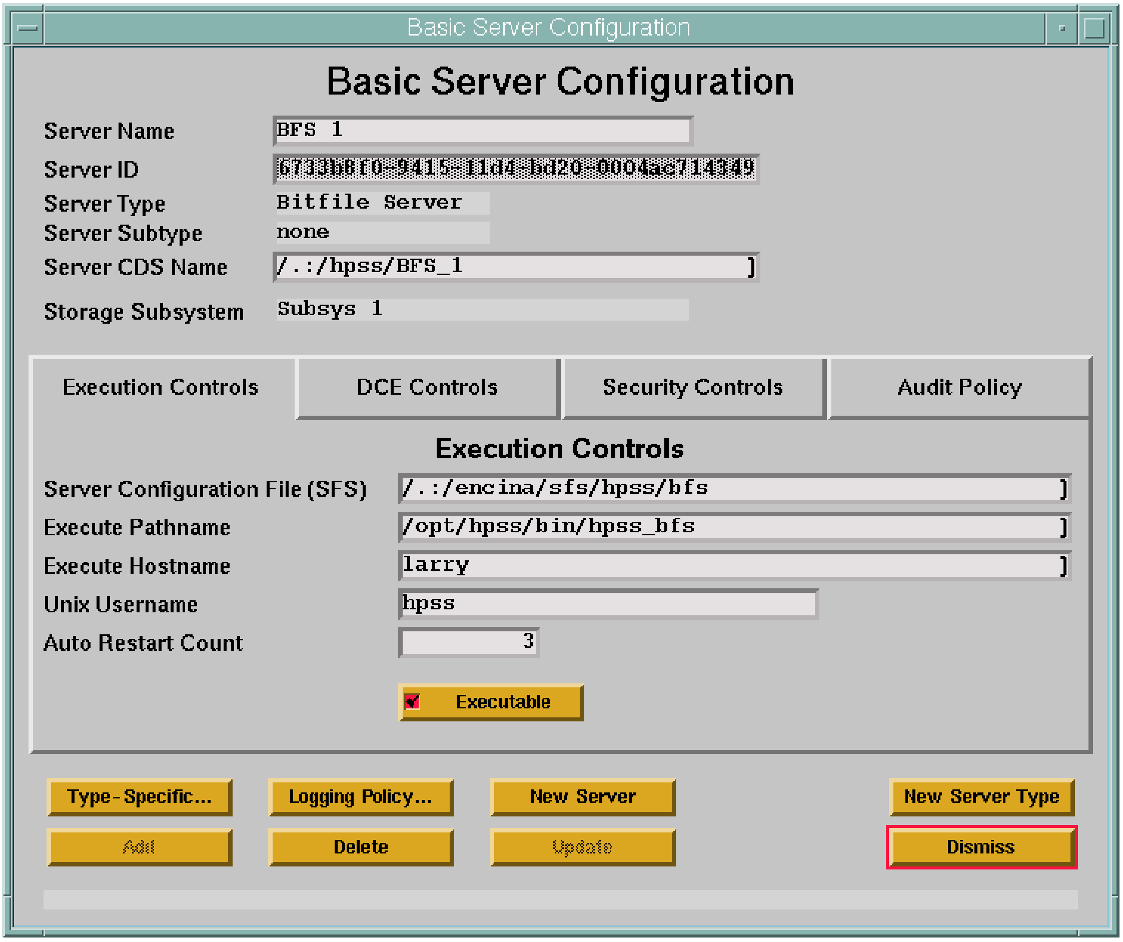
HPSS Servers

```
Status           | OpState  | Type  | Server Name                  | Subsystem  | Ex
CONNECTED        | ENABLED  | PVR   | 3494 PVR (moe)               |            | mo
CONNECTED        | ENABLED  | BFS   | BFS 1                        | Subsys #1  | mo
CONNECTED        | ENABLED  | BFS   | BFS 2                        | Subsys #2  | cu
NOT EXECUTABLE   | NONE     | BFS   | BFS 3                        | Subsys #3  |
CONNECTED        | ENABLED  | DMG   | DMAP Gateway                 |            | mo
CONNECTED        | ENABLED  | SS    | Disk Storage Server 1        | Subsys #1  | mo
CONNECTED        | ENABLED  | SS    | Disk Storage Server 2        | Subsys #2  | cu
NOT EXECUTABLE   | NONE     | SS    | Disk Storage Server 3        | Subsys #3  |
CONNECTED        | ENABLED  | GK    | Gatekeeper Server 1          |            | mo
CONNECTED        | ENABLED  | GK    | Gatekeeper Server 2          |            | cu
CONNECTED        | ENABLED  | LS    | Location Server              |            | mo
CONNECTED        | ENABLED  | LOGC  | Log Client (curly)           |            | cu
NOT EXECUTABLE   | NONE     | LOGC  | Log Client (goofy)           |            |
CONNECTED        | ENABLED  | LOGC  | Log Client (moe)             |            | mo
CONNECTED        | ENABLED  | LOGD  | Log Daemon                   |            | mo
CONNECTED        | ENABLED  | MPS   | MPS 1                        | Subsys #1  | mo
CONNECTED        | ENABLED  | MPS   | MPS 2                        | Subsys #2  | cu
NOT EXECUTABLE   | NONE     | MPS   | MPS 3                        | Subsys #3  |
CONNECTED        | ENABLED  | MMON  | Metadata Monitor 1           |            | mo
CONNECTED        | ENABLED  | MMON  | Metadata Monitor 2           |            | cu
NOT EXECUTABLE   | NONE     | MMON  | Metadata Monitor 3           |            |
CONNECTED        | ENABLED  | MNT1  | Mount Daemon                 |            | mo
CONNECTED        | ENABLED  | MOVER | Mover (curly)                |            | cu
NOT EXECUTABLE   | NONE     | MOVER | Mover (goofy)                |            |
CONNECTED        | ENABLED  | MOVER | Mover (moe)                  |            | mo
CONNECTED        | ENABLED  | MOVER | Mover (moe_s1)               |            | mo
```

Displayed	44
Total	44
Sorting By	Server Name

Initialization complete

Figure 6-5 HPSS Servers Window



Basic Server Configuration

Server Name	BFS 1
Server ID	6733b0f0-9415-11d4-bd20-0004ac714349
Server Type	Bitfile Server
Server Subtype	none
Server CDS Name	/.:/hpss/BFS_1
Storage Subsystem	Subsys 1

Execution Controls

Server Configuration File (SFS)	/.:/encina/sfs/hpss/bfs
Execute Pathname	/opt/hpss/bin/hpss_bfs
Execute Hostname	larry
Unix Username	hpss
Auto Restart Count	3

[✓ Executable]

Figure 6-6 Basic Server Configuration Window

6.5.1.1 *Basic Server Configuration Variables*

The fields in the Server Configuration window describe information necessary for servers to successfully operate. The fields also contain information that is necessary for successful interaction with the SSM component.

In addition, each HPSS basic server configuration includes Security Information and Audit Policy fields that determine the server's security environment. The security of the HPSS system will be a result of how these fields are set. HPSS defaults for the Security Information are set to provide authenticated communication between servers, authorization of clients to HPSS interfaces and objects, enforcement of authorization permissions, and audit of authentication and file operation failures.

The fields in the Basic Server Configuration window are grouped into five categories:

- General Information fields

- Execution Controls fields
- DCE Controls fields
- Security Controls fields
- Audit Policy fields

To save window space, the last four categories are presented in “layers”, and each layer has its name displayed on a “tab”. To access a different layer, click on the appropriate tab. Table 6-3 lists the fields on the Basic Server Configuration window in the approximate order that they appear on the window.

Table 6-3 Basic Server Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
General Information. The following fields describe the server’s general information.			
Server Name	The descriptive name used to look up a server’s configuration parameters in Encina SFS. As with UUIDs, each descriptive name in HPSS must be unique.	Any character string up to 31 bytes in length.	Based on the server type and the number of servers of that type that already exist.
	Advice: Ensure that the Descriptive Name is unique. A server’s descriptive name should be meaningful to local site administrators and operators, in contrast to the server’s corresponding UUID, which has meaning for DCE and HPSS. For HPSS systems spanning more than one subsystem, it is very helpful to append the subsystem ID to the Server Name of subsystem-specific servers. For instance, “BFS_2” for the BFS in subsystem 2.		
Server ID	A unique ID used to distinguish servers in the HPSS. Each server must have a unique Server ID.	Any valid non-null UUID.	A new and unique ID is generated by SSM during the configuration process as a default.
	Advice: Ensure that each server has a unique Server ID. If a server ID is not unique, SSM will not be able to distinguish servers and will not be able to correctly map between human-oriented descriptive names and machine-oriented UUIDs for display purposes.UUIDs are meaningful primarily to the storage system and DCE. Normally, they will not need to be directly specified or changed from SSM.		
Server Type	The type of HPSS server.	This field is filled in with the server type selected by the user as part of the window’s selection. It is not changeable.	Server type selected by the user.

Table 6-3 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Server Subtype	The subtype of the selected server. Most servers do not have subtypes.	This field is filled in with the server subtype selected by the user as part of the window's selection. It is not changeable.	Server subtype selected by the user (e.g. tape); none for servers that do not have subtypes.
Server CDS Name	The name of a Cell Directory Service (CDS) directory in which a server creates additional CDS objects and registers its endpoint information. Every server must have a different directory.	Any legal CDS directory name <128 bytes is permitted. The directory must exist when this particular server is started.	Based on **$HPSS_CDS_PREFIX** and a modification of the descriptive name from the Server Name field. All characters in the descriptive name except letters, numerals, and periods are changed to underscore to insure that the name is valid for DCE.
	Advice: *(1) Different names must be used for each server. If this is not observed, servers will have trouble contacting each other, or will talk to different servers than anticipated.* *(2) The directories must be writable by the process that uses them.* *(3) It is useful to have a convention for naming these directories. For example, all directories might be subdirectories of a common directory such as /.:/hpss. The directories might be named by the role that the corresponding server plays in the CDS—e.g., /.:/hpss/disk_ss1 and /.:/hpss/tape_ss1 might be the names used by two instances of the Storage Server.* *(4) If there are multiple Startup Daemons, Movers, and Log Clients running, use names that reflect the hosts they are running on. For example, a Startup Daemon running on a host named timelord might be named /.:/hpss/hpssd_timelord.* *(5) HPSS does not make any assumptions about which names are used.* *(6) SSM creates the specified CDS directory and the associated CDS Security Object when the server is configured from this window. The parent directory (e.g. /.:/hpss) must already exist, otherwise the creates will fail. See the discussion following this table for the particular ACLs.*		

Table 6-3 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Storage Subsystem	Name of the HPSS Storage Subsystem to which this server will be assigned. No more than one BFS, MPS, NS, Disk SS, and Tape SS can be assigned to any one subsystem.	Any configured Storage Subsystem name from the pop-up list. This field is required for BFS, MPS, NS, Disk SS, and Tape SS. For other server types, the field is blank and cannot be changed. This field can only be modified in the Add mode.	None
Execution Controls. The following fields describe the server's execution control configuration.			
Server Configuration File (SFS)	The name of the Encina SFS file containing configuration data for this server.	Any valid SFS filename.	Based on the selected server type.
Execute Pathname	The UNIX file system path name to a server's executable. If servers are running on several hosts, the name must be the name of a file on the host that is running the server.	Any legal UNIX file name.	/opt/hpss/bin/ <server-specific executable name>. Based on the selected server type.
	Advice: Use the full UNIX path name; otherwise, the Startup Daemon will try to start the file out of the current working directory of the Daemon.		

Table 6-3 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Execute Hostname	The hostname on which a particular server is to run. SSM uses this field to locate the Startup Daemon that will execute the server.	Any legal hostname, such as a name that might be obtained using the UNIX **hostname** command.	Local hostname.
	Advice: In order for a server to start, a Startup Daemon must be running. The server will be started on the host which has a startup daemon running whose configuration has a Execute Hostname field which matches exactly the one specified in the server's basic configuration record. For Movers which will execute in non-DCE mode, this field should contain the hostname corresponding to the node on which the Mover DCE/Encina processes will run. More correctly, there MUST be a startup daemon running on the node where the Mover DCE/Encina processes are running which specifies the same exact Execute Hostname as is specified in the non-DCE Mover's basic configuration. *Note: The SSM does not check if the server or the Startup Daemon is actually running on the specified host. It simply compares the Execute Hostname of the server with the Execute Hostname of each startup daemon until it finds a match.*		
UNIX Username	The UNIX user name that the server runs under. If servers are running on several hosts, the name must be the name that is used on the host that the server will run on. The name must be registered in the local UNIX authentication database (e.g., **/etc/passwd**).	Any legal user name for the host that the server runs on.	**root** for the Startup Daemon, NFS Daemon, NFS Mount Daemon, DMAP Gateway. **hpss** for all other servers (see Advice below).
	Advice: If there is no such user, the Startup Daemon will not be able to start the servers. The Startup Daemon, NFS Daemon, NFS Mount Daemon, DMAP Gateway, and Non-DCE Gateway should have their UNIX Usernames set to root, so do not modify the default for these servers.		

Table 6-3 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Auto Restart Count	The HPSS Startup Daemon uses this field to control automatic server restarts. If the server shuts down unexpectedly, the Startup Daemon will restart the server, without any intervention by SSM, up to this many times; after that, the Startup Daemon will not restart it again. If the server runs for more than an hour without failing, the Startup Daemon will set its failure count for that server back to zero. A negative value in this field will be interpreted as an infinite count; the Startup Daemon will always restart the server if it fails.	Any 32-bit integer value.	3
Executable	A flag that indicates whether a HPSS server can be started up by SSM.	ON, OFF	ON
DCE Controls. *The following fields describe the server's DCE controls information.*			

Table 6-3 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Maximum Connections	The highest number of connection contexts this server can establish.	Any positive 32-bit integer value.	BFS - 200 DMG- 100 GK - 200 Log Client - 10 Log Daemon- 10 LS - 20 MMON - 10 Mount Daemon - 10 MPS - 10 MVR - 20 NDCG -20 NFS Daemon - 20 NS - 200 PVL - 20 PVR - 20 SS - 20 SSM - 100 Startup Daemon - 10
	Advice: This value should be set based on the anticipated number of concurrent clients. Too large a value may slow down the system.		

Table 6-3 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Thread Pool Size	The highest number of threads this server can spawn in order to handle concurrent requests.	Any positive 32-bit integer value.	BFS - 200 DMG - 100 GK - 200 LS - 20 Log Client - 10 Log Daemon- 10 MMON - 10 Mount Daemon - 10 MPS - 10 MVR - 20 NS - 200 NDCG - 20 NFS Daemon - 20 PVL - 100 PVR - 30 SS - 50 SSM - 100 Startup Daemon - 10
	Advice: If necessary, the default values can be changed when defining servers. Too large a value may slow down the system. The Thread Pool Size should be equal to or larger than the value used for Maximum Connections. The SSM System Manager does not use this field. It uses the **ENCINA_TPOOL _SIZE** environment variable defined by the **/opt/hpss/config/hpss_env** script. The **ENCINA_TPOOL_SIZE** variable is initially set to 100 for the System Manager and can be changed if desired. Ensure that the value of the **HPSS_SFS_THREADS** environment variable defined by the **/opt/hpss/config/hpss_env** script is greater than the Thread Pool Size value defined for the Bitfile Server and the Name Server or Encina SFS may run out of threads.		
Security Controls. The following fields describe the server's security configuration.			

Table 6-3 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Principal Name	The DCE principal name defined for the server during the infrastructure configuration phase.	The principal name must exist in the DCE registry.	BFS: hpss_bfs DMG: hpss_dmg GK: hpss_gk Log Client: hpss_log Log Daemon: hpss_log LS: hpss_ls MMON: hpss_mmon Mount Daemon: hpss_mountd MPS: hpss_mps MVR: hpss_mvr NDCG: hpss_ndcg NS: hpss_cns NFS Daemon: hpss_nfs PVL: hpss_pvl PVR: hpss_pvr SS: hpss_ss SSM: hpss_ssm Startup Daemon: hpss_hpssd
	Advice: Ensure that the key table named by the Keytab Pathname field contains an entry for this principal; otherwise, authentication will fail.		
Protection Level	The amount of encryption that will be used in communication with peer applications.	Default, None, Connect, Call, Packet, Packet Integrity, Packet Privacy	Connect
	Advice: The higher the level of protection, the more encryption and overhead required in communications with peers. A level of None corresponds to no authentication and no authorization. The minimum suggested level is Connect. Refer to the description of the DCE Protection Levels included in the rpc_binding_set_auth_info call in the OSF/DCE Application Development Reference Manual.		

Table 6-3 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Authorization Service	The authorization service to use when passing identity information in communications to HPSS components.	None, Name, DCE	DCE
	Advice: The recommended authorization server is DCE. This ensures that the most complete identity information about the client is sent to the server. If None is used, anyone is authorized to call the HPSS components. This means you are in a trusted environment. If the Authentication Service value is None, the only acceptable Authorization Service value is None. If the Authentication Service value is Secret, the Authorization Service value can be Name or DCE. Refer to the description of the DCE Authorization Services included in the rpc_binding_set_auth_info call in the OSF/DCE Application Development Reference Manual.		
Authentication Service	The authentication service to use in communications.	None, Secret, Public, Default	Secret
	Advice: Secret provides for authenticated communications between HPSS components. The authentication service is the basis for all HPSS security. Refer to the description of the DCE Authentication Services included in the rpc_binding_set_auth_info call in the OSF/DCE Application Development Reference Manual.		
Security Site Name	The security services site name.	NULL or DCE site name where the security services may be obtained.	NULL
	Advice: If NULL is used, the local security site is used.		
Security Registry Scope	The security registry search scope for HPSS principal names.	NULL or DCE security registry directory.	NULL
	Advice: If NULL is used, the entire security registry is searched for the principal name. If your site has other DCE applications separate from HPSS, you may want to limit the search scope to an HPSS security registry directory.		

Table 6-3 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Keytab Pathname	The absolute pathname of the UNIX file containing the keytab entry that will be used by the server when setting up its identity.	Any legal UNIX file name can be used as long as it is the name of a keytable file.	/krb5/ hpss.keytabs
	Advice: The server must have read access to this file. Do not set other access permissions on this file or your security can be breached. Notes: (1) Each server can have its own key file, or all the servers can share a single key file. It is recommended that one key file be used for all of the servers on any given platform.(2) To use the standard DCE system wide key file, set this value to /krb/ v5srvtab (not recommended).		
Authentication Service Arg	The argument passed to the authentication service indicated by the Authentication Service configuration variable and used by the authentication service to validate communications. Currently, the only authentication services supported are **none** and **dce**.	NULL or any UNIX pathname that points to a key file.	/krb5/hpss.keytabs
	Advice: If dce authentication is to be used and the Keytab Pathname is /krb/ v5srvtab, set this variable to NULL. If dce authentication is to be used and the Keytab Pathname is not /krb/v5srvtab, set this variable to the value of Keytab Pathname. In either case, the server must have read access to the file. Do not set other permissions on this file or your security can be breached. If no authentication is to be used, set this value to NULL.		
Audit Policy. The following fields describe the server's audit policy configuration.			
AUTH	The Security Audit Policy for Authentication events. If set, security audit messages will be sent to the logging subsystem.	NONE- No audit messages will be generated. FAILURE - Audit messages will only be generated when there are errors. ALL - Audit messages will be generated for all related operations.	FAILURE
	Advice: Sites that must audit all login type events should set this value to ALL.		

Table 6-3 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
CHMOD	The Security Audit Policy for Name Server Object Permissions events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Name Server; NONE for other servers
CHOWN	The Security Audit Policy for Name Server Object Owner events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Name Server; NONE for other servers
CREAT	The Security Audit Policy for Name Server Bitfile Creation events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Name Server; NONE for other servers
	Advice: Sites that must audit object creation should set the CREAT field to ALL for Name Server.		
LINK	The Security Audit Policy for Name Server Hard Link Creation events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Name Server; NONE for other servers
	Advice: Sites that must audit object creation should set the LINK field to ALL for Name Server.		
MKDIR	The Security Audit Policy for Name Server Directory Creation events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	NONE
	Advice: Sites that must audit object creation should set the MKDIR field to ALL for Name Server.		
OPEN	The Security Audit Policy for Bitfile Server Bitfile Open events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Bitfile Server; NONE for other servers

Table 6-3 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
RENAME	The Security Audit Policy for Name Server Object Rename events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Name Server; NONE for other servers
	Advice: Sites that must audit object deletion should set the RENAME field to ALL for Name Server.		
RMDIR	The Security Audit Policy for Name Server Directory Remove events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Name Server; NONE for other servers
	Advice: Sites that must audit object deletion should set the RMDIR field to ALL for Name Server.		
UNLINK	The Security Audit Policy for Name Server Object Delete events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Name Server; NONE for other servers
	Advice: Sites that must audit object creation should set the UNLINK field to ALL for Name Server.		
UTIME	The Security Audit Policy for Bitfile Time Modified events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Bitfile Server and Name Server; NONE for other servers
ACL_SET	The Security Audit Policy for Name Server Access Control List Modification events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Name Server; NONE for other servers
CHBFID	The Security Audit Policy for Name Server Change Bitfile Identifier events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Name Server; NONE for other servers

Table 6-3 Basic Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
BFSETATTRS	The Security Audit Policy for Bitfile Server Set Bitfile Attribute events. If set, security audit messages will be sent to the logging subsystem.	NONE, FAILURE, ALL	FAILURE for Bitfile Server; NONE for other servers

6.5.1.2 Server CDS Security ACLs

The server's CDS directory inherits ACLs from the parent but SSM adds the following ACL for the HPSS Server Group:

```
group:${HPSS_GRP_NAME_SERVER}:rwdtcia
```

SSM creates the default ACLs for the server's CDS Security object as follows:

CDS Security object for the BFS Security object:
```
{user:${HPSS_PRINCIPAL_NS}:rwdtc}
{user:${HPSS_PRINCIPAL_DMG}:rwdtc}
{user:${HPSS_PRINCIPAL_MPS}:rwdtc}
{user:${HPSS_PRINCIPAL_NFSD}:rwdtc}
{user:${HPSS_PRINCIPAL_SSM}:rwdtc}
{group:subsys/dce/cds-admin:rwdtc}
{group:subsys/dce/cds-server:rwdtc}
{any_other:---t-}
```

CDS Security object for the DMG Security object:
```
{user:${HPSS_PRINCIPAL_FTPD}:r---c}
{user:${HPSS_PRINCIPAL_BFS}:r---c}
{user:${HPSS_PRINCIPAL_NDCG}:r---c}
{user:${HPSS_PRINCIPAL_NFSD}:r---c}
{user:${HPSS_PRINCIPAL_SSM}:rw--c}
{user:cell_admin:rwdtc}
{any_other:---t-}
```

CDS Security object for the GK Security object:
```
{user:${HPSS_PRINCIPAL_BFS}:rw---}
{user:${HPSS_PRINCIPAL_NS}:rw---}
{user:${HPSS_PRINCIPAL_SSM}:rw--c}
{group:subsys/dce/cds-admin:rwdtc}
{group:subsys/dce/cds-server:rwdtc}
{any_other:---t-}
```

CDS Security object for the LS Security object:
```
{user:${HPSS_PRINCIPAL_SSM}:r--tc}
{group:subsys/dce/cds-admin:rwdtc}
```

```
{group:subsys/dce/cds-server:rwdtc}
{any_other:r--t-}
```

CDS Security object for the MVR Security object:

```
{user:${HPSS_PRINCIPAL_SSM}:rw-tc}
{group:subsys/dce/cds-admin:rwdtc}
{group:subsys/dce/cds-server:rwdtc}
{any_other:---t-}
```

CDS Security object for the NS Security object:

```
{user:${HPSS_PRINCIPAL_FTPD}:r---c}
{user:${HPSS_PRINCIPAL_BFS}:r---c}
{user:${HPSS_PRINCIPAL_NDCG}:r---c}
{user:${HPSS_PRINCIPAL_NFSD}:r---c}
{user:${HPSS_PRINCIPAL_DMG}:rw--c}
{user:${HPSS_PRINCIPAL_SSM}:rw--c}
{user:cell_admin:rwdtc}
{any_other:---t-}
```

CDS Security object for the PVL Security object:

```
{user:${HPSS_PRINCIPAL_PVR}:rwdt-}
{user:${HPSS_PRINCIPAL_SS}:rwdt-}
{user:${HPSS_PRINCIPAL_SSM}:rwdtc}
{group:subsys/dce/cds-admin:rwdtc}
{group:subsys/dce/cds-server:rwdtc}
{any_other:---t-}
```

CDS Security object for the PVR Security object:

```
{user:${HPSS_PRINCIPAL_PVL}:rwdt-}
{user:${HPSS_PRINCIPAL_SSM}:rwdtc}
{group:subsys/dce/cds-admin:rwdtc}
{group:subsys/dce/cds-server:rwdtc}
{any_other:---t-}
```

CDS Security object (for all other servers):

```
{user:${HPSS_PRINCIPAL_SSM}:rwdtc}
{group:${HPSS_GRP_NAME_SERVER}:rwdt-}
{group:subsys/dce/cds-admin:rwdtc}
{group:subsys/dce/cds-server:rwdtc}
{any_other:---t-}
```

6.6 HPSS Storage Policy Configuration

The configuration of the HPSS Storage Policy includes creating the Migration Policies, Purge Policies, Accounting Policies, Logging Policies, Location Policy, and Remote Site Policies. These policies will subsequently be used in configuring the HPSS storage classes and the HPSS servers.

Before configuring the HPSS Storage Policy, the planning guidelines as described in Section 2.8 should be carefully considered.

6.6.1 Configure the Migration Policies

A migration policy is associated with a storage class and defines the criteria by which data is migrated from that storage class to storage classes at lower levels in the storage hierarchies. Note, however, that it is the storage hierarchy definitions, not the migration policy, which determines the number and location of the migration targets. Also note that a storage class may be used in multiple hierarchies and may have different migration targets in each.

A single basic migration policy must be assigned to any disk or tape storage class which requires migration services. This basic migration policy determines the migration parameters for this storage class across all storage subsystems. This is the simplest way to control migration on a storage class. If it is desired for migration on a storage class to behave differently within different subsystems, storage subsystem specific migration policies may be added which override the default values in the basic policy for the given storage class in the selected subsystem. The storage subsystem specific migration policies share the same migration Policy ID and Policy Name with the basic policy.

Both basic and subsystem specific migration policies are created using the Migration Policy window. The basic policy must be created before it is possible to create the subsystem specific policies. After the policies are created, they may be viewed, updated, or deleted using the same window. From the **HPSS Health and Status** window (shown in Figure 6-1 on page 252), click on the **Admin** menu, select the **Configure HPSS option**, and click on the **Migration Policies** option.

The **Migration Policy** window will be displayed as shown in Figure 6-7 on page 282. The fields in the basic policy are displayed with default values for a new policy. If the default data for the basic policy is not desired, change the necessary fields with the desired values. Click on the **Add Basic** button to write the new basic policy to the metadata file. To configure a subsystem specific policy, select an existing basic policy as described below. Once the existing basic policy is displayed, click on the **Start New** button on the **Storage Subsystem-Specific Policy** portion of the **Migration Policy** window. The fields in the new subsystem specific policy are displayed with data extracted from the basic policy. Change the desired fields in the subsystem specific policy and then use the **Add Specific** button to write the new subsystem specific policy to the metadata file.

To update an existing basic migration policy, select the **Load Existing** button on the **Basic Policy** portion of the **Migration Policy** window and select the desired policy from the popup list. The window will be refreshed with the configured basic policy data. After modifying the basic policy, click on the **Update Basic** button (which is only visible when no subsystem specific policy is selected) to write the changes to the metadata file. To update an existing subsystem specific migration policy, first select and load the existing basic policy as described above. Then use the **Load Existing** button and popup list on the **Storage Subsystem- Specific Policy** portion of the **Migration Policy** window to load the desired subsystem specific migration policy. After modifying the subsystem specific policy, click on the **Update Specific** button (which is only visible when a subsystem specific policy is selected) to write the changes to the metadata file.

To delete an existing migration policy, first delete all of the subsystem specific policies, and then delete the basic policy. Use the **Load Existing** button and popup list on the Basic Policy portion of the screen to select the overall policy to be deleted. Next, use the **Load Existing** button and popup list on the **Storage Subsystem-Specific Policy** portion of the screen to select the first of the subsystem specific policies to be deleted. Use the **Delete Specific** button (which is only visible after a subsystem specific policy has been selected) to delete each of the subsystem specific policies in turn. Finally, use the **Delete Basic** button (which is only visible when no subsystem specific policy is selected) to delete the basic migration policy.

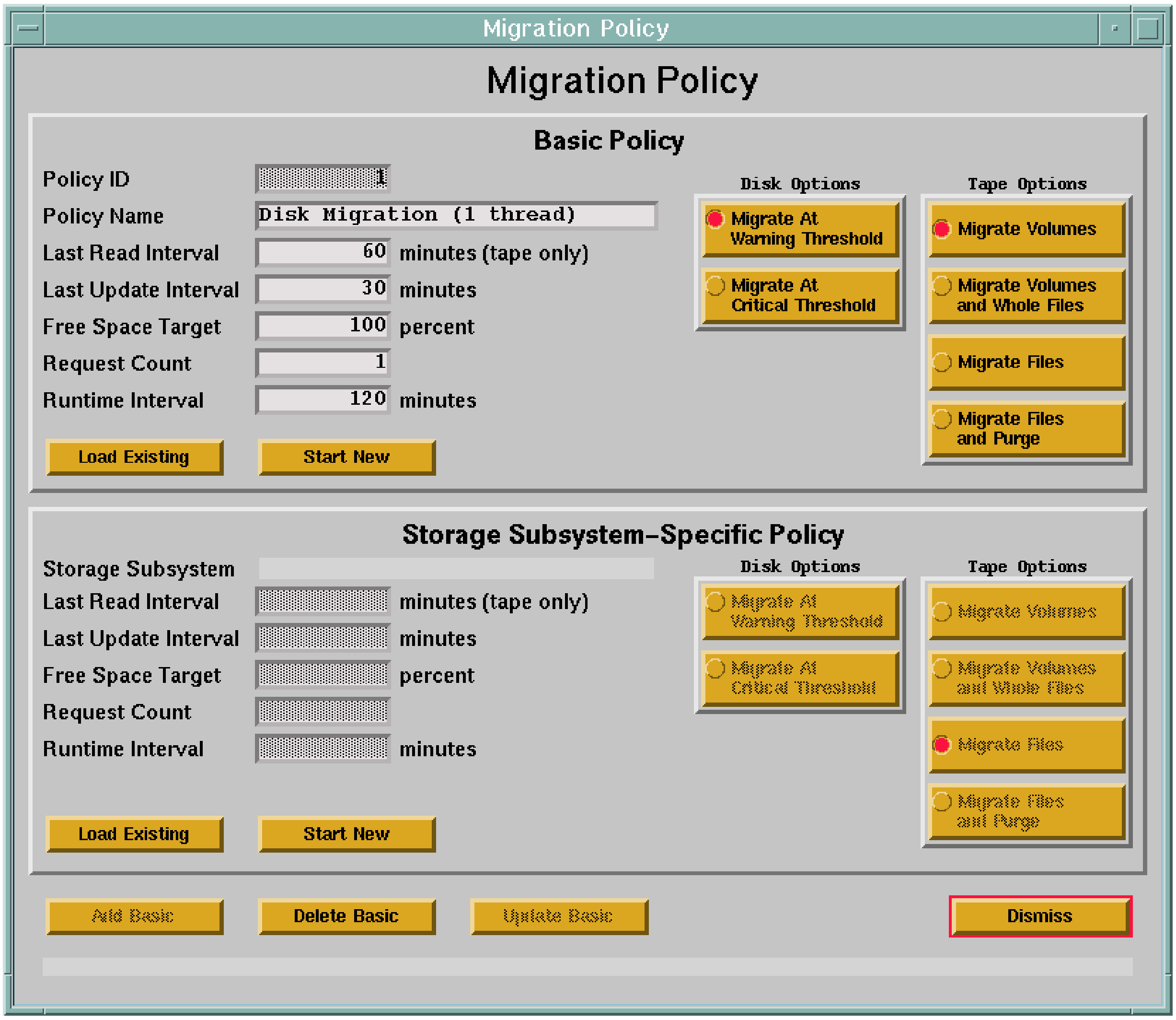
Before deleting a basic migration policy, make sure that it is not referenced in any storage class configurations. If a storage class configuration references a migration policy which does not exist, the Migration/Purge and Bitfile Servers will not start.

When a migration policy is added to or removed from a storage class configuration, the Migration/Purge Servers must be restarted in order for migration to begin or end on this storage class. It is not possible for Migration/Purge Servers to reread a migration policy for a storage class if the migration policy was not assigned to the storage class when the Migration/Purge Server was started. Likewise, if a migration policy is removed from a storage class configuration, rereading the migration policy for this storage class will not cause the Migration/Purge Server to stop migrating this storage class. It is necessary to recycle the Migration/Purge Servers when a storage subsystem specific migration policy is added or removed. It is also necessary to recycle when a storage class configuration is changed to use a different migration policy.

Before changes made to a migration policy take effect, the Migration/Purge Servers must be either restarted or instructed to reread the policy from the MPS Storage Class Information window. If the Migration/Purge Servers are restarted, the changes to the migration policy are applied to all storage classes which reference the policy. If the policy is reread, the changes are only applied to the storage class and storage subsystem for which the policy is reread. It is not necessary to recycle Bitfile Servers when the parameters in a migration policy are changed.

When a migration policy is added to or removed from a storage class configuration, the Bitfile Servers must be restarted in order for this change to take effect. If the Bitfile Servers are not restarted after a migration policy has been added to a storage class configuration, migration records will not be created for files written into the storage class. These files will never be able to migrate even after the Bitfile Servers are restarted. If the Bitfile Servers are not restarted after a migration policy has been removed from a storage class configuration, migration records will continue to be created for files written into this storage class. These migration records will become orphaned metadata since migration is no longer supported on this storage class. It is also necessary to restart the Bitfile Servers if a storage class configuration is changed to use a different migration policy. It is not necessary to restart the Bitfile Servers if a storage subsystem specific migration policy is added or removed.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.



Migration Policy

Basic Policy

Policy ID	1	
Policy Name	Disk Migration (1 thread)	
Last Read Interval	60	minutes (tape only)
Last Update Interval	30	minutes
Free Space Target	100	percent
Request Count	1	
Runtime Interval	120	minutes

Disk Options: [✓ Migrate At Warning Threshold] / Migrate At Critical Threshold

Tape Options: [✓ Migrate Volumes] / Migrate Volumes and Whole Files / Migrate Files / Migrate Files and Purge

Storage Subsystem-Specific Policy

Storage Subsystem		
Last Read Interval		minutes (tape only)
Last Update Interval		minutes
Free Space Target		percent
Request Count		
Runtime Interval		minutes

Disk Options: Migrate At Warning Threshold / Migrate At Critical Threshold

Tape Options: Migrate Volumes / Migrate Volumes and Whole Files / [✓ Migrate Files] / Migrate Files and Purge

Figure 6-7 Migration Policy Configuration Window

6.6.1.1 *Migration Policy Configuration Variables*

Table 6-4 lists the fields on the Migration Policy window and provides specific recommendations for configuring the Migration Policy for use by HPSS. Note that descriptions of fields which appear both in the **Basic Policy** and **Storage Subsystem-Specific Policy** sections of the window apply to both fields.

Table 6-4 Migration Policy Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Policy ID	A unique ID associated with the Migration Policy.	Any unique, non-zero, positive integer value.	Last configured Migration Policy ID plus 1.
Policy Name	The descriptive name of a Migration Policy.	Any character string up to 31 bytes in length.	Migration Policy ID
	Advice: A policy's descriptive name should be meaningful to local site administrators and operators.		
Last Read Interval	An integer that determines whether a file is a candidate for migration. The file must not have been read in the last <interval> minutes. This applies to tape volume migration and tape file migration with purge only.	0-1000000 (one million)	60
	Advice: Refer to Section 2.8.1.2 for more information on tape migration.		
Last Update Interval	An integer that determines whether a file is a candidate for migration. The file must not have been written to in the last <interval> minutes to be a candidate.	0-1000000 (one million)	60
Free Space Target	The desired percentage of purgeable disk or free tape space to have when migration is completed. The migration will be terminated when this goal is reached.	Any integer value between 1 and 100.	100
Request Count	The number of parallel migration threads which will be run during migration. This value applies to disk migration and tape file copy migration only.	Any positive 32-bit integer value greater than 0.	1

Table 6-4 Migration Policy Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Runtime Interval	An integer, in minutes, that dictates how often the migration process will occur within a storage class.	Any positive 32-bit integer value.	120 minutes
	Note: The value specifies the interval between the completion of one migration run and the beginning of the next.		
Migrate Volumes	Selects the tape volume migration algorithm. Applies to tape storage classes only.	ON, OFF	ON
Migrate Volumes and Whole Files	Selects the tape volume migration algorithm with the whole files option. Applies to tape storage classes only.	ON, OFF	OFF
Migrate Files	Selects the tape file migration algorithm. A duplicate copy of the tape files is made at the next lower level in the hierarchy. Applies to tape storage classes only.	ON, OFF	OFF
Migrate Files and Purge	Selects the tape file migration algorithm with the purge option. The tape files are moved to the next lower level in the hierarchy. Note that no duplicate copy is maintained. Applies to tape storage classes only.	ON, OFF	OFF
Migrate At Warning Threshold	A flag that indicates a migration run should be started immediately when the storage class warning threshold is exceeded. This option applies to disk migration only.	ON, OFF	OFF

Table 6-4 Migration Policy Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Migrate At Critical Threshold	A flag that indicates a migration run should be started immediately when the storage class critical threshold is exceeded. This option applies to disk migration only.	ON, OFF	OFF
Storage Subsystem	The descriptive name of the storage subsystem to which a subsystem-specific policy applies. This field is filled in with the selected storage subsystem name at the time a subsystem specific policy is created and may not be changed afterwards.	The descriptive name of any existing storage subsystem.	None

6.6.2 Configure the Purge Policies

A purge policy is associated with a disk storage class and defines the criteria by which data is purged from that storage class once migration has copied that data to storage classes at lower levels in the storage hierarchies.

A single basic purge policy must be assigned to any disk storage class which requires purge services. This basic purge policy determines the purge parameters for this storage class across all storage subsystems. This is the simplest way to control purge on a storage class. If it is desired for purge on a storage class to behave differently within different subsystems, storage subsystem specific purge policies may be added which override the default values in the basic policy for the given storage class in the selected subsystem. The storage subsystem specific purge policies share the same purge Policy ID and Policy Name with the basic policy.

Both basic and subsystem specific purge policies are created using the **Purge Policy** window. The basic policy must be created before it is possible to create the subsystem specific policies. After the policies are created, they may be viewed, updated, or deleted using the same window. From the **HPSS Health and Status** window (shown in Figure 6-1 on page 252), click on the **Admin** menu, select the **Configure HPSS option**, and click on the **Purge Policies** option.

The **Purge Policy** window will be displayed as shown in Figure 6-8 on page 287. The fields in the basic policy are displayed with default values for a new policy. If the default data for the basic policy is not desired, change the necessary fields with the desired values. Click on the **Add Basic** button to write the new basic policy to the metadata file. To configure a subsystem specific policy, select an existing basic policy as described below. Once the existing basic policy is displayed, click on the **Start New** button on the **Storage Subsystem-Specific Policy** portion of the **Purge Policy** window. The fields in the new subsystem specific policy are displayed with data extracted from the basic policy. Change the desired fields in the subsystem specific policy and then use the **Add Specific** button to write the new subsystem specific policy to the metadata file.

To update an existing basic purge policy, select the **Load Existing** button on the **Basic Policy** portion of the **Purge Policy** window and select the desired policy from the popup list. The window will be refreshed with the configured basic policy data. After modifying the basic policy, click on the **Update Basic** button (which is only visible when no subsystem specific policy is selected) to write the changes to the metadata file. To update an existing subsystem specific purge policy, first select and load the existing basic policy as described above. Then use the **Load Existing** button and popup list on the **Storage Subsystem- Specific Policy** portion of the **Purge Policy** window to load the desired subsystem specific purge policy. After modifying the subsystem specific policy, click on the **Update Specific** button (which is only visible when a subsystem specific policy is selected) to write the changes to the metadata file.

To delete an existing purge policy, first delete all of the subsystem specific policies, and then delete the basic policy. Use the **Load Existing** button and popup list on the **Basic Policy** portion of the screen to select the overall policy to be deleted. Next, use the **Load Existing** button and popup list on the **Storage Subsystem-Specific Policy** portion of the screen to select the first of the subsystem specific policies to be deleted. Use the **Delete Specific** button (which is only visible after a subsystem specific policy has been selected) to delete each of the subsystem specific policies in turn. Finally, use the **Delete Basic** button (which is only visible when no subsystem specific policy is selected) to delete the basic purge policy.

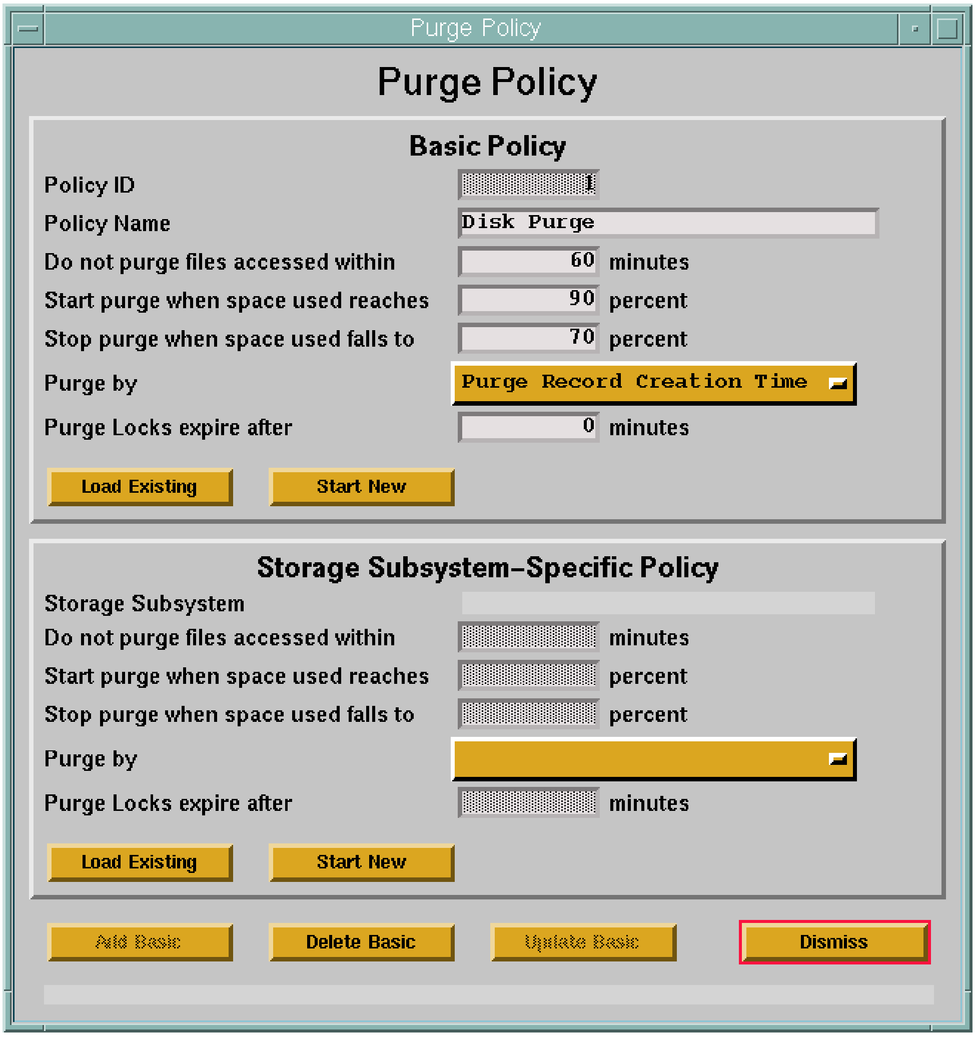
Before deleting a basic purge policy, make sure that it is not referenced in any storage class configurations. If a storage class configuration references a purge policy which does not exist, the Migration/Purge and Bitfile Servers will not start.

When a purge policy is added to or removed from a storage class configuration, the Migration/Purge Servers must be restarted in order for purge to begin or end on this storage class. It is not possible for MPS to reread a purge policy for a storage class if the purge policy was not assigned to the storage class when the MPS was started. Likewise, if a purge policy is removed from a storage class configuration, rereading the purge policy for this storage class will not cause the MPS to stop purge this storage class. It is necessary to recycle the Migration/Purge Servers when a storage subsystem specific purge policy is added or removed. It is also necessary to recycle when a storage class configuration is changed to use a different purge policy.

Before changes made to a purge policy take effect, the Migration/Purge Servers must be either restarted or instructed to reread the policy from the MPS Storage Class Information window. If the Migration/Purge Servers are restarted, the changes to the purge policy are applied to all storage classes which reference the policy. If the policy is reread, the changes are only applied to the storage class and storage subsystem for which the policy is reread. Bitfile Servers are not able to reread purge policies. If the Purge By field is changed on either a basic or subsystem specific purge policy, the relevant Bitfile Servers must be restarted.

When a purge policy is added to or removed from a storage class configuration, the Bitfile Servers must be restarted in order for this change to take effect. If the Bitfile Servers are not restarted after a purge policy has been added to a storage class configuration, purge records will not be created for files written into the storage class. These files will never be able to purge even after the Bitfile Servers are restarted. If the Bitfile Servers are not restarted after a purge policy has been removed from a storage class configuration, purge records will continue to be created for files written into this storage class. These purge records will become orphaned metadata since purge is no longer supported on this storage class. It is also necessary to restart the Bitfile Servers if a storage class configuration is changed to use a different purge policy. It is not necessary to restart the Bitfile Servers if a storage subsystem specific purge policy is added or removed.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.



Purge Policy

Basic Policy

Policy ID	1
Policy Name	Disk Purge
Do not purge files accessed within	60 minutes
Start purge when space used reaches	90 percent
Stop purge when space used falls to	70 percent
Purge by	Purge Record Creation Time
Purge Locks expire after	0 minutes

Storage Subsystem-Specific Policy

Storage Subsystem	
Do not purge files accessed within	minutes
Start purge when space used reaches	percent
Stop purge when space used falls to	percent
Purge by	
Purge Locks expire after	minutes

Figure 6-8 Purge Policy Window

6.6.2.1 *Purge Policy Configuration Variables*

Table 6-5 lists the fields on the Purge Policy window and provides specific recommendations for configuring the Purge Policy for use by HPSS. Note that descriptions of fields which appear both in the **Basic Policy** and **Storage Subsystem-Specific Policy** sections of the window apply to both fields.

Table 6-5 Purge Policy Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Policy ID	A unique ID associated with the Purge Policy.	Any unique, non-zero, positive integer value.	Last configured Purge Policy ID plus 1.
Policy Name	The descriptive name of a Purge Policy.	Any character string up to 31 bytes in length.	Purge Policy ID
	Advice: A policy's descriptive name should be meaningful to local site administrators and operators.		
Do not purge files accessed within	To be considered a candidate for purging, a file must have been migrated, and it must have remained unaccessed (for read or write) for the length of time specified by this field.	0-1000000 (one million)	60
Start purge when space used reaches	Purging will begin for a storage class when the amount of its space used exceeds this threshold. Used space includes any file in the storage class, whether it has been migrated or not.	Any integer value between 0 and 100.	90
Stop purge when space used falls to	Purging will stop for a storage class when the amount of its space used drops to this threshold. Note that the purge may stop before this point if it runs out of files which are available for purging.	Any integer number between 1 and 100.	70
Purge by	The MPS uses this time attribute of a file to determine which files are eligible to be purged.	Purge Record Creation Time, File Creation Time, or Last Data Access Time	Purge Record Creation Time
	Note: After changing the value for this field, the Bitfile Server must be restarted for the change in policy to take effect.		

Table 6-5 Purge Policy Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Purge Locks expire after	Maximum number of minutes that a file may hold a purge lock. Purge locked files are not eligible for purging.	Any integer value between 0 and 1000000 (one million). A value of 0 indicates that purge locks expire immediately.	0
Storage Subsystem	The descriptive name of the storage subsystem to which a subsystem-specific policy applies. This field is filled in with the selected storage subsystem name at the time a subsystem specific policy is created and may not be changed afterwards.	The descriptive name of any existing storage subsystem.	None

6.6.3 *Configure the Accounting Policy*

The accounting policy defines how each HPSS user account will be charged for using HPSS resources. Refer to Section 2.8.3 for more information on setting and using the Accounting Policy. An accounting policy can be created using the Accounting Policy window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Accounting Policy** option.

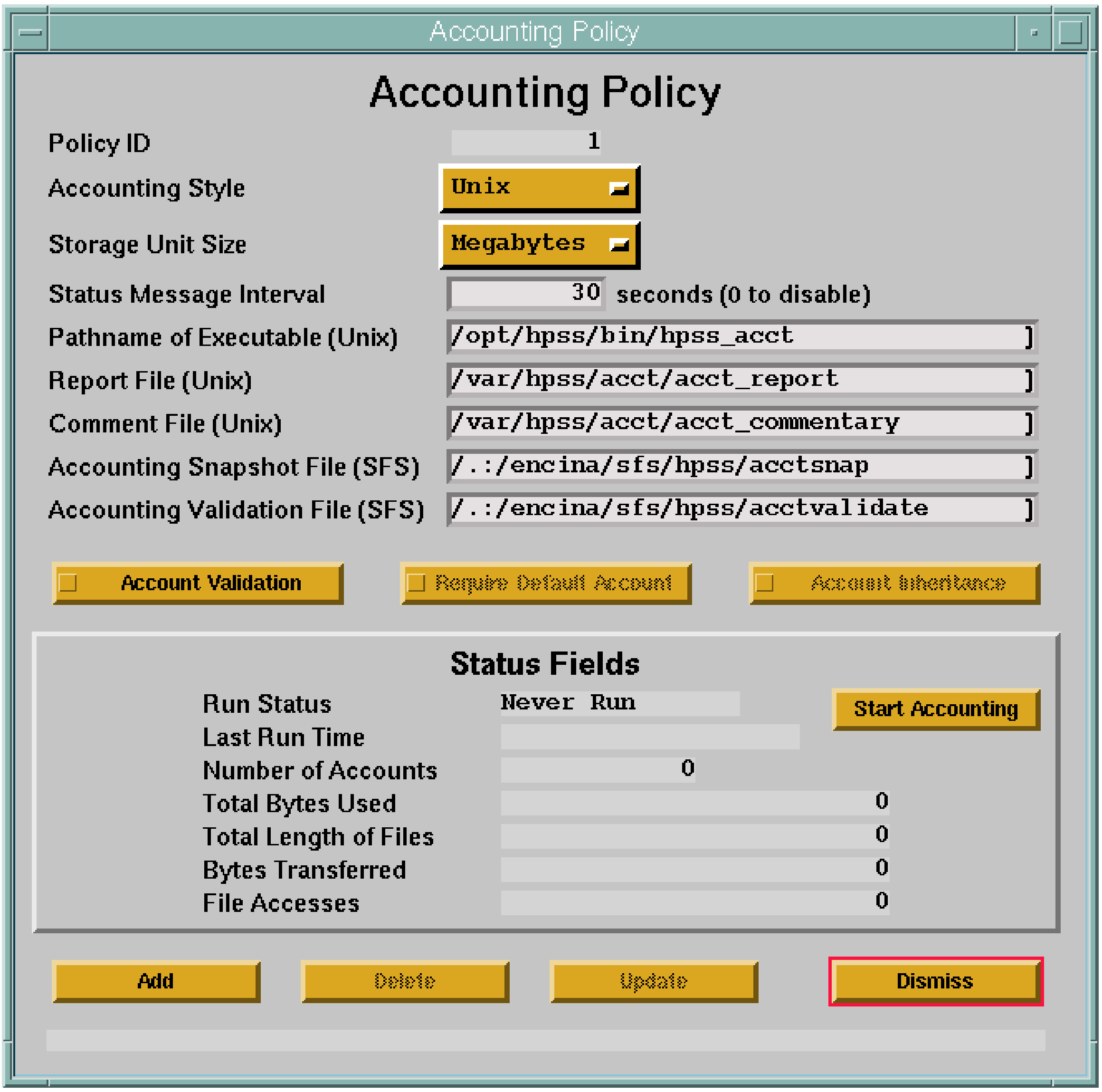
The Accounting Policy window will be displayed as shown in Figure 6-9. Since only one Accounting Policy can be configured, the fields are displayed with default values for a new policy if one does not exist. Otherwise the fields will be displayed with data from the configured policy.

To add a new policy, change the default fields with the desired values and click on the **Add** button to create the configuration entry.

To update an existing policy, modify the policy data and click on the **Update** button to write the changes to the SFS file.

To delete the existing policy, **c**lick on the **Delete** button to delete the policy.

Refer to the window’s help file for more information on the individual fields and buttons as well as the supported operations available from the window.



Accounting Policy

Accounting Policy

Policy ID	1
Accounting Style	Unix
Storage Unit Size	Megabytes
Status Message Interval	30 seconds (0 to disable)
Pathname of Executable (Unix)	/opt/hpss/bin/hpss_acct
Report File (Unix)	/var/hpss/acct/acct_report
Comment File (Unix)	/var/hpss/acct/acct_commentary
Accounting Snapshot File (SFS)	/.:/encina/sfs/hpss/acctsnap
Accounting Validation File (SFS)	/.:/encina/sfs/hpss/acctvalidate

Status Fields

Run Status	Never Run
Last Run Time	
Number of Accounts	0
Total Bytes Used	0
Total Length of Files	0
Bytes Transferred	0
File Accesses	0

Figure 6-9 Accounting Policy Window

6.6.3.1 *Accounting Policy Configuration Variables*

Table 6-6 lists the fields on the Accounting Policy window and provides specific recommendations for configuring the Accounting Policy for use by HPSS.

Table 6-6 Accounting Policy Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Policy ID	The unique ID associated with the Accounting Policy.	Always 1. This field is not changeable.	1
	Advice: This number is always 1 for this version of HPSS since only one accounting policy is currently allowed.		
Accounting Style	Style of accounting that is used by the entire HPSS system.	Site, UNIX	UNIX
	Advice: This field must be set up properly before any files are written to the system. UNIX-style accounting obtains account number from the user's UID. Site-style accounting allows the account to be set specifically. Once the accounting policy is configured, this value cannot be changed. *Note*: Please refer to Section 2.8.3: *Accounting Policy and Validation* on page 85 for more information.		
Storage Unit Size	The integral units to be displayed in the report file.	Bytes, Kilobytes, Megabytes, Gigabytes Terabytes	Bytes
Status Message Interval	The number of seconds between status messages sent to SSM by the Accounting utility during an accounting run.	Any positive 32-bit integer value.	300 seconds
	Advice: Set the interval to zero (0) if no accounting status messages are desired. Otherwise, for performance reasons, this value should be set to at least 15 seconds		
Pathname of Executable (Unix)	The UNIX path name of the accounting utility executable.	Valid UNIX path name.	/opt/hpss/bin/ hpss_acct
	Advice: SSM should have execute privileges for the pathname.		
Report File (Unix)	The UNIX pathname where the generated accounting report will be stored.	Any legal UNIX pathname	/var/hpss/acct/ acct_report
	Advice: SSM should have write access for the pathname.		
Comment File (Unix)	A UNIX pathname of an optional commentary text file.	Empty, or any legal UNIX pathname	/var/hpss/acct/ acct _commentary
	Advice: This pathname is optional. If it exists, SSM must have read access to it.		
Accounting Snapshot File (SFS)	The Encina SFS filename of the Accounting Snapshot metadata.	Any valid Encina SFS filename.	/.:/encina/sfs/ hpss/acctsnap

Table 6-6 Accounting Policy Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Accounting Validation File	The Encina SFS filename of the Accounting Validation metadata	Any valid Encina SFS filename.	/.:/encina/sfs/hpss/acctvalidate
	Advice: You need to create and populate this file with the account validation editor if Account Validation is enabled and Site-style accounting is in use (see Section 12.2.23: hpss_avaledit — Account Validation Editor on page 366 of the HPSS Management Guide).		
Account Validation	A flag that indicates whether or not authorization of user accounts should be performed	ON, OFF	OFF
	Advice: If you turn this flag ON, you must configure at least one Gatekeeper Server which will perform the account validation service.		
Require Default Account	A flag that indicates whether or not users must have a valid default account index before they are allowed to perform any operation. Only used if Account Validation has been enabled. If this flag is disabled, validation will only occur during required operations such as file creations, file ownership changes, and account change operations	ON, OFF	OFF
Account Inheritance	A flag that indicated whether or not newly created files and directories should automatically inherit the account index used by their parent directory. Only used if Account Validation has been enabled and Site-style accounting has been selected. If this flag is disabled, new files and directories have the user's current session account index applied to them.	ON, OFF	OFF
Status Fields *The fields below are statuses reported for the current or last accounting run. They are not configuration fields, and thus are not changeable.*			
Run Status	Status of the current or last accounting run	Never Run, Running, Failed, Completed, Report Generated List	Never Run
	Advice: This value is set by the Accounting Utility when the status of an accounting run changes.		

Table 6-6 Accounting Policy Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Last Run Time	The starting timestamp of the current accounting run or the completion time of the last accounting run.	A date and time value.	0
	Advice: This time is set when an accounting run begins, and it is set again when the accounting run terminates.		
Number of Accounts	Total number of accounts in the system.	A positive 32-bit integer value.	0
Total Bytes Used	Total number of bytes accounted for in the system.	A positive 64-bit integer value.	0
Total Length of Files	The total length of bitfiles, specified by the defined **Storage Unit Size**, in the HPSS system.	A positive 64-bit integer value.	0
Bytes Transferred	The total number of bytes transferred in and out of the HPSS system since the last accounting run.	A positive 64-bit integer value.	0
File Accesses	The total number of bitfiles accesses since the last accounting run.	A positive 64-bit integer value.	0

6.6.4 Configure the Logging Policies

For any HPSS server or utility which uses the HPSS logging facility, a Logging Policy can be configured to specify which log message types get sent to the HPSS logs, and which message types get sent to SSM for display. All that is required to configure a Logging Policy is a server name. For configured servers, this is the same as the Server Name field on the Basic Server Configuration window (see Section 6.5.1.1: *Basic Server Configuration Variables* on page 265). For servers and utilities which do not have configurations (such as the SSM Data Server), Logging Policies can be configured using the name by which the server identifies itself to the logging facility. These names can be obtained by examining server startup scripts or HPSS log entries.

If no Logging Policy is configured for a server, the server will use the Default Log Policy. However, if no default is defined (and the server doesn't have a log policy) all message types other than Trace will be logged.

6.6.4.1 HPSS Logging Policies Window

From the HPSS Health and Status window (shown in Figure 6-1 on page 252), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Logging Policies** option.

Multiple changes to the logging policy list can be made at one time. When all the changes are completed, clicking on the **Save Changes** button will save the modified logging policy list. As entries are added, deleted, or modified, an indicator (**NEW**, **DEL**, **MOD**) will be displayed in the Mod column. The **Cancel Changes** button can be used any time before the changes are saved to return a policy marked **MOD** back to its original settings. Deletions (**DEL**) can be "undone" by using the **Cancel Delete** button which becomes visible only after a policy has been marked for deletion.

To create a new logging policy, click on the **Start New** button. A new line will be highlighted and you can fill in the **Descriptive Name** field. **NEW** will be displayed in the Mod column after the name is entered. If a Default Logging Policy has been defined, the fields under column headings **Record Types to Log** and **SSM Types** will be automatically filled in to match the default logging policy. If there is no **Default Logging Policy** defined, all fields except **TRC** (Trace) will contain asterisks. An asterisk indicates the logging type is set to the "on" state. Clicking in a blank field turns on that logging type. Clicking on an asterisk will toggle the logging type to the "off" state which is blank. Repeat this sequence for each new policy which you want to add.

To delete a policy, click on the descriptive name to highlight the policy, then click on the **Delete** button. **DEL** will be displayed in the **Mod** column. Repeat this sequence for each policy which you want to remove. If you change your mind about a policy which has been marked **DEL**, click on that policy name again. The **Delete** button will change to **Cancel Delete**. Clicking on the **Cancel Delete** button will "undelete" the selected policy.

To update a displayed policy, click on the **Descriptive Name** field to highlight the policy. Click in the field which you want to modify to change its state to "on" or "off". **MOD** will be displayed in the **Mod** column. Repeat this sequence for each policy which you want to change.

When finished making all the changes to the list, clicking on the **Save Changes** button will update those logging policies which are marked **NEW**, **DEL**, or **MOD**. Use the **Cancel Changes** button to return a policy marked **MOD** back to its original settings.

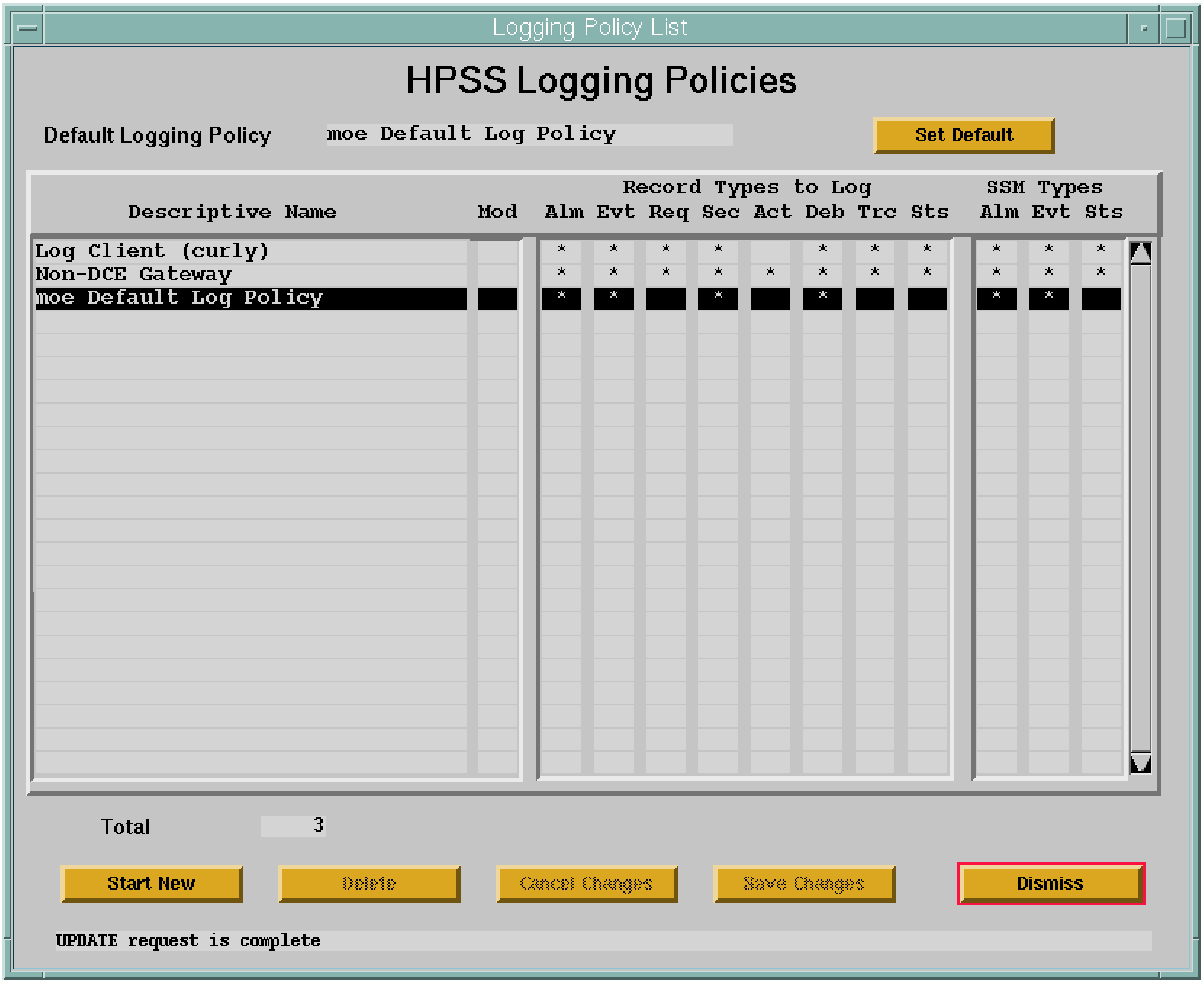
After changing any Logging Policy information, including the Default Logging Policy, the appropriate Logging Clients must be reinitialized to make the changes effective. For example, if you only modified policies for servers which execute on host A, then only the Logging Client which runs on host A needs to be reinitialized.

The HPSS Logging Policies window will be displayed as shown in Figure 6-10 on page 295. For any configured server, click on the entry for the desired server. The Logging Policy for a configured server may also be accessed from the HPSS Servers window, and from the Basic Server Configuration window for that server.

A Logging Policy can be created for any unique name which is typed into the Descriptive Name field on the Logging Policy window, regardless of whether or not the name refers to any actual server or utility.

Refer to the window’s help file for more information on the individual fields and buttons as well as the supported operations available from the window.

Once a Logging Policy is created or updated, it will not be in effect until its associated Log Client is started or reinitialized. The Log Client associated with a particular policy is the one which executes on the same host as the server to which the policy applies. The Reinitialize button on the HPSS Servers window(Figure 6-5 on page 264) can be used to reinitialize a running Log Client.



Default Logging Policy: moe Default Log Policy

		Record Types to Log								SSM Types		
Descriptive Name	Mod	Alm	Evt	Req	Sec	Act	Deb	Trc	Sts	Alm	Evt	Sts
Log Client (curly)		*	*	*	*		*	*	*	*	*	*
Non-DCE Gateway		*	*	*	*	*	*	*	*	*	*	*
moe Default Log Policy		*	*		*		*			*	*	

Total 3

```
UPDATE request is complete
```

Figure 6-10 HPSS Logging Policies Window

6.6.4.2 *HPSS Logging Policies List Variables*

Table 6-7 Logging Policies List Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Default Logging Policy	The descriptive name of the default logging policy. This policy will apply to all servers which do not have their own policy defined.	Blank or the Descriptive Name of one of the logging policies in the list. Simply highlight the desired policy in the list and click “Set Default” to make it the Default Logging Policy.	Blank

Display Field Name	Description	Acceptable Values	Default Value
Descriptive Name	The descriptive name of the HPSS server to which the Logging Policy will apply.	This field is filled in with the descriptive names of existing policies. If Start New is selected, a blank entry will be added, and any unique Descriptive Name may be entered.	Names of existing entries
Record Types to Log	Record types that are to be logged for the specified server.	Any combination of the following: Alarm, Event, Request, Security, Accounting, Debug, Trace, Status.	Values from the Default Logging Policy entry. If no Default Logging Policy name, the default is Alarm, Event, Request, Security, Accounting, Debug, Status
	Advice: It is recommended that at least the Alarm, Event, Security and Status record types be selected for all servers while they are running normally. Additional record types for debugging purposes should be selected for servers experiencing problems. If system performance is being degraded by excessive logging, first filter out Trace, Debug and Request record types from all servers. Other record types can be filtered out as necessary. However, the HPSS administrator may lose useful debugging information when problems occurred.		

Display Field Name	Description	Acceptable Values	Default Value
SSM Types	Record types that are to be sent to SSM for display.	Any combination of the following: Alarm, Event, Status.	Values from the Default Logging Policy entry. If no Default Logging Policy name, the default is Alarm, Event, Status
	Advice: If SSM or system performance is degraded because excessive messages are being sent to SSM by a particular server(s), set the Logging Policy to filter out Alarm and Event record types. However, the HPSS administrator will lose visibility into any subsequent Alarm or Event messages from that server(s). *Since anything can be typed into the Descriptive Name field, it is easy to make typos if you use the Start New button to create logging policies for preconfigured servers. For instance, if you enter "bfs" (lower case) as the logging policy descriptive name for a server whose Server Name is "BFS" (upper case), the logging policy "bfs" will not define the logging policy for server "BFS". To avoid this mistake, it is recommended that logging policies for configured servers be created by clicking the Log Policy button on the HPSS Servers window which automatically fills in the name of the selected server as the server's logging policy descriptive name.*		

6.6.4.3 Logging Policy Window

The Logging Policy window will be displayed as shown in Figure 6-11 on page 299. The Logging Policy for a configured server may be accessed from the HPSS Servers window, and from the Basic Server Configuration window for that server. With either of the last two methods, the Logging Policy window opens with the server name already filled in.

If the Logging Policy for the selected server does not exist, the fields will be displayed with default values for a new policy. Otherwise, the configured policy will be displayed.

To add a new policy, change the default fields as desired and click on the **Add** button to create the policy.

To update an existing policy, modify the desired fields and click on the **Update** button to write the changes to the SFS file.

To delete an existing policy, click on the **Delete** button to delete the policy.

The following log message types can be specified for logging and/or SSM display:

- Alarm—defines a high-level error condition of interest to the administrator. Typically, the policy would be to send alarms to both the log and to SSM for displaying in the HPSS Alarms and Events window (see Figure 1-5 on page 38 of the *HPSS Management Guide*).

- Event—defines an informational message (e.g., subsystem initializing, subsystem terminating). Typically, the policy would be to send events to both the log and to SSM for displaying in the HPSS Alarms and Events window (Figure 1-5 on page 38 of the *HPSS Management Guide*).

- Status - defines a status message to be output in a pop-up window or to the log. Typically, the policy would be to *not* send these messages to the log or to the screen. Currently, this message type is only used by the Accounting.

The following log message types can be specified for logging:

- Debug—defines a lower-level error message. For troubleshooting, these messages are important record types because they provide more detailed information about the root cause of an error. Typically, the policy would be to *not* send these messages to the log.

- Request—used to log the fact that a particular client or server request is being processed. Typically, the policy would be to send these messages to the log.

- Security—used to log security related events (e.g., authorization failures). Typically, the policy would be to *not* send these messages to the log.

- Accounting—used to log accounting information. This record type is not currently used.

- Trace—used to trace any type of entry / exit processing flows. Typically, the policy would be to *not* send these messages to the log.

The Logging Policy can be created using the HPSS Logging Policies window. After the Logging Policy is created, it can be viewed, updated, or deleted through the same window.

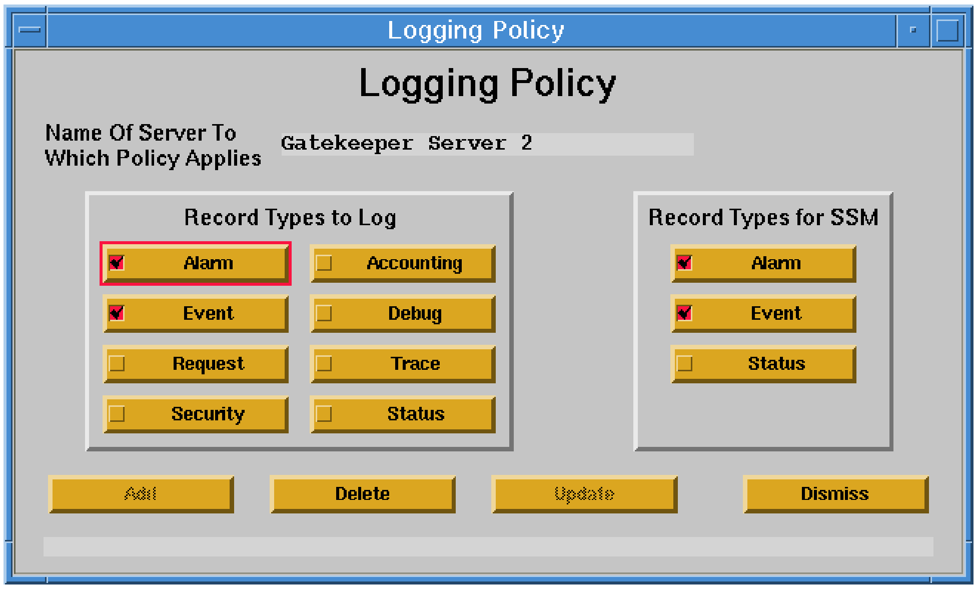
From the HPSS Servers window (shown in Figure 6-5), select a server and click on the **Log Policy...** button.

The Logging Policy window will be displayed as shown in Figure 6-11. For configured server, click on the **Name of Server to Which Policy Applies** popup button and select the desired server. The Logging Policy for a configured server may also be accessed from the HPSS Servers window, and from the Basic Server Configuration window for that server. With either of the last two methods, the Logging Policy window opens with the server name already filled in.

For non-configured servers or utilities, simply type a server name into the **Name of Server to Which Policy Applies** field, and press **Enter**.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

Once a Logging Policy is created or updated, it will not be in effect until its associated Log Client is started or reinitialized. The Log Client associated with a particular policy is the one which executes on the same host as the server to which the policy applies. The Reinitialize button on the HPSS Servers window(Figure 1-1 on page 20 of the *HPSS Management Guide*) can be used to reinitialize a running Log Client.



Name Of Server To Which Policy Applies: Gatekeeper Server 2

Record Types to Log: Alarm ✓, Event ✓

Record Types for SSM: Alarm ✓, Event ✓

Figure 6-11 Logging Policy Window

6.6.4.4 *Logging Policy Configuration Variables*

Table 6-8 lists the fields on the Logging Policy window and provides Logging Policy configuration information.

Table 6-8 Logging Policy Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Name of Server to Which Policy Applies	The descriptive name of the HPSS server to which the Logging Policy will apply.	This field is filled in with the descriptive name of the server that the user had selected and cannot be changed from this display.	The name of the selected server.

Table 6-8 Logging Policy Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Record Types to Log	Record types that are to be logged for the specified server.	Any combination of the following: Alarm, Event, Request, Security, Accounting, Debug, Trace, Status.	Alarm, Event, Request, Security, Accounting, Debug, Status
	Advice: It is recommended that at least the Alarm, Event, Security and Status record types be selected for all servers while they are running normally. Additional record types for debugging purposes should be selected for servers experiencing problems. If system performance is being degraded by excessive logging, first filter out Trace, Debug and Request record types from all servers. Other record types can be filtered out as necessary. However, the HPSS administrator may lose useful debugging information when problems occurred.		
Record Types for SSM	Record types that are to be sent to SSM for display.	Any combination of the following: Alarm, Event, Status.	Alarm, Event, Status
	Advice: If SSM or system performance is degraded because excessive messages are being sent to SSM by a particular server(s), set the Logging Policy to filter out Alarm and Event record types. However, the HPSS administrator will lose visibility into any subsequent Alarm or Event messages from that server(s).		

6.6.5 Configure the Location Policy

All of the Location Servers at the local HPSS site share the same Location Policy. The Location Policy is used by the Location Servers to determine how and how often information should be updated. In general, most of the default values for the policy can be used as defined.

The Location Policy can be created using the Location Policy window. After the Location Policy is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Location Policy** option. The Location Policy window will be displayed as shown in Figure 6-12. If the Location Policy does not exist, the fields will be displayed with default values for a new policy. Otherwise, the configured policy will be displayed.

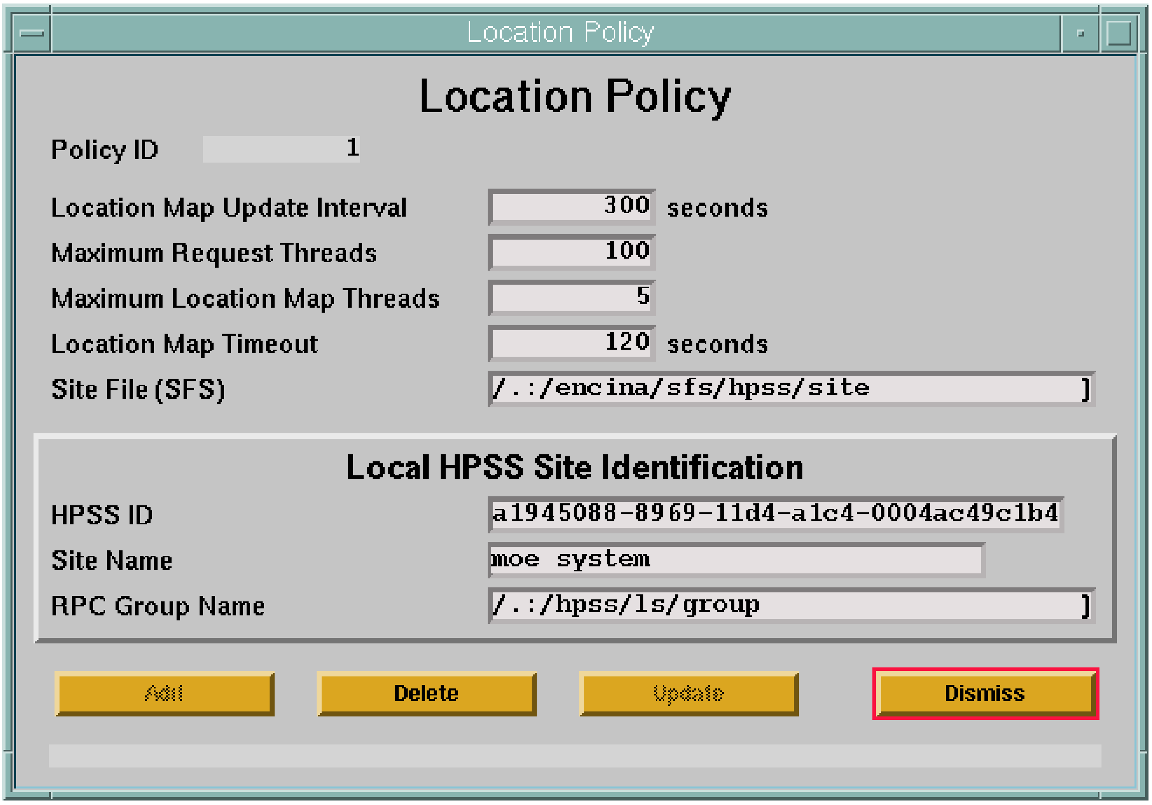
To add a new policy, change the default fields as desired and click on the **Add** button to create the policy.

To update an existing policy, modify the desired fields and click on the **Update** button to write the changes to the SFS file.

To delete an existing policy, click on the **Delete** button to delete the policy.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

Once a Location Policy is created or updated, it will not be in effect until all local Location Servers are started or reinitialized. The Reinitialize button on the HPSS Servers window(Figure 1-1 on page 20 of the *HPSS Management Guide*) can be used to reinitialize a running Location Server.



Location Policy

Policy ID 1

Location Map Update Interval 300 seconds

Maximum Request Threads 100

Maximum Location Map Threads 5

Location Map Timeout 120 seconds

Site File (SFS) /.:/encina/sfs/hpss/site

Local HPSS Site Identification

HPSS ID a1945088-8969-11d4-a1c4-0004ac49c1b4

Site Name moe system

RPC Group Name /.:/hpss/ls/group

Figure 6-12 Location Policy Window

6.6.5.1 *Location Policy Configuration Variables*

Table 6-9 lists the fields on the Location Policy window and provides Location Policy configuration information.

Table 6-9 Location Policy Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Policy ID	The unique ID associated with a Location Policy.	This field is always one (1) and may not be changed.	1

Table 6-9 Location Policy Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Location Map Update Interval	Interval in seconds that the LS rereads general server configuration metadata.	Any positive integer value.	300
	Advice: If this value is set too low, a load will be put on SFS while reading configuration metadata and the LS will be unable to contact all remote LSs within the timeout period. If set too high, new servers will not be registered in a timely manner. Set this value higher if timeouts are occurring during remote LS communications.		
Maximum Request Threads	The maximum number of concurrent client requests allowed.	Any positive integer greater than 2 and less than or equal to 400	100
	Advice: If the LS is reporting heavy loads, increase this number. If this number is above 300, consider replicating the LS on a different machine. Note if this value is changed, the general configuration thread value should be adjusted as well.		
Maximum Location Map Threads	The maximum number of threads allocated to contact remote LSs concurrently.	Any positive integer value.	5
	Advice: This value does not need to be changed unless there are defined remote sites and the system is experiencing timeout problems contacting the remote LS.		
Location Map Timeout	Maximum amount of time in seconds to wait for a single remote LS to return location map information.	Any positive integer value.	120
	Advice: This value should only be changed if the system is experiencing very long delays while contacting remote Location Servers.		
Site File (SFS)	The Encina SFS filename of the Remote HPSS Site metadata.	Any valid Encina SFS filename.	/.:/encina/sfs/hpss/site
HPSS ID	The unique identifier (UUID) for this HPSS installation.	Any valid UUID value.	A random UUID is generated.
	Advice: If you change this value, make sure it is unique. This value is used by remote HPSS sites to connect to your HPSS site.		
Site Name	Descriptive name of the local HPSS installation	Any string value	None (““)
	Advice: Pick a name to uniquely describe the HPSS system. This name is needed for Remote Site Policy configuration to support federated name space.		

Table 6-9 Location Policy Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
RPC Group Name	The CDS pathname where the DCE RPC group containing local LS path information should be stored.	Any valid CDS pathname.	/.:/hpss/ls/ group
	Advice: All clients will need to know this group name since it is used by them when initializing to contact the Location Server. If the default is not used, ensure that associated environment variable for this field is changed accordingly for all HPSS interfaces.		

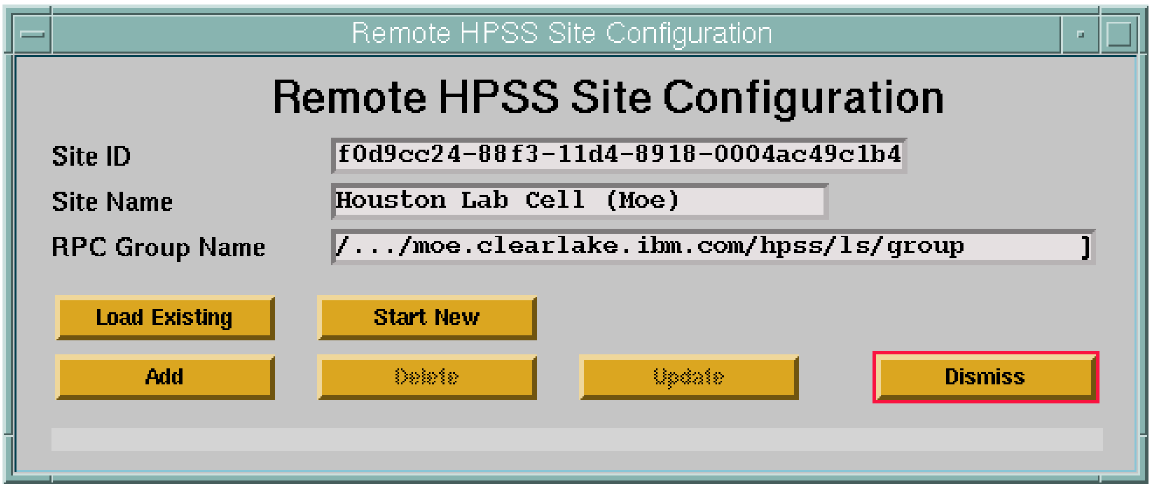
6.6.6 Configure the Remote Site Policy

When connecting multiple HPSS systems together, the Location Servers at the local site need to be told about their counterparts at the remote sites. Each record in the Remote Site file tells the local Location Servers how to contact the Location Servers at that particular remote site. All of the fields can be obtained by looking at the Local HPSS Site Identification information on the remote HPSS system's Location Policy screen.

Remote Site Policy configuration is required to support a federated name space Configuration (Section 1.2.8: *Federated Name Space* on page 21).

Remote site information can be created and maintained using the **Remote Sites** window.

From the **HPSS Health and Status** window (Figure 6-1 on page 252), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Remote Sites** option. The Remote Sites window will be displayed as shown in Figure 6-13. If a single Remote Site already exists, it will be displayed in the window. Otherwise, the fields will be blank.


Remote HPSS Site Configuration

Site ID: f0d9cc24-88f3-11d4-8918-0004ac49c1b4
Site Name: Houston Lab Cell (Moe)
RPC Group Name: /.../moe.clearlake.ibm.com/hpss/ls/group

Figure 6-13 Remote HPSS Site Configuration Window

To add a new Remote Site, enter the information about the remote HPSS system and click on the **Add** button.

To update an existing Remote Site, modify the desired fields and click on the **Update** button to write the changes to the SFS file.

To delete an existing Remote Site, click on the **Delete** button to delete the policy.

To load an existing policy click on the **Load Existing** button.

To clear out the fields on the screen in order to add a new Remote Site, click on the **Start New** button.

Refer to the window's help file for more information on the individual fields and the buttons as well as the supported operations available from the window.

Once a Remote Site is created or updated, it may take several minutes for the Location Server to notice the change and contact the site. To speed up this process, reinitialize or shutdown and restart the Location Server.

Table 6-10 Remote HPSS Site Configuration Fields

Field	Description	Acceptable Values	Default Value
Site ID	The unique DCE UUID which identifies the remote HPSS site.	You must obtain this information from the remote site’s administrator.	None

Table 6-10 Remote HPSS Site Configuration Fields

Field	Description	Acceptable Values	Default Value
Site Name	The descriptive text identifier for this site.	Any text string. This name should be unique among all site records.	None
RPC Group Name	The name of the DCE rpcgroup of the Location Servers at the remote site.	You must obtain this information from the remote site's administrator.	None

6.7 HPSS Storage Characteristics Configuration

Before an HPSS system can be used, storage classes, storage hierarchies, and classes of service must be created. Before configuring them, the planning guidelines described in Section 2.9 should be carefully considered.

6.7.1 Configure the Storage Classes

Storage class information must be created for each storage class that is to be supported by the HPSS system. A storage class can be created using the HPSS Storage Classes window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Storage Classes** option, which will open the HPSS Storage Classes window. Next click on the **New Storage Class...** button for a new window. The **HPSS Storage Class** window will appear. To create a new storage class, click on the **New Storage Class** button.

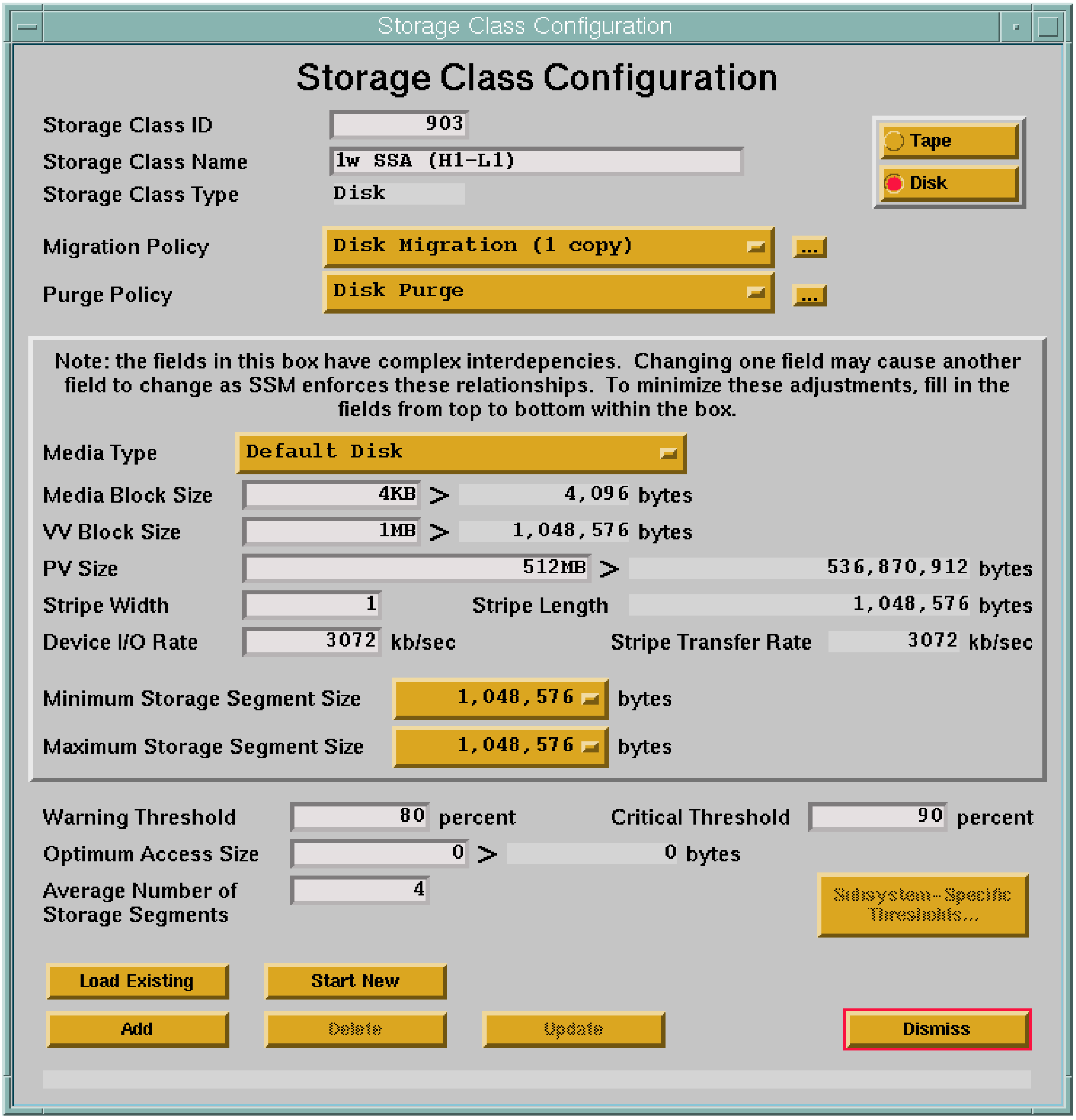
The **Storage Class Configuration** window will be displayed as shown in Figure 6-14. The fields are displayed with default values for a new disk storage class. If a tape storage class is desired, click on the **Tape** button to display the default tape storage class (Figure 6-15) before modifying any other fields. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing storage class, select the **Load Existing** button on the Storage Class Configuration window and select the desired storage class from the popup list. The window will be refreshed with the data from the selected storage class. After modifying the data, click on the **Update** button to write the changes to the SFS file. Refer to Section 3.11.1: *Changing Storage Class Definition* (page 79) in the *HPSS Management Guide* for more guidelines on changing a storage class configuration.

To delete an existing storage class, select the **Load Existing** button on the Storage Class Configuration window and select the desired storage class from the popup list. The window will be refreshed with the configured data. Click on the **Delete** button to delete the storage class. Refer to Section 3.12.1: *Deleting Storage Class Definition* (page 81) in the *HPSS Management Guide* for more guidelines on deleting a storage class configuration.

Before deleting a storage class configuration, be sure that all of the storage subsystem specific warning and critical thresholds are set to default. If this is not done, one or more threshold records will remain in metadata and will become orphaned when the storage class configuration is deleted.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.



Storage Class Configuration

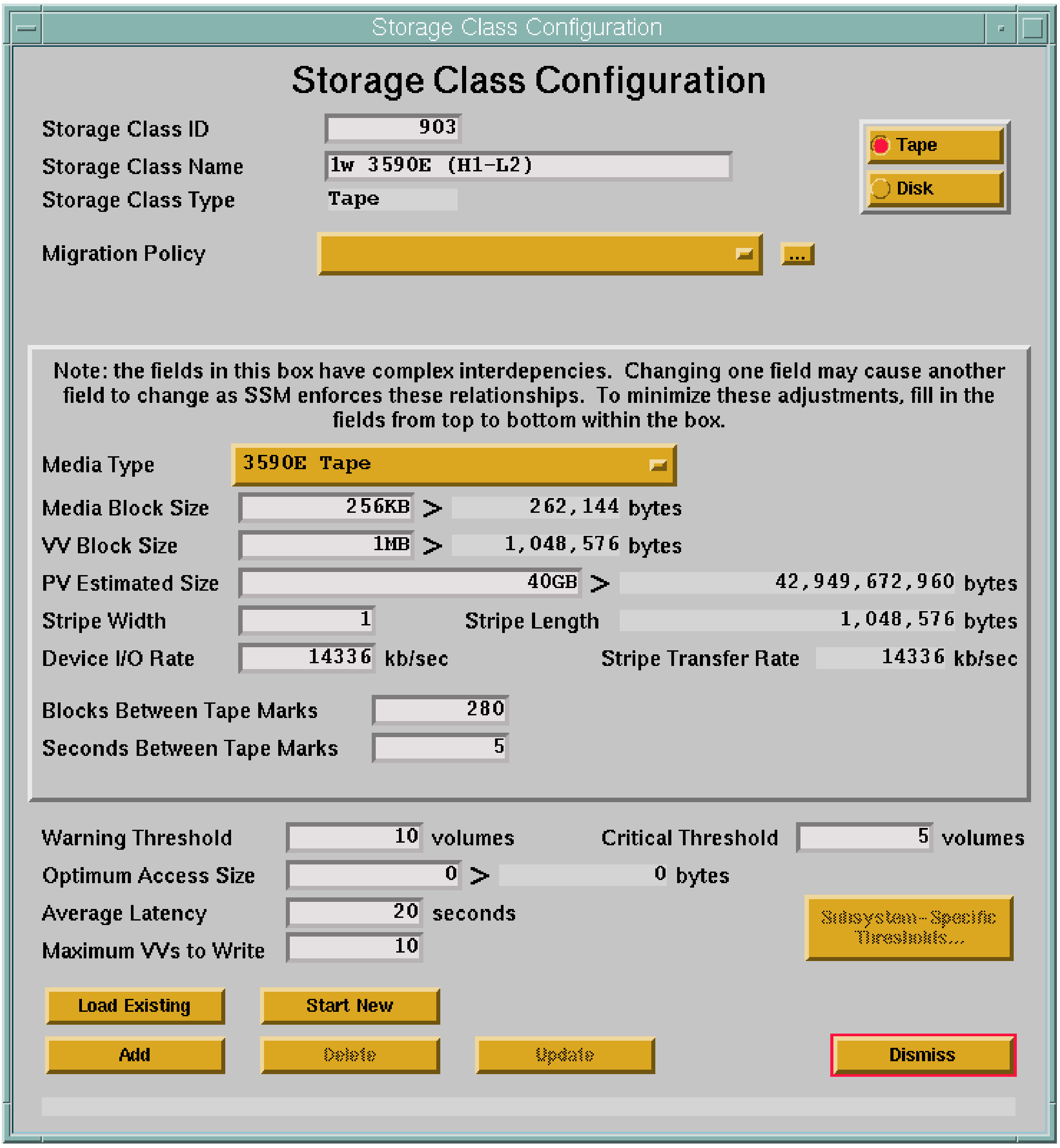
Storage Class ID: 903
Storage Class Name: 1w SSA (H1-L1)
Storage Class Type: Disk
Tape / [✓ Disk]
Migration Policy: Disk Migration (1 copy)
Purge Policy: Disk Purge

Note: the fields in this box have complex interdepencies. Changing one field may cause another field to change as SSM enforces these relationships. To minimize these adjustments, fill in the the fields from top to bottom within the box.

Media Type: Default Disk
Media Block Size: 4KB > 4,096 bytes
VV Block Size: 1MB > 1,048,576 bytes
PV Size: 512MB > 536,870,912 bytes
Stripe Width: 1 — Stripe Length: 1,048,576 bytes
Device I/O Rate: 3072 kb/sec — Stripe Transfer Rate: 3072 kb/sec
Minimum Storage Segment Size: 1,048,576 bytes
Maximum Storage Segment Size: 1,048,576 bytes

Warning Threshold: 80 percent — Critical Threshold: 90 percent
Optimum Access Size: 0 > 0 bytes
Average Number of Storage Segments: 4

Figure 6-14 Disk Storage Class Configuration Window



Storage Class Configuration

Storage Class Configuration

Storage Class ID	903
Storage Class Name	1w 3590E (H1-L2)
Storage Class Type	Tape
Migration Policy	

(● Tape) ○ Disk

Note: the fields in this box have complex interdepencies. Changing one field may cause another field to change as SSM enforces these relationships. To minimize these adjustments, fill in the fields from top to bottom within the box.

Media Type	3590E Tape			
Media Block Size	256KB >	262,144 bytes		
VV Block Size	1MB >	1,048,576 bytes		
PV Estimated Size	40GB >	42,949,672,960 bytes		
Stripe Width	1		Stripe Length	1,048,576 bytes
Device I/O Rate	14336 kb/sec		Stripe Transfer Rate	14336 kb/sec
Blocks Between Tape Marks	280			
Seconds Between Tape Marks	5			

Warning Threshold	10 volumes	Critical Threshold	5 volumes
Optimum Access Size	0 > 0 bytes		
Average Latency	20 seconds		
Maximum VVs to Write	10		

Figure 6-15 Tape Storage Class Configuration Window

6.7.1.1 *Storage Class Configuration Variables*

Table 6-11 lists the fields on the Storage Class Configuration window and provides specific recommendations for configuring the storage class for use by HPSS. SSM enforces certain relationships between the SC fields and will not allow fields to be set to inappropriate values.

Table 6-11 Storage Class Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Storage Class ID	A unique numeric ID associated with this storage class.	Any non-zero, positive 32-bit integer value.	Last configured ID plus 1.
Storage Class Name	A text string used to describe this storage class.	Any character string up to 31 bytes in length.	Storage Class ID
	Advice: Choose a Storage Class Name that describes the function of the storage class. Good examples are 4-Way striped 3490 or Non-Striped SCSI Disk.		
Storage Class Type	An indicator of whether the media to be associated with this storage class are tapes or disks.	This field is set with the Tape/Disk toggle buttons and cannot otherwise be modified.	Tape or Disk based on the toggle option selected.
Migration Policy	The name of the migration policy associated with this storage class, if data is to be migrated from this storage class.	Any configured migration policy name from the pop-up list. May be left blank if migration is not desired for this storage class.	Blank.
	Advice: Do not configure a migration policy for a storage class at the lowest level in a hierarchy. Adding a migration policy after files are created in a storage class can result in those files never being migrated.		
Purge Policy	The name of the purge policy associated with this storage class, if data is to be purged from this storage class. This field is only applicable to disk storage classes.	Any configured purge policy name from the pop-up list. May be left blank if purge is not desired for this storage class.	Blank.
	Advice: Do not configure a purge policy for a tape storage class or a storage class which does not support migration.		
Media Type	The type of media comprising this storage class.	Any valid media type from the pop-up list.	Default Tape or Default Disk based on Storage Class Type.
Media Block Size	The size of a data block on the physical volume in bytes.	Any positive 32-bit integer value.	Based on selected Media Type.
	Advice: The block size should be set to a value appropriate for the volume type. See Section 2.9.1.1 for more details.		

Table 6-11 Storage Class Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
VV Block Size	The size of the logical data block on the new virtual volume.	A 32-bit integer value.	First multiple of Media Block Size which equals or exceeds 1MB.
	*Advice: The VV Block Size chosen will determine the performance characteristics of the virtual volume. This value must meet the following constraining requirements: (1) it must be an integer multiple of the Media Block Size. (2) For tape, it must be an even divisor of the virtual volume section size (Media Block Size * Blocks Between Tape Marks). For example, if the Media Block Size is 64 KB, and the Blocks Between Tape Marks is 512, the virtual volume section size is 32 MB. The VV Block Size could be 64 K, 128 K, 256 K, or larger, but not 192 K. See Sections 2.9.1.2 and 2.9.1.3 for more details on selecting a good value for VV Block Size.*		
PV Estimated Size	The estimated amount of data that can be written on the tape physical volume.	Any positive 64-bit integer value.	Based on selected Media Type
	Advice: See Section 2.9.1.10 for considerations on selecting a good value for PV Estimated Size.		
PV Size	The size of a disk PV in the storage class, in bytes.	Any positive non-zero 64-bit value. Must be a multiple of the VV Block Size and the Media Block Size.	See section 2.9.1.10 for considerations on selecting disk PV Size values.
Stripe Width	The number of physical volumes in a virtual volume.	For tape: 1 - 16 For disk: 1 - 256	1.
Stripe Length	The number of bytes in a stripe.	This value is calculated by SSM. It is the product of the specified VV Block Size and Stripe Width fields.	Same as VV Block Size.
Device I/O Rate	The approximate data transfer speed, in kilobytes per second, which can be achieved by devices corresponding to “Media Type”. For Tape, this field is used in calculating a reasonable setting for “Blocks Between Tape Marks” (see below). For Disk, this field is mostly informational.	Any positive 32-bit integer value The value should, however, be as close as possible to the actual I/O speed of the device.	Based on selected Media Type.

Table 6-11 Storage Class Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Stripe Transfer Rate	The approximate data transfer rate for the entire stripe.	This value is calculated by SSM. It is the product of the Device I/O Rate and Stripe Width fields.	Same as Device I/O Rate.
Blocks Between Tape Marks	The maximum number of data blocks that can be written on a tape between consecutive tape marks. This field is applicable to tape storage classes only.	Any positive 32-bit integer value.	Based on selected Media Type.
	Advice: The number of blocks between the physical volume tape marks should be chosen to use the media efficiently while allowing for quick positioning to a tape mark. See Section 2.9.1.5 for more details.		
Seconds Between Tape Marks	The number of seconds between tape mark writes during a tape write operation derived from Blocks Between Tape Marks and Device I/O Rate. This field is applicable to tape storage classes only.	Any positive 32-bit integer value.	Based on selected Media Type.
	Advice: Enter a value into the Blocks Between Tape Marks field, and use Seconds Between Tape Marks value for feedback on the resulting time interval; or enter a value into the Seconds Between Tape Marks and let SSM calculate the Blocks Between Tape Marks value. Changing either field automatically modifies the other one. As a rule of thumb, it is best to choose "Blocks Between Tape Marks" such that 5 to 30 seconds elapse.		
Minimum Storage Segment Size	The smallest allocatable unit of disk space. This field is applicable to disk storage classes only.	Click on the option menu button to pop up a list of acceptable sizes.	Cluster Length.
	Advice: Care should be taken in selecting the Storage Segment Size. Consult Section 2.9.1.6 for considerations related to this selection.		
Maximum Storage Segment Size	When this field is larger than the Storage Segment Size, the Bitfile Server is allowed to use multiple storage segment sizes for disks in this storage class, up to this maximum.	Click on the option menu button to pop up a list of acceptable sizes.	Same as Storage Segment Size.
	Advice: If a value larger than 10 percent of PV Size is specified, the user will be warned of non-optimal use of disk space.		

Table 6-11 Storage Class Configuration Variables (Continued)

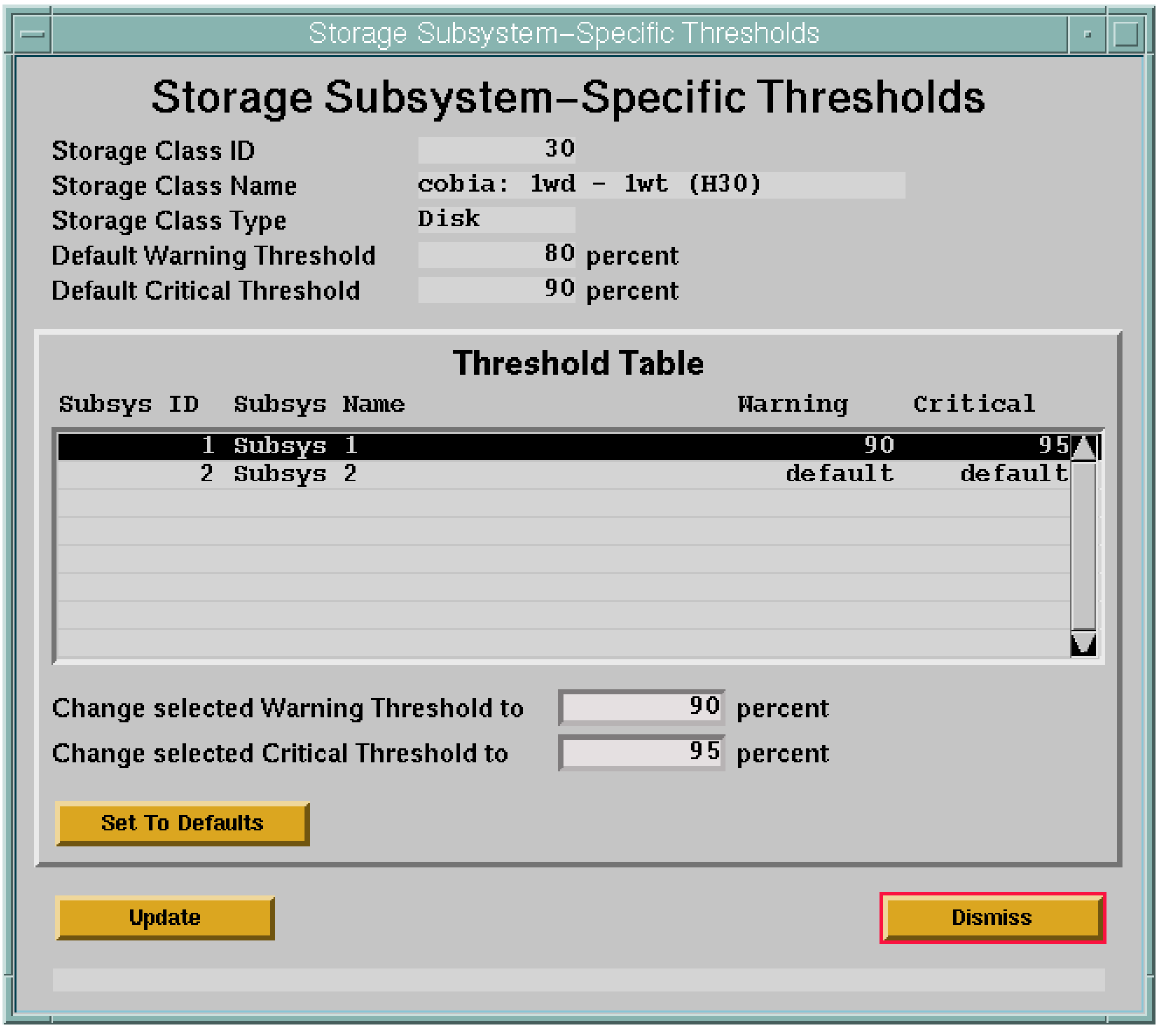
Display Field Name	Description	Acceptable Values	Default Value
Warning Threshold	Low threshold for the amount of space used in this storage class. For disk this is the percentage of total space used. For tape this is the number of free VVs remaining. Alarms will be sent to SSM periodically when this threshold is exceeded, and the **Space Thresholds** field on the **HPSS Health and Status** window will be changed to **Warning**. Note that this field may be overridden for specific storage subsystems.	For disk, any integer percentage between 1 and 100 (inclusive) which is less than or equal to the **Critical Threshold**. For tape, any non-negative integer VV count which is greater than or equal to the **Critical Threshold**. Note: The threshold for a disk class may be disabled by setting this value to 100 percent. The threshold for a tape class may be disabled by setting this value to zero.	Disk: 80, Tape: 10
Critical Threshold	High threshold for the amount of space used in this storage class. For disk this is the percentage of total space used. For tape this is the number of free VVs remaining. Alarms will be sent to SSM periodically when this threshold is exceeded, and the **Space Thresholds** field on the **HPSS Health and Status** window will be changed to **Critical**. Note that this field may be overridden for specific storage subsystems.	For disk, any integer percentage between 1 and 100 (inclusive) which is greater than or equal to the **Warning Threshold**. For tape, any non-negative integer VV count which is less than or equal to the **Warning Threshold**. Note: The threshold for a disk class may be disabled by setting this value to 100 percent. The threshold for a tape class may be disabled by setting this value to zero.	Disk: 90, Tape: 5
Optimum Access Size	The optimal transmission size to be used for a transfer request using this storage class. Not currently used.	Any positive 32-bit integer value.	0.

Table 6-11 Storage Class Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Average Latency	The average time (in seconds) that elapses when a data transfer request is scheduled and the time the data transfer begins. This field is only applicable to tape storage classes.	Any positive 32-bit integer value.	Based on selected Media Type.
	Advice: For additional information relating to the selection of storage system characteristics fields, refer to Section 2.9.3.5.		
Maximum VVs to Write	The number of tape virtual volumes in the storage class that a Tape Storage Server will use for concurrent writes. This field is only applicable to tape storage classes.	Any positive 32-bit integer value.	10.
	Advice: Small values in this field restrict files being written in the storage class to a small number of tapes, reducing the number of tape mounts. The number of tape drives used to write files in the storage class will be limited to approximately the value of this field times the stripe width. Read operations are not limited by this value.		
Average Number of Storage Segments	If the Maximum Storage Segment Size field is greater than the Storage Segment Size field, the Bitfile Server attempts to select a storage segment size for a bitfile such that the bitfile can be stored in this number of storage segments. This field is only applicable to disk storage classes.	Any positive 32-bit integer value limited by the maximum number of VVs created for the storage class.	4.
	Advice: The number of Storage Segments of a particular file that can be stored simultaneously on tape is unlimited; however, no file can have more than 10,000 disk storage segments at any one time. Also, increasing the number of storage segments will increase the amount of metadata activity required to process file reads and writes. So it is a good idea to keep this number low.		

6.7.1.2 Storage Subsystem Specific Thresholds

If different Warning and Critical Thresholds are desired for a storage class in different storage subsystems, it is possible to override the default values. By default, the Warning and Critical Thresholds specified on the **Storage Class Configuration** window are applied to that storage class across all subsystems. This is the simplest way to configure storage class thresholds. If it is desired to override these default values for one or more subsystems, use the **Subsystem-Specific Thresholds** button on the **Storage Class Configuration** window to bring up the **Storage Subsystem-Specific Thresholds** window, shown in Figure 6-16.


Storage Subsystem–Specific Thresholds

Storage Subsystem–Specific Thresholds

Storage Class ID	30
Storage Class Name	cobia: 1wd - 1wt (H30)
Storage Class Type	Disk
Default Warning Threshold	80 percent
Default Critical Threshold	90 percent

Threshold Table

Subsys ID	Subsys Name	Warning	Critical
1	Subsys 1	90	95
2	Subsys 2	default	default

Change selected Warning Threshold to 90 percent

Change selected Critical Threshold to 95 percent

Figure 6-16 Storage Subsystem-Specific Thresholds window

The center of the **Storage Subsystem-Specific Thresholds** window includes a section labeled **Threshold Table**. The **Threshold Table** contains a single line for each storage subsystem configured in the HPSS. For each subsystem, this line indicates the Warning and Critical Thresholds being used for that storage class. The text **default** indicates that, within the specified subsystem, the storage class is using the default thresholds configured on the Storage Class Configuration window. If a value other than **default** is present, then an override value has been specified for the given storage subsystem. Note that the override values may be the same as the default thresholds.

To override the default Warning or Critical Thresholds for a storage class within a given storage subsystem, select that subsystem in the **Threshold Table** on the **Storage Subsystem-Specific Thresholds** window. Enter the desired override values in the **Change selected Warning Threshold**

to and **Change selected Critical Threshold to** and press [Enter]. The override values will be applied in the **Threshold Table**. Use the **Update** button at the bottom of the screen to apply these changes to metadata. Changes to the override values are accomplished the same way.

To delete the override values, select the desired storage subsystem in the **Threshold Table** and click on the **Set To Defaults** button. The text "default" will be returned to the **Threshold Table** for both thresholds in the selected subsystem. Use the **Update** button at the bottom of the screen to apply these changes to metadata.

The Migration/Purge Servers must be restarted in order for a change in the Warning and Critical Thresholds to take effect. If the default thresholds are changed on the Storage Class Configuration window, the Migration/Purge Servers in all storage subsystems using the default thresholds for this storage class must be restarted. If subsystem specific thresholds are added, removed, or changed, the Migration/Purge Servers for those storage subsystems must be restarted.

6.7.1.3 Storage Subsystem-Specific Thresholds Variables

The following table (Table 6-12) describes the fields on the **Storage Subsystem-Specific Thresholds** SSM window:

Table 6-12 Storage Subsystem-Specific Thresholds Variables

Display Field Name	Description	Acceptable Values	Default Values
Storage Class ID	The ID of the Storage Class to which these subsystem-specific thresholds apply.	Any valid Storage Class ID	[Display only]
Storage Class Name	The Descriptive Name of the Storage Class.	Any valid Storage Class Descriptive Name	[Display only]
Storage Class Type	The Type of Storage Class.	DISK, TAPE	[Display only]
Default Warning Threshold	The Default Warning Threshold for this Storage Class.	An integer between 1 and 100	[Display only]
Default Critical Threshold	The Default Critical Threshold for this Storage Class.	An integer between 1 and 100	[Display only]

Table 6-12 Storage Subsystem-Specific Thresholds Variables

Display Field Name	Description	Acceptable Values	Default Values
Change selected Warning Threshold to	Override value of the Storage Class Warning Threshold for the selected subsystem.	For disk, any integer percentage between 1 and 100 (inclusive) which is less than or equal to the Critical Threshold. For tape, any non-negative integer VV count which is greater than or equal to the Critical Threshold. Note that the threshold for a disk class may be disabled by setting this value to 100 percent, and the threshold for a tape class may be disabled by setting this value to zero.	None
Change selected Critical Threshold to	Override value of the Storage Class Critical Threshold for the selected subsystem.	For disk, any integer percentage between 1 and 100 (inclusive) which is greater than or equal to the Warning Threshold. For tape, any non-negative integer VV count which is less than or equal to the Warning Threshold. Note that the threshold for a disk class may be disabled by setting this value to 100 percent. The threshold for a tape class may be disabled by setting this value to zero.	None

6.7.2 Configure the Storage Hierarchies

Storage hierarchy information must be created for each storage hierarchy that is to be supported by the HPSS system. An HPSS storage hierarchy can be created using the HPSS Storage Hierarchy window. After the configuration entry is created, it can be viewed, updated or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Storage Hierarchies** option. The **HPSS Hierarchies** window will be displayed. To create a new hierarchy, click on the "**New Hierarchy ...**" button.

The Storage Hierarchy Configuration window will be displayed as shown in Figure 6-17. The fields are displayed with default values for a new storage hierarchy. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

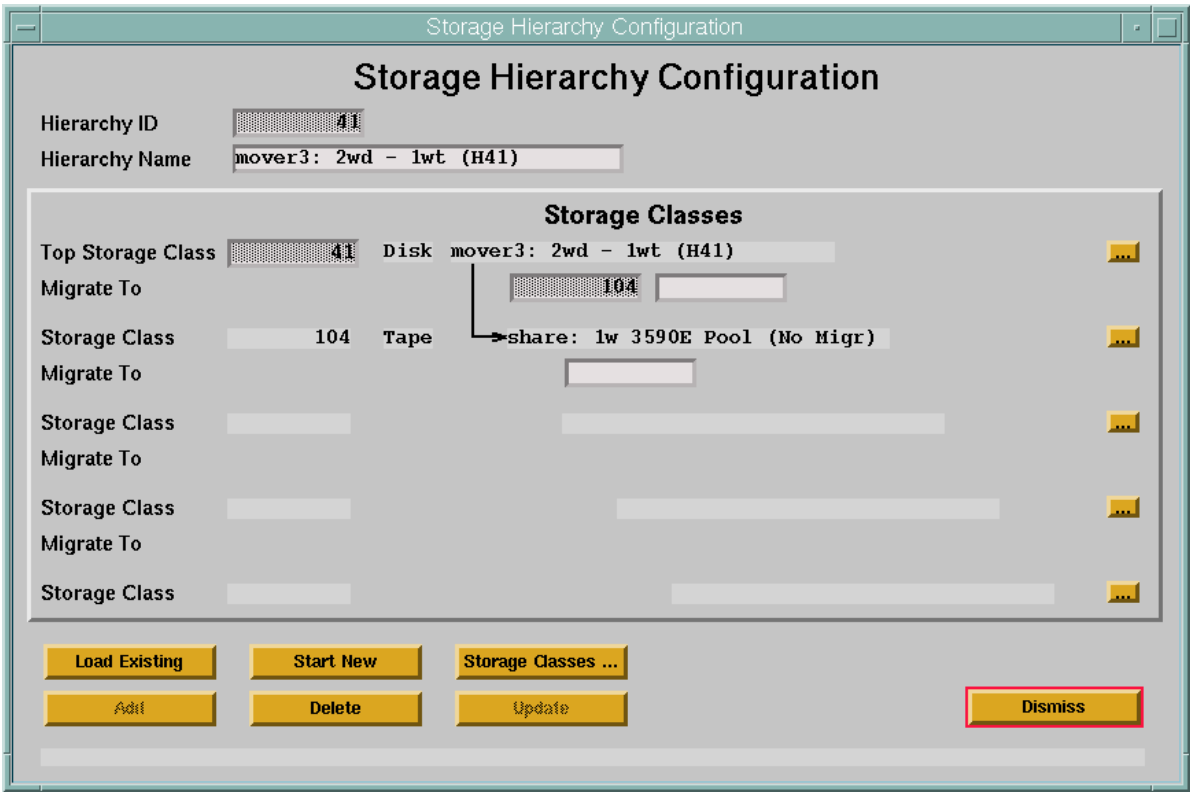
To update an existing storage hierarchy, select the **Load Existing** button on the Storage Hierarchy Configuration window and select the desired storage hierarchy from the popup list. The window will be refreshed with the data from the selected storage hierarchy. After modifying the data, click on the **Update** button to write the changes to the SFS file. Refer to Section 3.11.2: *Changing Storage Hierarchy Definition* (page 80) in the *HPSS Management Guide* for more guidelines on changing a storage hierarchy configuration.

To delete an existing storage hierarchy, select the **Load Existing** button on the Storage Hierarchy Configuration window and select the desired storage hierarchy from the popup list. The window will be refreshed with the configured data. Click on the **Delete** button to delete the storage hierarchy. Refer to Section 3.12.2: *Deleting Storage Hierarchy Definition* (page 81) in the *HPSS Management Guide* for more guidelines on deleting a storage hierarchy configuration.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

The following rules for creating valid storage hierarchies are enforced by the **Storage Hierarchy Configuration Window**:

1. A given storage class may only be used once in a given hierarchy.
2. Disk migration may migrate to one or more target levels in the hierarchy. The first target level must be immediately below the migrating level. Subsequent target levels must also be below the migrating level, but need not be consecutive.
3. Tape migration may only migrate to a single target level in the hierarchy. This target level must be immediately below the migrating level.
4. A given storage class may only be a migration target for a single level above it in the hierarchy.



Hierarchy ID 41
Hierarchy Name mover3: 2wd - 1wt (H41)

Top Storage Class 41 Disk mover3: 2wd - 1wt (H41)
Migrate To 104
Storage Class 104 Tape share: 1w 3590E Pool (No Migr)
Migrate To
Storage Class
Migrate To
Storage Class
Migrate To
Storage Class

Figure 6-17 Storage Hierarchy Configuration Window

6.7.2.1 *Storage Hierarchy Configuration Variables*

Table 6-13 lists the fields on the Storage Hierarchy Configuration window and provides specific recommendations for configuring the Storage Hierarchy for use by HPSS.

Table 6-13 Storage Hierarchy Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Hierarchy ID	The unique, numeric ID associated with this hierarchy.	Any unique, non-zero, positive 32-bit integer value.	Last configured ID plus 1.
Hierarchy Name	The descriptive name associated with this hierarchy.	Any character string up to 31 bytes in length.	Hierarchy ID
	Advice: A hierarchy's descriptive name should be meaningful to local site administrators and operators.		
Top Storage Class	The name of the storage class associated with the top level in the hierarchy.	Any configured storage class name.	None
Mid-Top Storage Class	The name of the storage class associated with the mid-top level in the hierarchy.	Any configured storage class name.	None
Middle Storage Class	The name of the storage class associated with the middle level in the hierarchy.	Any configured storage class name.	None
Mid-Bottom Storage Class	The name of the storage class associated with the mid-bottom level in the hierarchy.	Any configured storage class name.	None
Bottom Storage Class	The name of the storage class associated with the bottom level in the hierarchy.	Any configured storage class name.	None
Advice: No two hierarchy levels can reference the same storage class.			

6.7.3 *Configure the Classes of Service*

Class of Service (COS) information must be created for each class of service that is to be supported by the HPSS system. A COS can be created using the HPSS Class of Service window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

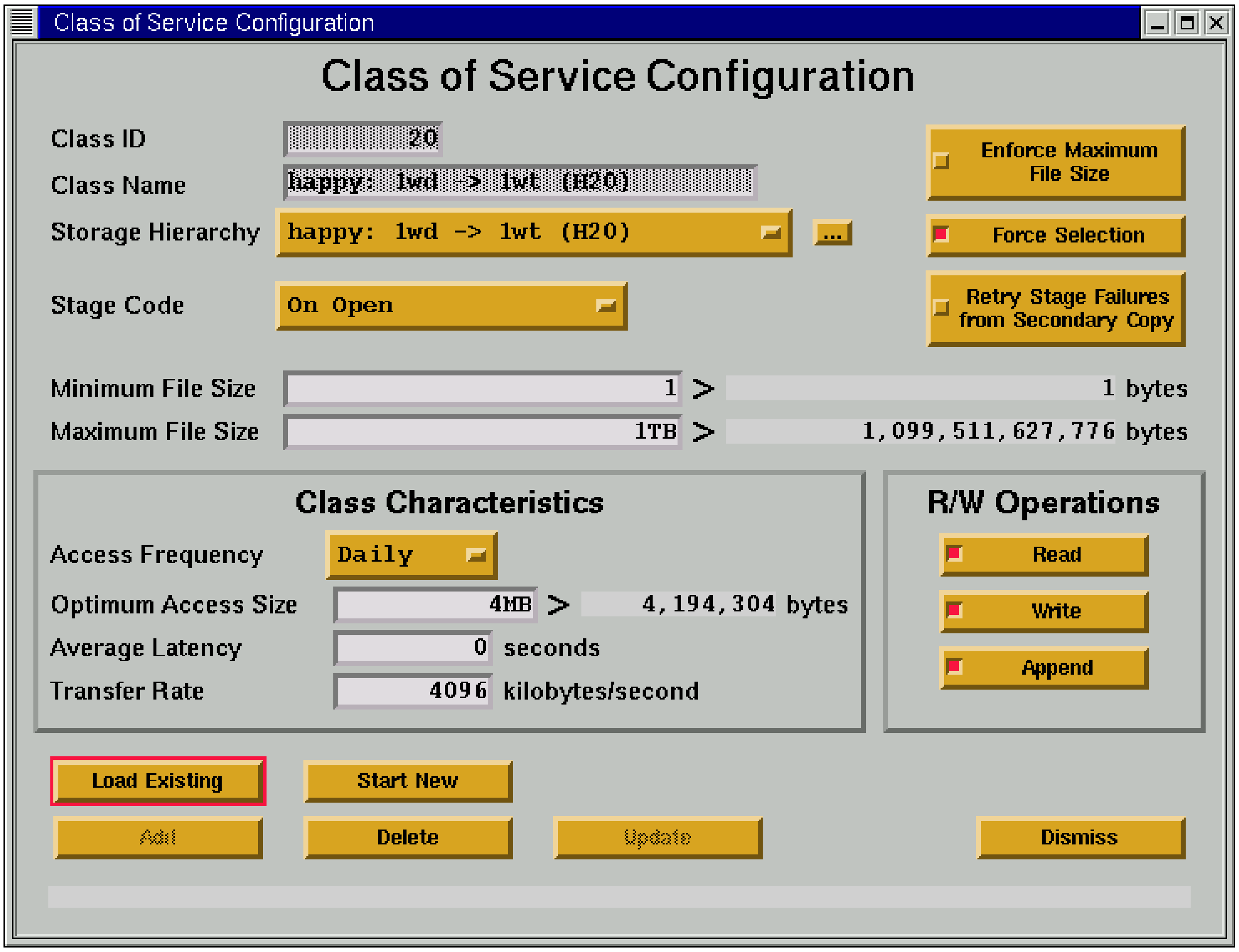
From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Classes of Service** option.

The Class of Service Configuration window will be displayed as shown in Figure 6-18. The fields are displayed with default values for a new class of service. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing class of service, select the **Load Existing** button on the Class of Service Configuration window and select the desired class of service from the popup list. The window will be refreshed with the configured data. After modifying the data, click on the **Update** button to write the changes to the SFS file. Refer to Section 3.11.3: *Changing Class of Service Definition* (page 80) in the *HPSS Management Guide* for more guidelines on changing a class of service configuration.

To delete an existing class of service, select the **Load Existing** button on the Class of Service Configuration window and select the desired class of service from the popup list. The window will be refreshed with the configured data. Click on the **Delete** button to delete the class of service. Refer to Section 3.12.3: *Deleting Class of Service Definition* (page 82) in the *HPSS Management Guide* for more guideline on deleting a class of service configuration.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.



Class of Service Configuration

Class ID 20

Class Name happy: 1wd -> 1wt (H20)

Storage Hierarchy happy: 1wd -> 1wt (H20)

Stage Code On Open

Enforce Maximum File Size

[✓ Force Selection]

Retry Stage Failures from Secondary Copy

Minimum File Size 1 > 1 bytes

Maximum File Size 1TB > 1,099,511,627,776 bytes

Class Characteristics

Access Frequency Daily

Optimum Access Size 4MB > 4,194,304 bytes

Average Latency 0 seconds

Transfer Rate 4096 kilobytes/second

R/W Operations

[✓ Read]

[✓ Write]

[✓ Append]

Figure 6-18 Class of Service Configuration Window

6.7.3.1 *Class of Service Configuration Variables*

Table 6-14 lists the fields on the HPSS Class of Service window and provides Class of Service configuration information.

Table 6-14 Class of Service Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Class ID	An unique integer ID for the COS.	Any non-zero, positive 32-bit integer value. [Only modifiable at create time]	Last configured COS ID plus 1.

Table 6-14 Class of Service Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Class Name	The descriptive name of the COS.	A character string up to 31 bytes in length. [Only modifiable at create time]	None
	Advice: Select a name that describes the COS in some functional way. A good example would be High Speed Disk Over Tape.		
Storage Hierarchy	The name of the storage hierarchy associated with this COS.	Any configured hierarchy name.	Same as last configured COS.
Stage Code	A code that indicates the file staging options.	On Open, On Open Async, No Stage On Open Background	Same as last configured COS.
	Advice: Refer to Section 2.9.3.3 for more information on selecting the appropriate stage code.		
Minimum File Size	The size, in bytes, of the smallest bitfiles supported by this COS.	Any positive 64-bit integer value.	Same as last configured COS.
Maximum File Size	The size, in bytes, of the largest bitfile supported by this COS.	Any positive 64-bit integer value	Same as last configured COS.
Enforce Maximum File Size	A flag that indicates that a bitfile larger than the Maximum File Size cannot be created in this COS.	ON, OFF	Same as last configured COS.
Force Selection	A flag to determine how a COS will be selected. If ON, a client must explicitly select this COS in order to have a file assigned to it; if the client merely supplies general COS hints for a file, this COS will not be selected	ON, OFF	Same as last configured COS.

Table 6-14 Class of Service Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Retry Stage Failures from Secondary Copy	When this flag is turned on, HPSS will automatically retry a failed stage from the primary copy if a valid secondary copy exists. For this to work properly, the COS must be set up with at least 2 copies and a valid second copy must have been created during HPSS migration processing.	ON, OFF	Same as last configured COS.
R / W Operations	The operations supported for this COS.	READ, WRITE, APPEND	Same as last configured COS.
Class Characteristics. The fields below describe the characteristics of a COS.			
Access Frequency	The frequency, on average, for accessing files in this COS.	Hourly, Daily, Weekly, Monthly, Archive	Daily
Optimum Access Size	The suggested number of bytes that should be written at one time for maximum efficiency. Not currently used.	Any positive 32-bit integer value.	Same as last configured COS.
Average Latency	The average time (in seconds) that elapses between the time a transfer request is accepted for processing and the time the data transfer begins.	Any positive 32-bit integer value.	Same as last configured COS.
Transfer Rate	The average throughput (in KB per second) that can be transferred using this COS.	Any positive 32-bit integer value.	Same as last configured COS.

6.7.4 *File Family Configuration*

File family information must be created for each file family that is to be supported by the HPSS system. A file family can be created using the File Family Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

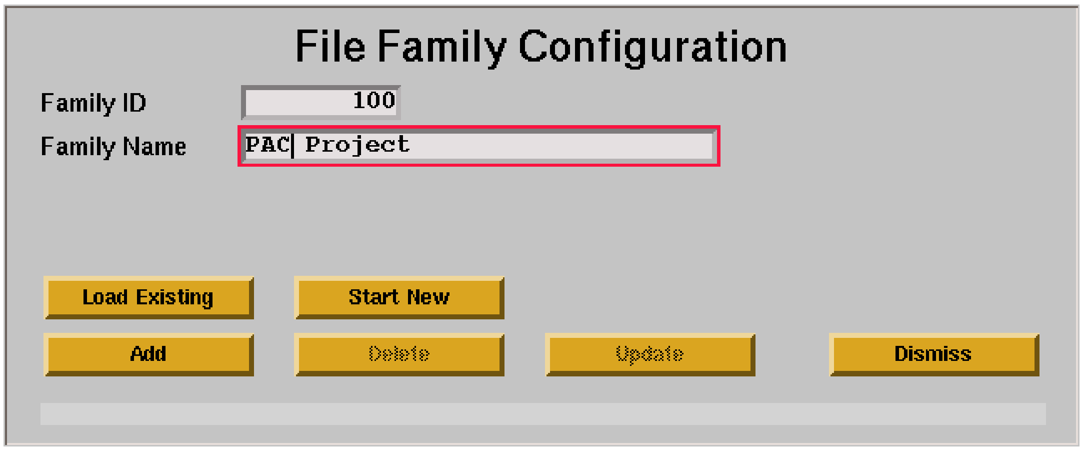
From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **File Families** option.

The File Family Configuration window will be displayed as shown in Figure 6-19. The fields are displayed with default values for a file family. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing file family, select the **Load Existing** button on the File Family Configuration window and select the desired family from the popup list. The window will be refreshed with the configured data. After modifying the data, click on the **Update** button to write the changes to the SFS file.

To delete an existing file family, select the **Load Existing** button on the File Family Configuration window and select the desired family from the popup list. The window will be refreshed with the configured data. Click on the **Delete** button to delete the configuration.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.


File Family Configuration

Family ID: 100

Family Name: PAC Project

Figure 6-19 File Family Configuration Window

6.7.4.1 Configure File Family Variables

Table 6-15 describes the file family variables.

Table 6-15 Configure File Family Variables

Display Field Name	Description	Acceptable Values	Default Value
Family ID	An unsigned non-zero integer which serves as a unique identifier for this file family. A unique default value is provided, which may be overwritten if desired. However, if an ID which is already in use by another file family is specified, the Add request will fail. This field may only be modified in Add mode.	Any non-zero, positive 32-bit integer value.	Last configured family ID plus 1
Family Name	A text string which serves as a descriptive identifier for this file family. The name should be unique among all file families. The name should also be informative.	A character string up to 31 characters in length.	File Family “ID”

6.8 Specific Server Configuration

In addition to creating the basic server configuration entries, a specific server configuration entry must be created for each HPSS server of the following types:

- Bitfile Server
- DMAP Gateway
- Gatekeeper
- Log Client
- Log Daemon
- Metadata Monitor
- Migrate/Purge Server

- Mover
- Name Server
- NFS Daemon
- Non-DCE Client Gateway
- Physical Volume Library
- Physical Volume Repository
- Storage Server

Sections 6.8.1 through 6.8.14 describe the specific configuration for each of the above servers.

The SSM servers, Location Servers, NFS Mount Daemons, and Startup Daemons do not have specific configurations.

6.8.1 *Bitfile Server Specific Configuration*

The Bitfile Server specific configuration entry can be created using the Bitfile Server Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

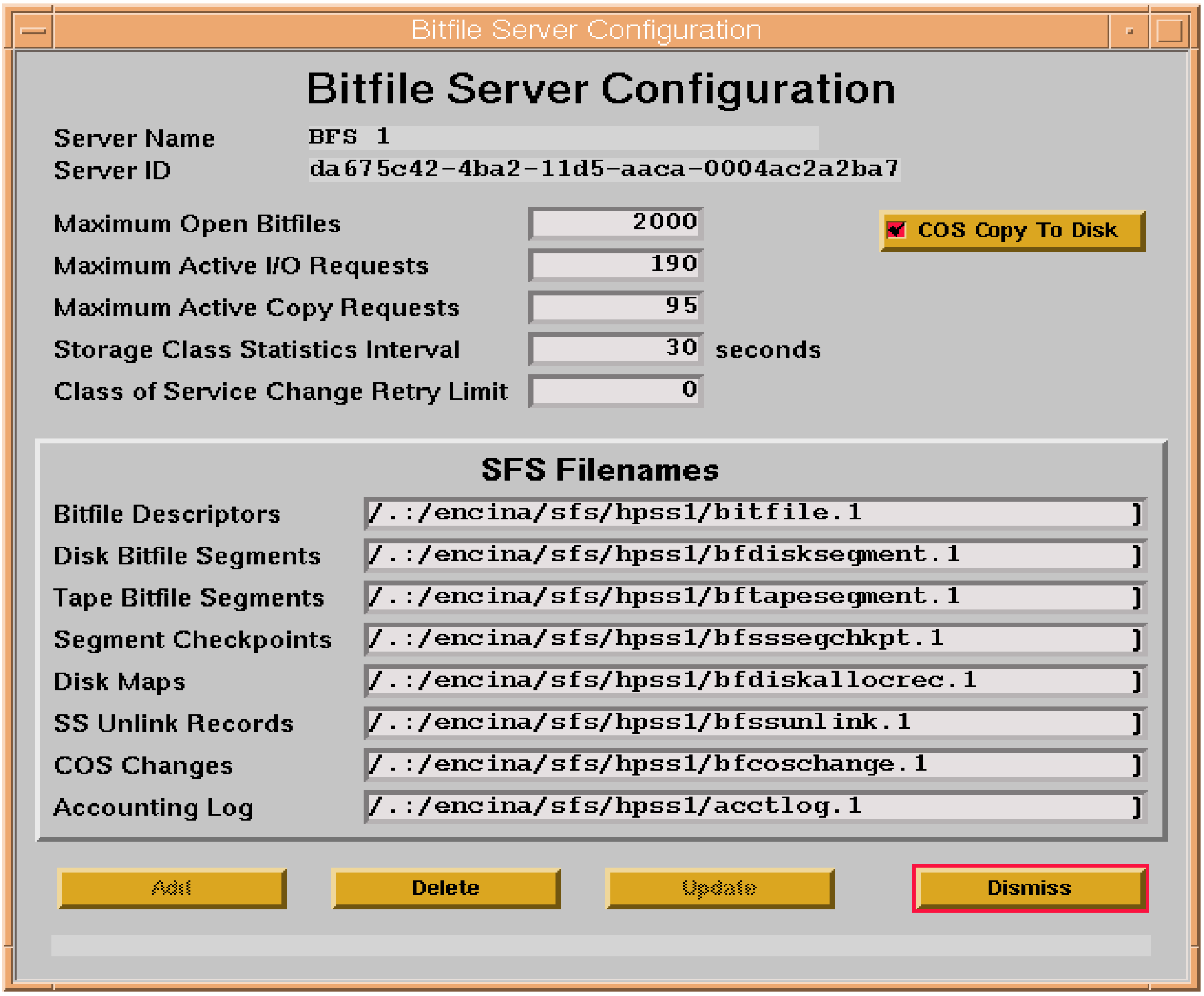
From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 6-5.

To add a new specific configuration, select the Bitfile Server entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The Bitfile Server Configuration window will be displayed as shown in Figure 6-20 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Bitfile Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Bitfile Server Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the Bitfile Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Bitfile Server Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.



Bitfile Server Configuration

Server Name	BFS 1
Server ID	da675c42-4ba2-11d5-aaca-0004ac2a2ba7
Maximum Open Bitfiles	2000
Maximum Active I/O Requests	190
Maximum Active Copy Requests	95
Storage Class Statistics Interval	30 seconds
Class of Service Change Retry Limit	0
COS Copy To Disk	✓

SFS Filenames

Bitfile Descriptors	/.:/encina/sfs/hpss1/bitfile.1
Disk Bitfile Segments	/.:/encina/sfs/hpss1/bfdisksegment.1
Tape Bitfile Segments	/.:/encina/sfs/hpss1/bftapesegment.1
Segment Checkpoints	/.:/encina/sfs/hpss1/bfsssegchkpt.1
Disk Maps	/.:/encina/sfs/hpss1/bfdiskallocrec.1
SS Unlink Records	/.:/encina/sfs/hpss1/bfssunlink.1
COS Changes	/.:/encina/sfs/hpss1/bfcoschange.1
Accounting Log	/.:/encina/sfs/hpss1/acctlog.1

Figure 6-20 Bitfile Server Configuration Window

6.8.1.1 Bitfile Server Configuration Variables

Table 6-16 lists the fields on the Bitfile Server Configuration window and provides specific recommendations for configuring the BFS for use by HPSS.

Table 6-16 Bitfile Server Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	Descriptive name of the BFS. This name is copied over from the BFS general configuration entry.	This field cannot be modified. It is displayed for reference only.	Selected BFS descriptive name.
Server ID	The UUID of the Bitfile Server. This ID is copied over from the BFS general configuration entry.	This field cannot be modified. It is displayed for reference only.	Extracted from the BFS general configuration entry.
Maximum Open Bitfiles	The maximum number of bitfiles that can be open concurrently in the BFS.	Any positive 32-bit integer value.	2000
Maximum Active I/O Requests	The maximum number of I/O type requests that can be open concurrently in the BFS.	Any positive 32-bit integer value.	190
	Advice: This value includes all requests that might result in data movement except migrate requests. This includes read, write, copy file and all non-background type stage requests. If an open request results in any stage other than a background stage, the open request is also counted. This value should be set high enough to cover the maximum number of active and queued requests expected in the system. This value cannot be higher than (the maximum number of BFS threads - 4). Since migrate requests are not counted, the maximum number of concurrent migrate requests expected should be computed and this value should be set no higher than ((the number of BFS threads - 4) - max migrate requests). Each storage class that is in the process of being migrated will generate 1 migrate request for each copy being created.		
Maximum Active Copy Requests	The maximum number of copy type requests that can be active concurrently in the BFS.	Any positive 32-bit integer value.	95
	Advice: This value counts only copy file requests and various forms of stage requests. For copy file requests and non-background type stage requests, this limit results in the requests being given a BUSY error when the request is generated. Background stage requests are queued internally to BFS without using thread resources and up to 2000 can be queued before a BUSY error will be returned. This value will limit the actual number of threads that are busy in the background processing the background stage requests. The copyfile command that is generated by the background COS change thread is included in this count. This count does not include migrate requests.		

Table 6-16 Bitfile Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Storage Class Statistics Interval	An interval in seconds that indicates how often the BFS needs to contact each SS to get up-to-date statistics on each storage class that the SS manages. This information is used in load balancing across multiple storage classes.	Any positive 32-bit integer value.	180
	Advice: For systems with mostly disk or small files, the number should be set to a short interval (between 30 and 60 seconds). For systems with mostly large or tape files, this value should be set higher (e.g., 1 to 5 minutes).		
Class of Service Change Retry Limit	Max number of times a COS change 'retry forever' is processed on a bitfile before the request to change the COS of the bitfile is rejected.	0 - $(2^{(32)}-1)$	0
	Note: A SIGUSR1 signal can be sent to the Bitfile Server to toggle emptying the COS change file and thus rejecting all COS changes. When BFS is initialized, it is set up to honor all COS change requests. If you send a SIGUSR1 signal to BFS, it will then delete all COS change file requests that are placed in the COS change file, until a second SIGUSR1 signal is sent which will reinstate the default behavior of honoring COS change requests. When the signal is sent, a message will appear in the Alarms and Events window which will indicate the new status. This feature should be used very carefully! *The default value of 0 tells the BFS to retry indefinitely.*		

Table 6-16 Bitfile Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
COS Copy To Disk	A flag affecting the COS changes for a bitfile. By default, when the COS of a bitfile is changed, the BFS copies the file to the highest level tape storage class in the target hierarchy. If this flag is ON, and if the target hierarchy has a disk storage class as its highest level, the BFS will copy the file to that disk storage class. Otherwise, the bitfile will be copied to the highest tape level.	ON, OFF	OFF
	Advice: This flag should be ON when changing the COS of a file to a duplicate (2-copies) COS. Otherwise, only the first copy of the file will be created on tape. The second copy will not be created until the file is staged to disk, modified and then migrated to tape.		
SFS Filenames. The fields below list the names of the SFS files used by the BFS.			
Bitfile Descriptors	The file name of the SFS file where the bitfile descriptor information is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/bitfile.#
Disk Bitfile Segments	The file name of the SFS file where the bitfile disk segment information is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/bfdisksegment.#
Tape Bitfile Segments	The file name of the SFS file where the bitfile tape segment information is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/bftapesegment.#
Segment Checkpoints	The file name of the SFS file where the BFS checkpoints are stored.	Valid Encina file name.	/.:/encina/sfs/hpss/bfsssegchkpt.#
Disk Maps	The file name of the SFS file where the bitfile disk map information is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/bfdiskallocrec.#

Table 6-16 Bitfile Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
SS Unlink Records	The file name of the SFS file where the information representing storage segments to be unlinked is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/bfssunlink.#
COS Changes	The file name of the SFS file where the information indicating which bitfiles need to have the COS changed is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/bfcoschange.#
Accounting Log	The file name of the SFS file where accounting log records are stored.	Valid Encina file name.	/.:/encina/sfs/hpss/acctlog.#

6.8.2 DMAP Gateway Specific Configuration

The DMAP Gateway specific configuration entry can be created using the DMAP Gateway Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

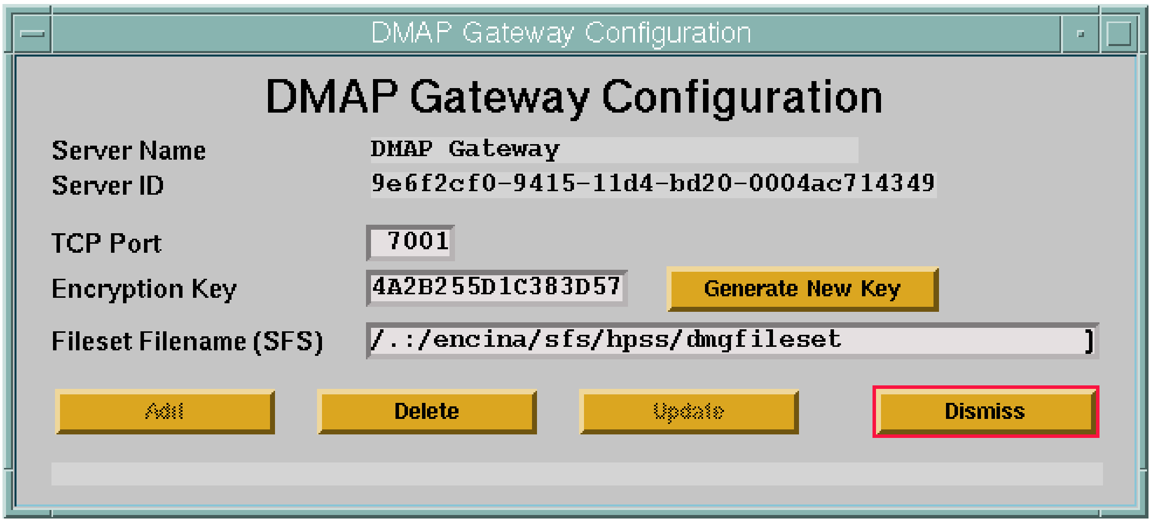
From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 6-5.

To add a new specific configuration, select the DMAP Gateway entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The DMAP Gateway Configuration window will be displayed as shown in Figure 6-21 with default values. If the default data is not desired, change the fields to the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the DMAP Gateway entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The DMAP Gateway Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the DMAP Gateway entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The DMAP Gateway Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.



DMAP Gateway Configuration

Server Name: DMAP Gateway
Server ID: 9e6f2cf0-9415-11d4-bd20-0004ac714349
TCP Port: 7001
Encryption Key: 4A2B255D1C383D57
Fileset Filename (SFS): /.:/encina/sfs/hpss/dmgfileset

Figure 6-21 DMAP Gateway Configuration Window

6.8.2.1 *DMAP Gateway Configuration Variables*

Table 6-17 lists the fields on the HPSS DMAP Gateway Configuration window and provides specific recommendations for configuring the DMAP Gateway for use by HPSS.

Table 6-17 DMAP Gateway Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the DMAP Gateway. This name is copied over from the selected DMAP Gateway general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected DMAP Gateway descriptive name.
Server ID	The UUID of the DMAP Gateway. This ID is copied over from the selected DMAP Gateway general configuration entry.	This field cannot be modified. It is displayed for reference only.	Extracted from the DMAP Gateway general configuration entry.
TCP Port	The TCP port number used by the DMAP Gateway to listen for requests from HPSS/DMAP servers.	Any integer from 1 to 65,535.	7001

Table 6-17 DMAP Gateway Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Encryption Key	A number used as an encryption key in message passing. A specific value can be typed in, or the **Generate New Key** button can be clicked to generate a random key value.	Any positive 64-bit integer, displayed as hexadecimal.	0
	Advice: Should not use key 0 which implies no protection. This key must be identical to the one defined in the /var/hpss/hdm/hdm<id>/gateways.dat file.		
Fileset Filename (SFS)	The name of the Encina SFS file containing the DMAP Gateway fileset information.	Valid Encina file name.	/.:/encina/sfs/hpss/dmgfileset

6.8.3 Gatekeeper Specific Configuration

The GK specific configuration entry can be created using the Gatekeeper Server Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

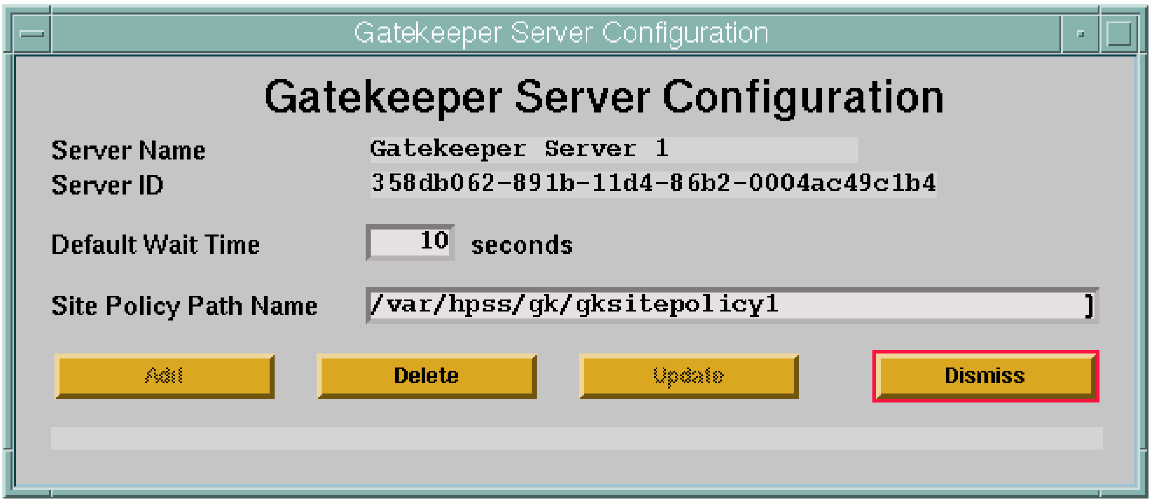
From the HPSS Health and Status window (Figure 6-1 on page 252), click on the **Admin** menu, select the **Configure HPSS** option and click on the "Servers" option. The HPSS Servers window will be displayed as show in Figure 6-5 on page 264.

To add a new specific configuration, select the Gatekeeper Server entry and click on the **Type-specific...** button from the **Configuration** button group on the **HPSS Servers** window. The **Gatekeeper Server Configuration** window will be displayed as shown in Figure 6-22 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Gatekeeper Server entry on the **HPSS Server** window and click on the **Type-specific...** button from the **Configuration** button group. The **Gatekeeper Server Configuration** window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the Gatekeeper Server entry on the **HPSS Servers** window and click on the **Type-specific...** button from the **Configuration** button group. The **Gatekeeper Server Configuration** window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.


Gatekeeper Server Configuration

Server Name: Gatekeeper Server 1
Server ID: 358db062-891b-11d4-86b2-0004ac49c1b4
Default Wait Time: 10 seconds
Site Policy Path Name: /var/hpss/gk/gksitepolicy1

Figure 6-22 Gatekeeper Server Configuration window

To use a Gatekeeper Server for Gatekeeping Services then the Gatekeeper Server must also be configured into the Storage Subsystem (see Section 6.4: *Storage Subsystems Configuration* on page 259).

To use the Gatekeeper Server for Account Validation Services, then the **Account Validation** button of the Accounting Policy must be ON (see Section 6.6.3: *Configure the Accounting Policy* on page 289).

6.8.3.1 Gatekeeper Configuration Variables

Table 6-18: *Gatekeeper Configuration Fields* on page 332 lists the fields on the Gatekeeper Server Configuration window and provides specific recommendations for configuring the GK for use by HPSS.

Table 6-18 Gatekeeper Configuration Fields

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the GK. This name is copied from the GK general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected GK descriptive name.
Server ID	The UUID of the GK. This ID is copied over from the GK general configuration entry.	This field cannot be modified. It is displayed for reference only.	Extracted from the GK general configuration entry.

Table 6-18 Gatekeeper Configuration Fields

Display Field Name	Description	Acceptable Values	Default Value
Default Wait Time	The default number of seconds the client will wait before retrying a request if not determined by the Site Interface.	The value must be greater than zero and is only used if the Site Interface returns a wait time of zero for the create, open, or stage request being retried.	10
	Note: The Client API uses the environment variable HPSS_GKTOTAL_DELAY to place a maximum limit on the number of seconds a call will delay because of HPSS_ERETRY status codes returned from the Gatekeeper. See Section 7.1: Client API Configuration on page 413 for more information.		
Site Policy Path Name	The UNIX pathname where the site policy will be stored. The contents of this file will be defined by the site and should be coordinated with the site written interface library.	Empty, or any legal UNIX pathname.	/var/hpss/gk/gksitepolicy

6.8.4 Log Client Specific Configuration

The Log Client specific configuration entry can be created using the Logging Client Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

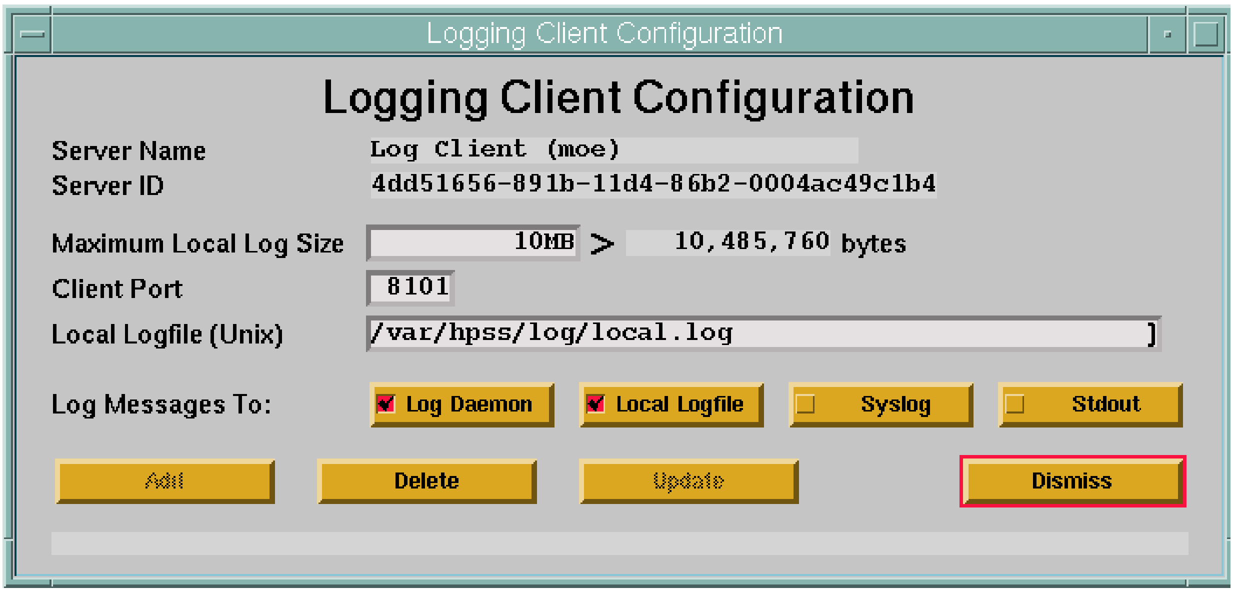
From the **HPSS Health and Status** window (shown in Figure 6-1 on page 252), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The **HPSS Servers** window will be displayed as shown in Figure 6-5 on page 264.

To add a new specific configuration, select the Log Client entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The Logging Client Configuration window will be displayed as shown in Figure 6-23 on page 334 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Log Client entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Logging Client Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the Log Client entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Logging Client Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window



Server Name: Log Client (moe)
Server ID: 4dd51656-891b-11d4-86b2-0004ac49c1b4
Maximum Local Log Size: 10MB > 10,485,760 bytes
Client Port: 8101
Local Logfile (Unix): /var/hpss/log/local.log
Log Messages To: [✓ Log Daemon] [✓ Local Logfile] Syslog Stdout

Figure 6-23 Logging Client Configuration Window

6.8.4.1 *Log Client Configuration Variables*

Table 6-19 lists the fields on the Logging Client Configuration window and provides specific recommendations for configuring a Log Client for use by HPSS.

Table 6-19 Log Client Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the Log Client. This name is copied over from the Log Client general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected Log Client descriptive name.
Server ID	The UUID of the Log Client. This ID is copied over from the Log Client general configuration entry.	The UUID of the Log Client. This field cannot be modified. It is displayed for reference only.	Extracted from the Log Client general configuration entry.

Table 6-19 Log Client Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Maximum Local Log Size	The maximum size in bytes of the local log file. Once this size is reached, the log will be reused in a wraparound fashion. The local log is not automatically archived.	A positive integer up to the maximum file size allowed by the operating system.	5,242,880
Client Port	The port number for communication between the Log Client and the HPSS Servers.	Any positive 32-bit integer value not previously assigned.	8101
	Advice: Ensure that the specified port is not being used by other executing applications. The port number must be a different number than the one used by the Log Daemon.		
Local Logfile (Unix)	The fully qualified path name of the Log Client-formatted log file. If Local LogFile is specified in the **Log Messages To** field, those messages sent to this instance of the Log Client will be formatted and written to the designated file name.	Any valid POSIX path name. The string length (in bytes) is limited to a minimum of the operating system maximum allowed file name size or 1024.	/var/hpss/log/local.log
	Note: The specified file name will contain formatted messages from HPSS applications executing only from the node on which this instance of the Log Client is executing. This option is provided as a convenience feature. All HPSS messages will be written to a central log if Log Daemon is specified in the Log Messages To field.		

Table 6-19 Log Client Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Log Messages To:	A mask of options to apply to local logging.	A combination of the following values: Log Daemon—Send log messages to the central log. Local LogFile—format and log messages from servers on the same node as the Log Client to a local file. It is an error if both Local LogFile and Standard Output are both specified. Syslog—format and log messages locally to syslog. Stdout—Send messages to standard output.	Log Daemon, Local Log File
	Advice: If neither Local LogFile nor Syslog is specified, no local logging will occur. If Log Daemon is not specified, messages from HPSS processes executing on the same node as this Log Client will not be written to the central log. The Syslog option should be used with care. The syslog file will grow without bound until it is deleted/ truncated, or until the file system runs out of space.		

6.8.5 Log Daemon Specific Configuration

The Log Daemon specific configuration entry can be created using the Logging Daemon Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

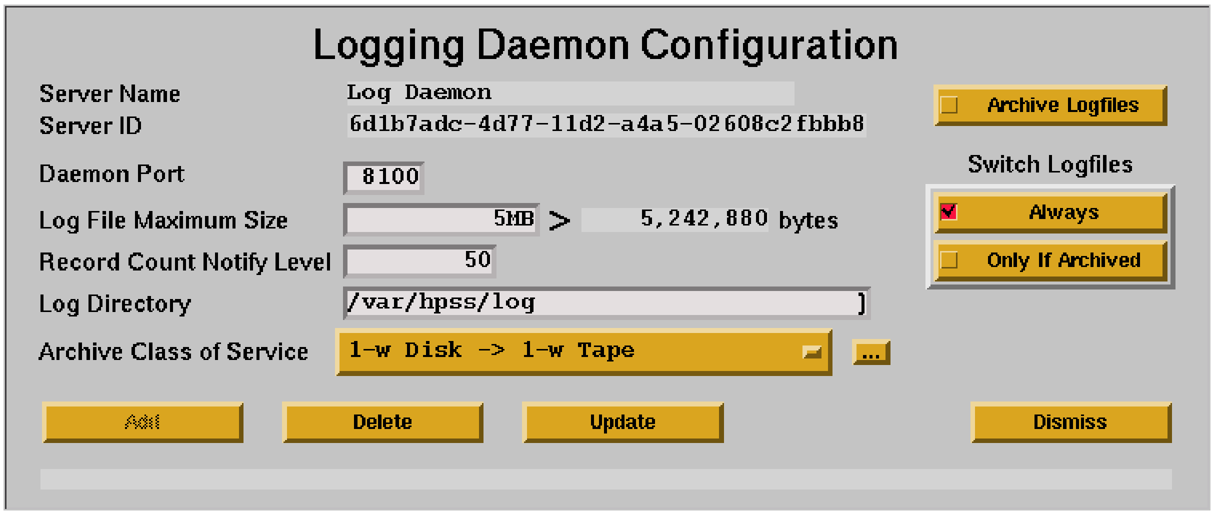
From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 6-5.

To add a new specific configuration, select the Log Daemon entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The Logging Daemon Configuration window will be displayed as shown in Figure 6-24 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Log Daemon entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Logging Daemon Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the Log Daemon entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Logging Daemon Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.


Logging Daemon Configuration

Server Name: Log Daemon
Server ID: 6d1b7adc-4d77-11d2-a4a5-02608c2fbbb8
Daemon Port: 8100
Log File Maximum Size: 5MB > 5,242,880 bytes
Record Count Notify Level: 50
Log Directory: /var/hpss/log
Archive Class of Service: 1-w Disk -> 1-w Tape

☐ Archive Logfiles

Switch Logfiles
☑ Always
☐ Only If Archived

Figure 6-24 Logging Daemon Configuration Window

6.8.5.1 *Log File Archival*

The Log Daemon can be configured to automatically archive log files to HPSS. If log files are to be archived, the **/log** directory must be created in the HPSS root directory. The **/log** directory can be created by the root user using **ftp** as follows:

```
ftp "node" "HPSS Port"
Login as root user
mkdir /log
quote site chown hpss_log /log
quote site chmod 744 /log
```

6.8.5.2 *Log Daemon Configuration Variables*

Table 6-20 lists the fields on the Logging Daemon Configuration window and provides specific recommendations for configuring a Log Daemon for use by HPSS.

Table 6-20 Log Daemon Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the Log Daemon. This name is extracted over from the Log Daemon general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected Log Daemon descriptive name.
Server ID	The UUID of the Log Daemon. This ID is copied over from the Log Daemon general configuration entry.	The UUID of the Log Daemon. This field cannot be modified. It is displayed for reference only.	Extracted from the Log Daemon general configuration entry.
Daemon Port	The port number for communication between the Log Client and the Log Daemon.	Any positive 32-bit integer value not previously assigned.	8100
	Advice: Ensure that the port number assigned does not conflict with any other executing application. The port value must be a different value than the port number in the Log Client configuration entries.		
Log File Maximum Size	The maximum size in bytes of the central log file. Once this size is reached, logging will switch to a second log file. The log file that filled up will then be archived to an HPSS file if the Archive Flag is on.	A positive integer up to the maximum file size allowed by the operating system.	5,242,880
	Note: Since two log files are allocated to accommodate log switching and log file archiving, the amount of space used by logging will be twice the specified value.		
Record Count Notify Level	The difference in the log record count since the previous SSM notification that must be reached to trigger a subsequent notification to SSM when any registered log file attribute changes.	Any positive 32-bit integer value.	50
	Advice: Since log file attributes change each time a log record is written (e.g., current log record), this variable is used to prevent transmission of excessive messages to SSM as a result of logging activity. Setting the variable to a small value will result in increased traffic to SSM.		

Table 6-20 Log Daemon Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Log Directory	The name of the directory in which log files are stored.	Any valid UNIX path name. The string length (in bytes) is limited to a minimum of the operating system maximum allowed file name size, or 1024.	/var/hpss/log
Archive Class of Service	The COS that will determine where HPSS logs are archived. When a log file fills, it will be archived according to this class of service if the Archive Logfiles Flag is set.	Any configured COS name from the pop-up list.	First COS name found in the SFS file.
Archive Logfiles	A flag that indicates whether log files should be automatically archived when they fill.	ON, OFF	OFF
Switch Logfiles	A flag that indicates whether a switch to the second log file should be performed if the archive of the second log file has not yet completed.	Always, Only If Archived	Always
	Advice: Use the default value of Always. Although a log may not get archived, the alternative may be a loss of logged messages to the current log from executing servers.		

6.8.6 Metadata Monitor Specific Configuration

The Metadata Monitor specific configuration entry can be created using the Metadata Monitor Configuration window. After the configuration entry is created, it can be viewed, updated or deleted through the same window.

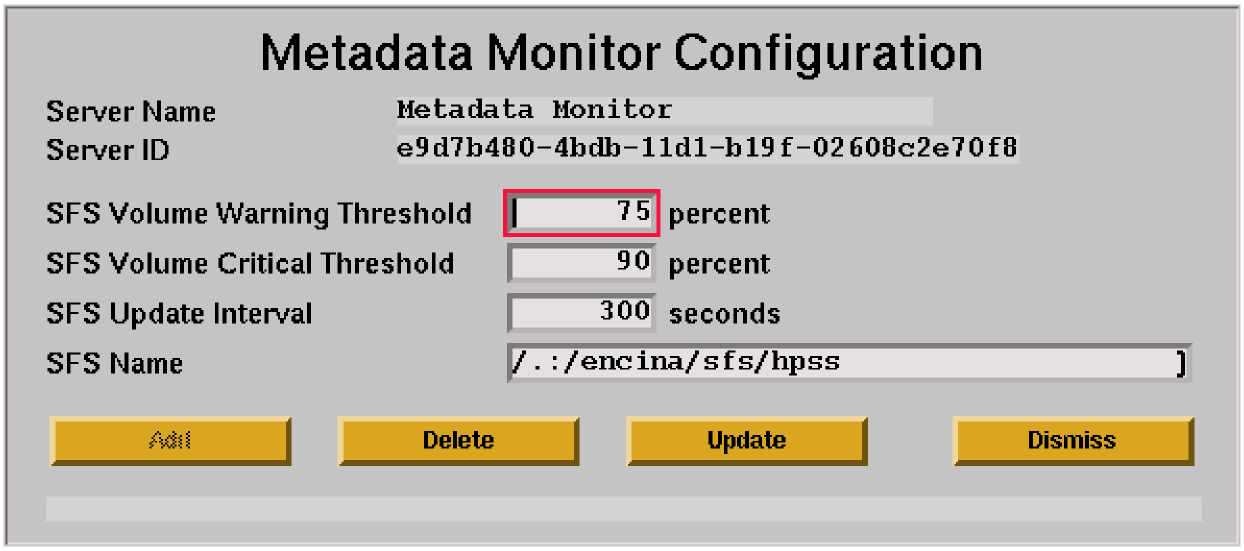
From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 6-5.

To add a new specific configuration, select the Metadata Monitor entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The Metadata Monitor Configuration window will be displayed as shown in Figure 6-25 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Metadata Monitor entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Metadata Monitor Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the Metadata Monitor entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Metadata Monitor Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.


Metadata Monitor Configuration

Server Name	Metadata Monitor	
Server ID	e9d7b480-4bdb-11d1-b19f-02608c2e70f8	
SFS Volume Warning Threshold	75	percent
SFS Volume Critical Threshold	90	percent
SFS Update Interval	300	seconds
SFS Name	/.:/encina/sfs/hpss	

Figure 6-25 Metadata Monitor Configuration window

6.8.6.1 *Metadata Monitor Configuration Variables*

Table 6-21 lists the fields on the Metadata Monitor Configuration window and provides specific recommendations for configuring a MMON for use by HPSS.

Table 6-21 Metadata Monitor Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the MMON. This name is copied over from the MMON general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected MMON descriptive name.
Server ID	The UUID of the Metadata Monitor. This ID is copied over from the MMON general configuration entry.	The UUID of the MMON. This field cannot be modified. It is displayed for reference only.	Extracted from the MMON general configuration entry.
SFS Volume Warning Threshold	A percentage value that indicates when warning alarms should be issued. The MMON will issue warning alarms to SSM when the space used by any SFS volume reaches this percentage.	Integer value between 1 and 100.	75
SFS Volume Critical Threshold	A percentage value that indicates when critical alarms should be issued. The MMON will issue critical alarms to SSM when the space used by any SFS volume reaches this percentage.	Integer value between 1 and 100.	90
SFS Update Interval	An indication of how often (in seconds) the SFS server should be queried.	Integer value between 60 and 600.	300
SFS Name	The name of the Encina SFS server to monitor.	Valid DCE CDS name that refers to a valid Encina SFS server.	/.:/encina/sfs/hpss

6.8.7 Migration Purge Server Specific Configuration

The Migration / Purge Server specific configuration entry can be created using the Migration / Purge Server Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

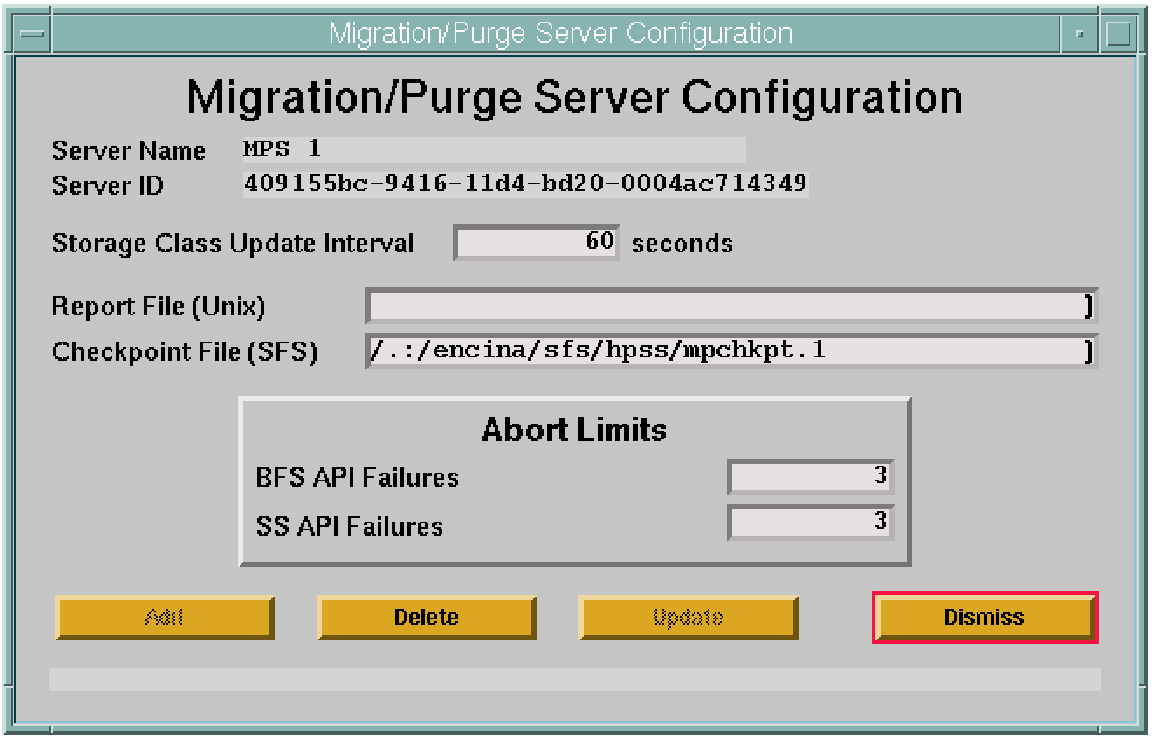
From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 6-5.

To add a new specific configuration, select the Migration / Purge Server entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The Migration / Purge Server Configuration window will be displayed as shown in Figure 6-26 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Migration / Purge Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Migration / Purge Server Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the Migration / Purge Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Migration / Purge Server Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.


Migration/Purge Server Configuration

Server Name: MPS 1
Server ID: 409155bc-9416-11d4-bd20-0004ac714349
Storage Class Update Interval: 60 seconds
Report File (Unix):
Checkpoint File (SFS): /.:/encina/sfs/hpss/mpchkpt.1

Abort Limits
BFS API Failures: 3
SS API Failures: 3

Figure 6-26 Migration / Purge Server Configuration Window

6.8.7.1 *Migration/Purge Server Configuration Variables*

Table 6-22 lists the fields on the HPSS Migration / Purge Server Configuration window and provides specific recommendations for configuring the MPS for use by HPSS.

Table 6-22 Migration/Purge Server Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the MPS. This name is copied over from the selected MPS general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected MPS descriptive name.
Server ID	The UUID of the MPS. This ID is copied over from the selected MPS general configuration entry.	This field cannot be modified. It is displayed for reference only.	Extracted from the MPS general configuration entry.
Storage Class Update Interval	The interval, in seconds, that indicates how often the MPS will query each SS to get the latest data on each storage class that the SS manages. This is also the interval the MPS uses to check periodically to see whether it needs to initiate a purge operation on the storage class based on the associated purge policy.	Any positive integer between 10 and 600.	60 seconds

Table 6-22 Migration/Purge Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Report File (Unix)	A prefix string used by the MPS to construct a report file name. The full file name will consist of this string with a date string and subsystem Id appended to it. This prefix should include a full UNIX path and file name. If a full path is not specified, the location of the migration report files may be unpredictable. A new MPS report file is started every 24 hours. If the MPS reports are not desired, leave this field blank.	Blank -or- Any valid UNIX file name which is writable by MPS.	blank - no reports will be generated.
Checkpoint File	The name of the SFS file where the migration / purge checkpoint information is stored.	Valid Encina file name.	/.:/encina/sfs/hpss/mpchkpt.#
BFS API Failures	For both disk and tape migration, MPS allows a consecutive number of BFS API failures up to the error limit configured in this field. If the configured number of consecutive calls fail during a disk migration run, MPS will skip to the next hierarchy. This number of consecutive failures during a tape migration run leads MPS to abort the run.	Any positive number between 1 and 1000000 (one million)	3
SS API Failures	For tape migration, MPS allows a consecutive number of SS API failures up to the error limit configured in this field. This number of consecutive failures leads MPS to abort the run. This field applies to tape migration only.	Any positive number between 1 and 1000000 (one million)	3

6.8.8 *Mover Specific Configuration*

The Mover (MVR) specific configuration entry can be created using the Mover Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

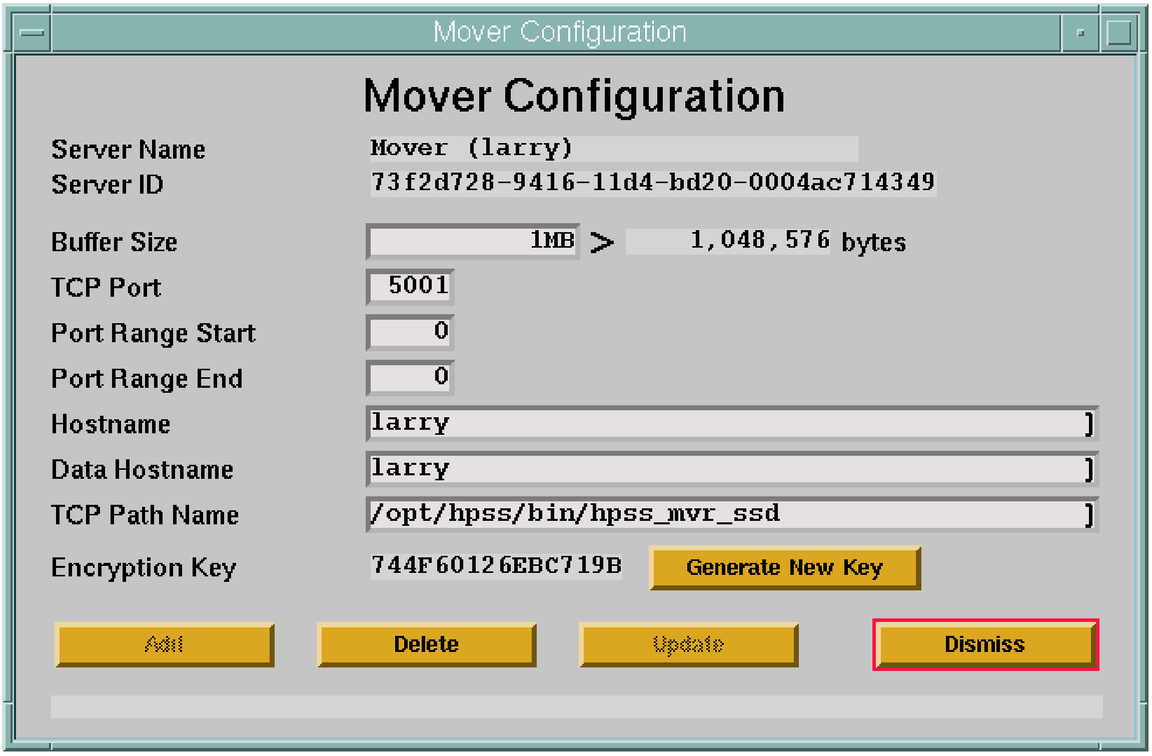
From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 6-5.

To add a new specific configuration, select the Mover entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The Mover Configuration window will be displayed as shown in Figure 6-27 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Mover entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Mover Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the Mover entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Mover Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.



Mover Configuration

Server Name	Mover (larry)
Server ID	73f2d728-9416-11d4-bd20-0004ac714349
Buffer Size	1MB > 1,048,576 bytes
TCP Port	5001
Port Range Start	0
Port Range End	0
Hostname	larry
Data Hostname	larry
TCP Path Name	/opt/hpss/bin/hpss_mvr_ssd
Encryption Key	744F60126EBC719B

Figure 6-27 Mover Configuration window

6.8.8.1 Mover Configuration Variables

Table 6-23 lists the fields on the Mover Configuration window and provides specific recommendations for configuring the MVR for use by HPSS.

Table 6-23 Mover Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the MVR. This name is copied over from the selected MVR general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected MVR descriptive name.

Table 6-23 Mover Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Server ID	The UUID of the MVR. This ID is copied over from the selected MVR general configuration entry.	The UUID of the MVR. This field cannot be modified. It is displayed for reference only.	Extracted from the MVR general configuration entry.
Buffer Size	The buffer size (of each buffer) used for double buffering during data transfers.	The minimum mover buffer size is the size of smallest block size for any device the Mover will handle. The maximum value will be bounded by the available system memory and the number of concurrent mover requests anticipated.	1,048,576
	Advice: This value should be tuned based on device and networking configuration and usage. The trade-off for this value is that large buffer sizes will use more system memory and may be inefficient for small transfers (e.g., if the mover buffer size is 4MB, but client requests are 512KB, the Mover will not achieve any double buffering benefit because the entire amount of the transfer fits in one mover buffer). A smaller buffer size will cause device and network I/O to be broken more often, usually resulting in reduced throughput rates for all but the smallest transfers.		
TCP Port	The TCP/IP port number used for the TCP/IP listen process to receive connections.	Should be a value over 5000.	5001
	Advice: The port number must be unique for MVRs running in the same node. *For Movers running in non-DCE mode, the Mover will internally use the port one greater than the one configured in this field on the non-DCE/Encina platform during initialization (e.g., if 5001 is entered, port 5002 is used on the non-DCE/ Encina node); therefore available port ranges on both nodes must be taken into consideration when selecting this value.*		

Table 6-23 Mover Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Port Range Start	The beginning of a range of local TCP/IP port numbers to be used by the Mover when connecting to clients (required by some sites for communication across a firewall). If zero, the operating system will select the port number; otherwise the Mover selects a local port number between Port Range Start and Port Range End (inclusive).	Zero or any valid TCP port number to which the Mover may bind (that is less than or equal to the value of Port Range End).	0
	Advice: If non-zero, this field must be less than or equal to Port Range End. If this field is zero, Port Range End field must also be zero.		
Port Range End	Used in conjunction with Port Range Start (See above).	Zero or any valid TCP port number to which the Mover may bind (that is greater than or equal to the value of Port Range Start).	0
	Advice: If non-zero, this field must be equal or greater than Port Range Start. If this field is zero, Port Range Start field must also be zero.		
Hostname	The name of the host interface used by the TCP/IP listen process.	Valid host name for the network interfaces on the MVR machine.	Extracted from the MVR general configuration entry.
	Advice: If the Mover is running in non-DCE mode, this field must correspond to a network interface on the non-DCE/Encina node (to which the Storage Servers and PVL will connect when communicating with the Mover, see Section 6.8.8.2: MVR Configuration to Support Non-DCE Execution Mode on page 349), whereas the Execute Hostname set in the Mover's basic configuration corresponds to a network interface on which the Mover DCE/Encina processes run.		
Data Hostname	The host network interface name to be used by the Mover when creating listen ports for data transfers (used during migration, repacking, and staging operations).	Valid host name for the network interfaces on the MVR machine.	Extracted from the MVR general configuration entry.
	Advice: If the Mover is running in non-DCE mode, this field must correspond to a network interface on the non-DCE/Encina node (see Section 6.8.8.2: MVR Configuration to Support Non-DCE Execution Mode on page 349).		

Table 6-23 Mover Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
TCP Path Name	The pathname of the MVR TCP/IP listen executable.	Fully qualified file name of the MVR TCP/IP listen executable.	/usr/lpp/ hpss/bin/ hpss_mvr_tcp
	Advice: The TCP MVRs currently supported are listed below. All TCP MVRs support common disk/tape interfaces, TCP/IP, and shared memory data transfers: *The *_omi executables indicate support for the Gresham AdvanTape Device Driver. The *_ssd executables indicate support for the IBM SCSI Tape Device Driver. Check the sources for a complete list of supported devices and platform availability* *Name — Options Supported* *hpss_mvr_tcp — Standard Disk/Tape Devices* *hpss_mvr_ssd — SCSI 3490E/3590/3590E/3590H/3580* *hpss_mvr_omi — SCSI Redwood/Timberline/Eagle* *hpss_mvr_dd2 — Ampex DST-312*		
Encryption Key	An encryption key used to secure the MVR's IOD/ IOR interface.	This value can only be changed by clicking on the **Generate New Key** button.	0
	Advice: A non-zero value must be configured for the security routines to allow any client access to this interface.		

6.8.8.2 MVR Configuration to Support Non-DCE Execution Mode

The Mover can be run in a non-DCE mode, in which case the part of the Mover that handles accessing configuration metadata and servicing the Mover administrative interface (still built on top of DCE and Encina) runs on a DCE/Encina platform (currently AIX or Solaris), while the part of the Mover (to which the PVL and Storage Servers communicate) that manages the storage devices and performs data transfers runs on a remote node that does not require DCE or Encina services. To support this mode of operations, there is some additional configuration which must be done on the remote (non-DCE/Encina) node.

6.8.8.2.1 Add Non-DCE Mover Configuration

Use SSM to add non-DCE mover configuration.

Be sure to consider the 'Advice' sections for the Hostname and Data Hostname fields noted in Table 6-23 when filling in the Non-DCE Mover's specific configuration.

6.8.8.2.2 /etc/services, /etc/inetd.conf, and /etc/xinetd.d

To invoke the non-DCE/Encina part of the mover, the remote nodes' **inetd** is utilized to start the parent process when a connection is made to a port based on the Mover's type specific configuration (see section 6.8.8).

An entry must be added to the **/etc/services** file so that the **inetd** will be listening on the port to which the Mover parent process (running on a DCE/Encina node) will connect. The port number is one greater than that configured for the "TCP Listen Port" in the Mover's type specific configuration. For example, if the configure listen port is 5001, the following line must be added to */etc/services*:

```
hpss_mvr1      5002/tcp
```

For non-Linux systems

A corresponding entry must be added to the **/etc/inetd.conf** file, which will define which executable to run and the arguments to use when a connection is detected. The sole argument (other than the program name) is the pathname to a file that will contain an ASCII representation of the configured encryption key for the Mover (see the next section for further details). For example:

```
hpss_mvr1 stream tcp nowait root /opt/hpss/bin/hpss_mvr_tcp
hpss_mvr_tcp /var/hpss/etc/mvr_ek
```

Which will cause the executable **/opt/hpss/bin/hpss_mvr_tcp** to be run under the **root** user ID when a connection is detected on port 5002. The Mover process will use the **/var/hpss/etc/mvr_ek** file to read the encryption key that will be used to authenticate all connections made to this Mover.

After modifying the **/etc/inetd.conf** file, be sure to refresh the **inetd** daemon using the following commands:

```
% ps -ef | grep inetd
root  6450  3154   0   Apr 29      -  0:02 /usr/sbin/inetd
hpss 17852 59370   2 16:50:25 pts/18  0:00 grep inetd
% kill -s USR2 6450
```

For Linux systems

A corresponding file must be added to the **/etc/xinetd.d** directory, which will define which executable to run and the arguments to use when a connection is detected. The file will be given the same name as the entry in **/etc/services**. Continuing the above example, a file called **/etc/xinetd.d/hpss_mvr1** would be created with the following contents:

```
service hpss_mvr1
{
   disable     = no
   socket_type = stream
   protocol    = tcp
   wait        = yes
```

```
      port        = 5002
      user        = root
      server      = /opt/hpss/bin/hpss_mvr_tcp
      server_args = /var/hpss/etc/mvr_ek
   }
```

The specified port will be one greater than the port listed as the **TCP Listen Port** in the Mover's type specific configuration. For example, the port value in the example corresponds to a Mover with a **TCP Listen Port** value of 5001.

The template will cause the executable **/opt/hpss/bin/hpss_mvr_tcp** to be run under the root user ID when a connection is detected on port 5002. The Mover process will use the **/var/hpss/etc/mvr_ek** file to read the encryption key that will be used to authenticate all connections made to this Mover.

After modifying the file in **/etc/xinetd.d**, be sure to refresh the **xinetd** daemon using the following commands:

```
% /sbin/service xinetd --full-restart
Stopping xinetd:                                       [  OK  ]
Starting xinetd:                                       [  OK  ]
```

6.8.8.2.3 The Mover Encryption Key Files

To authenticate access made to the non-DCE Mover processes, the encryption key configured in this Mover's specific configuration (see section 6.8.8) is read from a file accessible from the local file system. This file contains an ASCII representation of the encryption key (the pathname of the file is passed to the Mover executable as specified in either the **/etc/inetd.conf** or **/etc/xinetd.d** file). For example, if the encryption key in the Mover's type specific configuration is **1234567890ABCDEF**, then the encryption key file (**/var/hpss/etc/ek.mvr1**) should contain:

```
0x12345678 0x90ABCDEF
```

6.8.8.3 System Configuration Parameters for IRIX, Solaris, and Linux

IRIX

When running the non-DCE Mover process on an IRIX platform, there are a number of system configuration parameters which may need to be modified before the Mover can be successfully run. The values can be modified with the *systune* utility (and will likely require the system to be rebooted before they take effect). The following table defines the parameter names and minimum required values.

Table 6-24 IRIX System Parameters

Parameter Name	Minimum Value	Parameter Description
semmsl	512	Maximum number of semaphores per set

Table 6-24 IRIX System Parameters

Parameter Name	Minimum Value	Parameter Description
maxdmasz	513	Maximum DMA size (required for Ampex DST support)

Solaris

When running the Mover or non-DCE Mover process on a Solaris platform, there are a number of system configuration parameters which may need to be modified before the Mover can be successfully run. The values can be modified by editing the */etc/system* configuration file and rebooting the system. The following table defines the parameter names and minimum required values.

Note that the **semmns** and **semmnu** values should be increased if running more than one Mover on the Solaris machine (multiply the minimum value by the number of Movers to be run on that machine).

Table 6-25 Solaris System Parameters

Parameter Name	Minimum Value	Parameter Description
semsys:seminfo_semmns	1024	Maximum number of semaphores in system
semsys:seminfo_semmnu	512	Maximum number of undo structures in system
semsys:seminfo_semmsl	512	Maximum number of semaphores per set
shmsys:shminfo_shmmax	8388608	Maximum shared memory segment size (will allow up to a 2MB Mover buffer size)

Linux

When running the Mover or non-DCE Mover process on a Linux platform, there are a number of system configuration parameters which may need to be modified before the Mover can be successfully run. The values can be modified by editing the Linux kernel source header and then rebuilding the kernel. The following table defines the parameter names and minimum required values.

Note that the **SEMMSL** value should be increased if running more than one Mover on the Linux machine (multiply the minimum value by the number of Movers to be run on that machine).

Table 6-26 Linux System Parameters

Parameter Name	Header File	Minimum Value	Parameter Description
SEMMSL	include/linux/sem.h	512	Maximum number of semaphores per ID
SHMMAX	include/linux/shm.h	0x2000000	Maximum shared memory segment size (bytes)

6.8.8.4 MVR Configuration to Support Local File Transfer

The Mover local file transfer (LFT) protocol allows the Movers to transfer data directly between a Unix file system and HPSS. The requirement for this feature was to allow lower overhead data transfers between global filesystems, not necessarily parallel filesystems, and HPSS.

Caveats:

- This transfer can only occur if both the Mover and the clients have direct access to the Unix filesystem on which the file resides.
- If the multinode features of the Parallel FTP Client are used, the Unix filesystem must be global to all nodes or unexpected results may occur.
- The Mover must be built with the LFT option. This is the default option for all Movers. If not all Movers have been built with this option, clients must explicitly specify a class of service which is valid for a Mover supporting the local file transfer option.
- The Mover must be running as root. If there are other Movers running on the same node, they must also run as root to take advantage of the Mover-to-Mover shared memory data transfer.
- A new configuration file (**`/var/hpss/etc/hpss_mvr_localfilepath.conf`**) is required on all nodes running an enhanced Mover.

  The configuration file contains a list of Unix paths. These paths specify which files are allowed to be moved using this special data transfer protocol. Any Unix file on the Mover node who's path starts with a path in the list of entries is allowed to be transferred using the special data protocol.

  If the configuration file does not exist, then the SSM will issue a minor alarm when the Mover starts, and no Unix files will be transferred with the special protocol.

  Here is an example configuration file:

```
# This file contains Unix paths that can be accessed by the local Movers
# on this node. If the client wishes to make a local file transfer and
# the start of the client path matches any of the paths in this list
# then the transfer will proceed, otherwise the Mover will not transfer
# the file.
#
# The format of this file is simply a list of paths, one per line.
/gpfs
/local/globalfilesystem
```

In the above sample configuration, any file under the path, **`/gpfs`** or the path **`/local/globalfilesystem`** can be transferred using the special data protocol subject to the caveats specified above.

Extreme care should be taken when using this feature. The Mover will move data from any file in its list. If the Unix file system is not global to all Mover nodes, the request will fail.

The following commands have been added to the HPSS Parallel FTP Clients to take advantage of these new features: **`lfget`**, **`lfput`**, **`mlfget`**, **`mlfput`**, **`lfappend`**. See the *HPSS User's Guide* for more details. The hpss_ReadList and hpss_WriteList Client APIs can also be used to utilize the local file transfer capability. See the *HPSS Programmer's Reference, Volume 1* for more details.

6.8.9 Configure the Name Server Specific Information

The NS specific configuration entry can be created using the Name Server Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

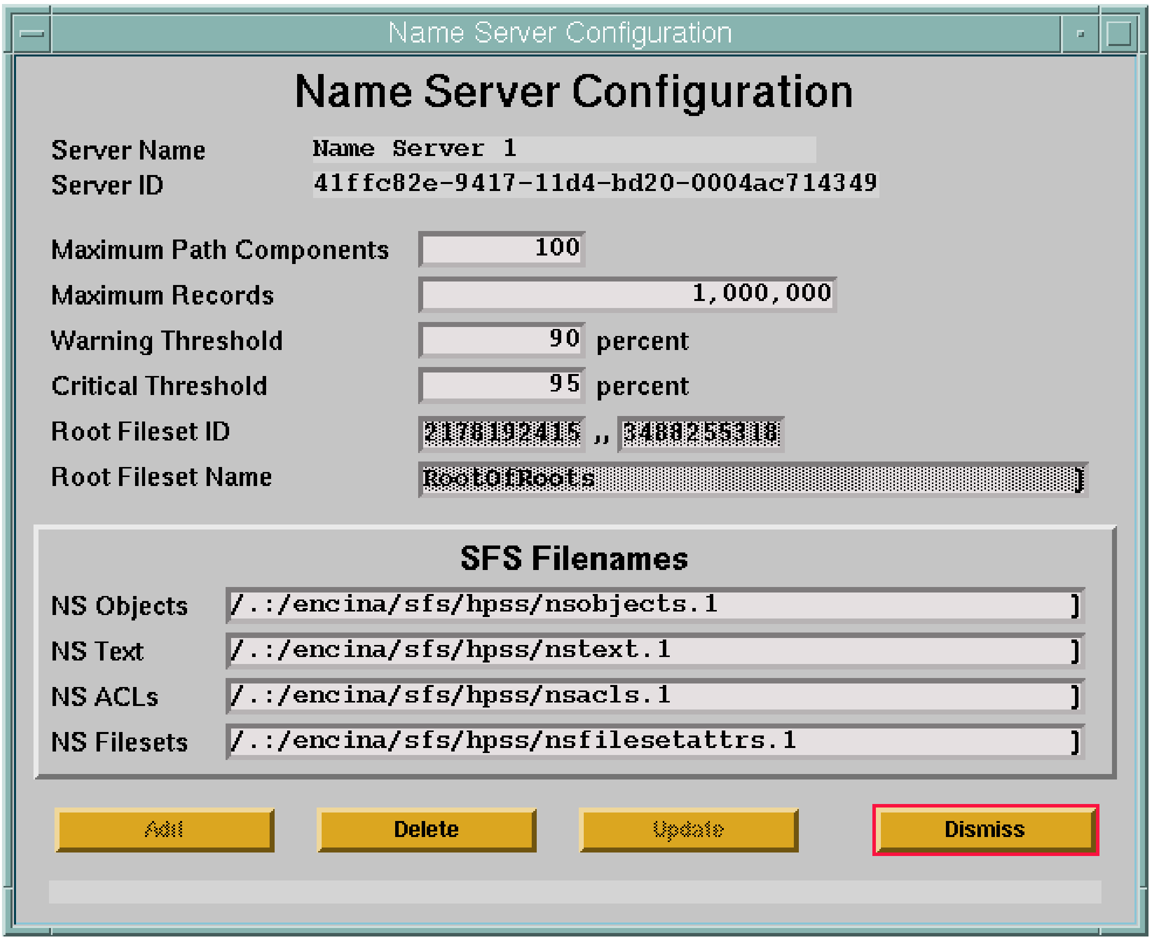
From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 6-5.

To add a new specific configuration, select the Name Server entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The Name Server Configuration window will be displayed as shown in Figure 6-28 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Name Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Name Server Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the Name Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Name Server Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.



Name Server Configuration

Server Name	Name Server 1
Server ID	41ffc82e-9417-11d4-bd20-0004ac714349
Maximum Path Components	100
Maximum Records	1,000,000
Warning Threshold	90 percent
Critical Threshold	95 percent
Root Fileset ID	2178192415 ,, 3488255318
Root Fileset Name	RootOfRoots

SFS Filenames

NS Objects	/.:/encina/sfs/hpss/nsobjects.1
NS Text	/.:/encina/sfs/hpss/nstext.1
NS ACLs	/.:/encina/sfs/hpss/nsacls.1
NS Filesets	/.:/encina/sfs/hpss/nsfilesetattrs.1

Figure 6-28 Name Server Configuration Window

6.8.9.1 *Name Server Configuration Variables*

Table 6-27 lists the fields on the Name Server Configuration window and provides specific recommendations for configuring the NS for use by HPSS.

Table 6-27 Name Server Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the NS. This name is copied over from the NS general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected NS descriptive name
Server ID	The UUID of the NS. This ID is copied over from the NS general configuration entry.	This field cannot be modified. It is displayed for reference only.	Extracted from the NS general configuration entry.
Maximum Path Components	The maximum number of components permitted in a path name.	Any positive integer value between 1 and 512.	100
Maximum Records	The maximum number of SFS records that can be used by the NS to store metadata objects.	Any positive 64-bit integer value.	1,000,000
	Advice: This limit must consider the maximum anticipated number of files and directories in the system, the upper limit for SFS, and the amount of disk space available to SFS. See Section 2.10.2.4 for additional details.		
Warning Threshold	A percentage value indicating when to issue warning messages. When the percentage of used space exceeds this value, the NS will issue warning messages to SSM.	Any integer value between 1 and 100.	90
Critical Threshold	A percentage value indicating when to issue critical messages. When the percentage of used space exceeds this value, the NS will issue critical messages to SSM.	Any integer value between 1 and 100.	95
Root Fileset ID	The fileset ID to be assigned to the Name Server's local root fileset. A fileset ID consists of two numbers (high and low) separated by a double comma.	The highest part must be an unsigned 32-bit number greater than 2147483648. The low part can be any unsigned 32-bit number.	Generated automatically by SSM

Table 6-27 Name Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Root Fileset Name	The name to be assigned to the Name Server's local root fileset. The name must be unique among all filesets within the DCE cell.	A character string of up to 127 bytes in length.	Generated automatically by SSM
SFS Filenames. The fields below list the names of the SFS files used by the NS.			
NS Objects	The path name to the SFS file containing the metadata for the NS objects.	Any valid Encina file name.	/.:/encina/sfs/hpss/nsobjects.#
NS Text	The path name to the SFS file containing the overflow text metadata for the NS objects.	Any valid Encina file name.	/.:/encina/sfs/hpss/nstext.#
NS ACLs	The path name to the SFS file containing the overflow ACL entries for the NS objects.	Any valid Encina file name.	/.:/encina/sfs/hpss/nsacls.#
NS Filesets	The path name to the SFS file containing the metadata for the NS filesets.	Any valid Encina file name.	/.:/encina/sfs/hpss/nsfilesetattrs.#

6.8.10 NFS Daemon Specific Configuration

The NFS Daemon specific configuration entry can be created using the NFS Daemon Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 6-5.

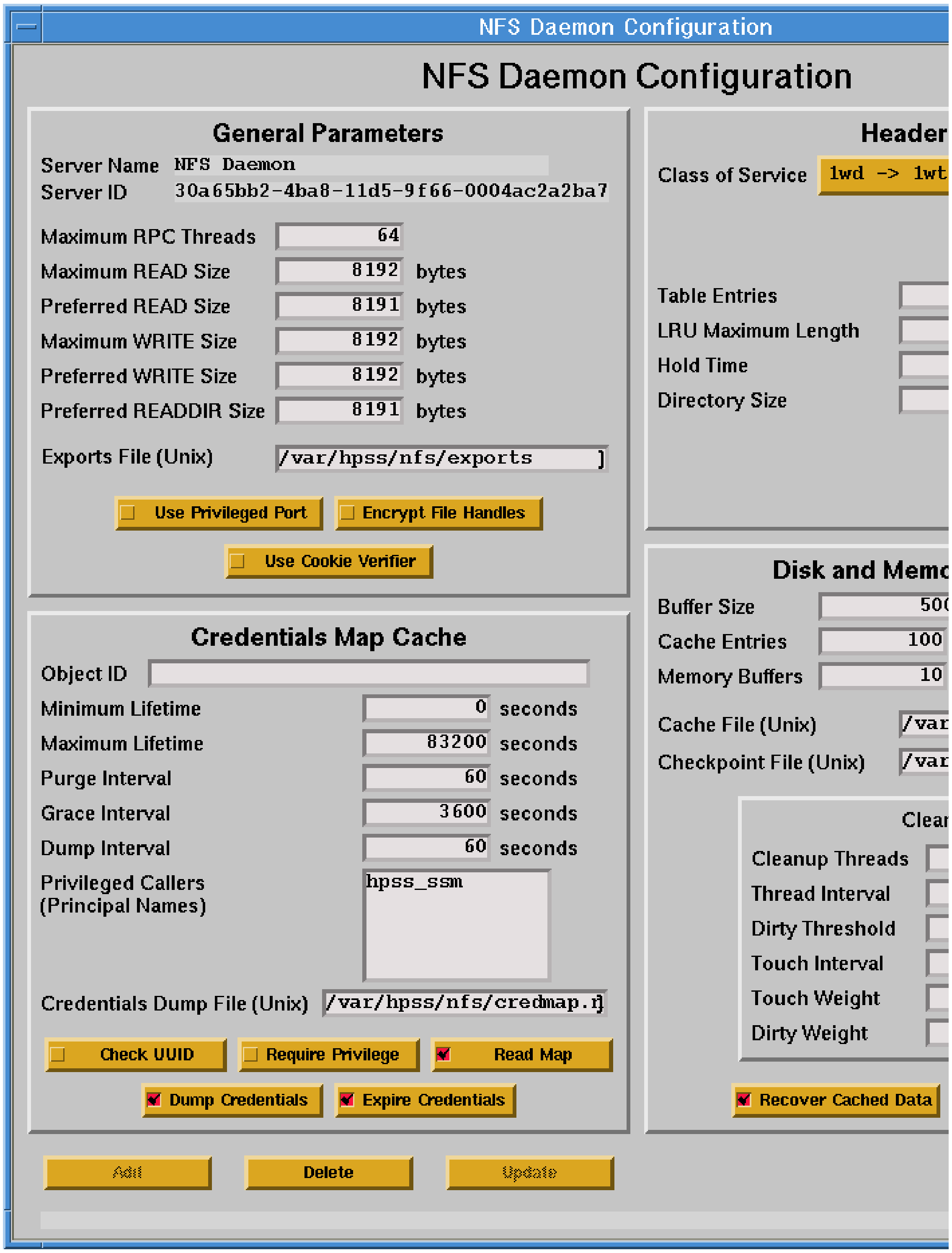
To add a new specific configuration, select the NFS Daemon entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The NFS Daemon Configuration window will be displayed as shown in Figure 6-29 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the NFS Daemon entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The NFS Daemon

Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the NFS Daemon entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The NFS Daemon Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.



NFS Daemon Configuration

NFS Daemon Configuration

General Parameters

Server Name	NFS Daemon	
Server ID	30a65bb2-4ba8-11d5-9f66-0004ac2a2ba7	
Maximum RPC Threads	64	
Maximum READ Size	8192	bytes
Preferred READ Size	8191	bytes
Maximum WRITE Size	8192	bytes
Preferred WRITE Size	8192	bytes
Preferred READDIR Size	8191	bytes
Exports File (Unix)	/var/hpss/nfs/exports	

☐ Use Privileged Port ☐ Encrypt File Handles

☐ Use Cookie Verifier

Credentials Map Cache

Object ID		
Minimum Lifetime	0	seconds
Maximum Lifetime	83200	seconds
Purge Interval	60	seconds
Grace Interval	3600	seconds
Dump Interval	60	seconds
Privileged Callers (Principal Names)	hpss_ssm	
Credentials Dump File (Unix)	/var/hpss/nfs/credmap.n	

☐ Check UUID ☐ Require Privilege [✓ Read Map]

[✓ Dump Credentials] [✓ Expire Credentials]

Header

Class of Service	1wd -> 1wt
Table Entries	
LRU Maximum Length	
Hold Time	
Directory Size	

Disk and Memo

Buffer Size	500
Cache Entries	100
Memory Buffers	10
Cache File (Unix)	/var
Checkpoint File (Unix)	/var

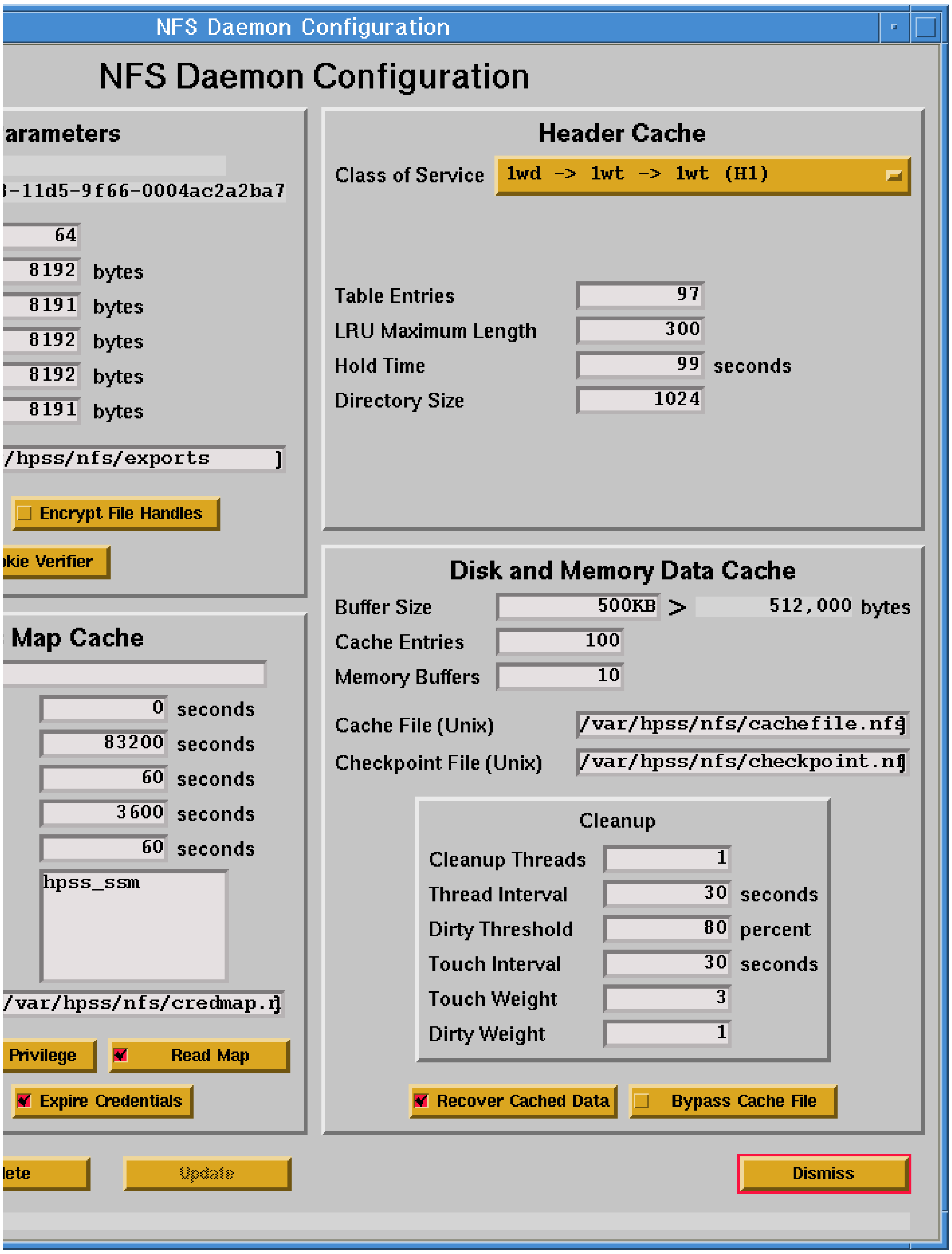
Clea

Cleanup Threads	
Thread Interval	
Dirty Threshold	
Touch Interval	
Touch Weight	
Dirty Weight	

[✓ Recover Cached Data]

Add Delete Update

Figure 6-29 NFS Daemon Configuration Window (left side)



NFS Daemon Configuration

NFS Daemon Configuration

arameters

3-11d5-9f66-0004ac2a2ba7

64	
8192	bytes
8191	bytes
8192	bytes
8192	bytes
8191	bytes

/hpss/nfs/exports

☐ Encrypt File Handles

kie Verifier

Map Cache

0	seconds
83200	seconds
60	seconds
3600	seconds
60	seconds

hpss_ssm

/var/hpss/nfs/credmap.n

Privilege

☑ Read Map

☑ Expire Credentials

Header Cache

Class of Service: 1wd -> 1wt -> 1wt (H1)

Table Entries	97	
LRU Maximum Length	300	
Hold Time	99	seconds
Directory Size	1024	

Disk and Memory Data Cache

Buffer Size	500KB > 512,000	bytes
Cache Entries	100	
Memory Buffers	10	
Cache File (Unix)	/var/hpss/nfs/cachefile.nfs	
Checkpoint File (Unix)	/var/hpss/nfs/checkpoint.nf	

Cleanup

Cleanup Threads	1	
Thread Interval	30	seconds
Dirty Threshold	80	percent
Touch Interval	30	seconds
Touch Weight	3	
Dirty Weight	1	

☑ Recover Cached Data

☐ Bypass Cache File

lete

Update

Dismiss

Figure 6-30 NFS Daemon Configuration window (right side)

6.8.10.1 NFS Daemon Configuration Variables

Table 6-28 lists the fields on the NFS Daemon Configuration window and provides specific recommendations for configuring the NFS Daemon for use by HPSS.

Table 6-28 NFS Daemon Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
General Parameters. The following fields define general information for the NFS Daemon.			
Server Name	The descriptive name of the NFS Daemon. This name is copied over from the NFS Daemon general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected NFS Daemon descriptive name
Server ID	The UUID of the NFS Daemon. This ID is copied over from the NFS Daemon general configuration entry.	The UUID of the NFS Daemon. This field cannot be modified. It is displayed for reference only.	Extracted from the NFS Daemon general configuration entry.
Maximum RPC Threads	The number of NFS requests that can be processed concurrently.	Any positive 32-bit integer value	32
	Advice: Set this field based on the maximum number of concurrent requests expected. In general, it should be at least 8.		
Maximum READ Size	The maixmum size for a read request supported by the server	Any positive 32-bit integer value	8192
Preferred READ Size	The size of read buffer the client should be using to get optimal performance	Any positive 32-bit integer value less than or equal to Maximum READ Size	8192
Maximum WRITE Size	The maximum size for a write request supported by the server	Any positive 32-bit integer value	8192
Preferred WRITE Size	The size of write buffer the client should be using to get optimal performance	Any positive 32-bit integer value less than or equal to Maximum WRITE Size	8192
Preferred READDIR Size	Recommended size for a READDIR request	Any positive 32-bit integer value	8192

Table 6-28 NFS Daemon Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Exports File (Unix)	The name of a UNIX file containing the HPSS directory/file exported for NFS access.	Valid and fully qualified UNIX file name.	/var/hpss/nfs/exports
Use Privileged Port	A flag that indicates whether the NFS clients will use privileged port. If set, the NFS clients must use port numbers less than 1024.	ON, OFF	OFF
Encrypt File Handles	A flag that indicates whether incoming and outgoing file handles will contain an encrypted checksum.	ON, OFF	OFF
Use Cookie Verifier	A flag for NFS V3 that allows the client to manage a readdir request in a smarter way and guarantees that the information form the readdir request is correct.	ON, OFF	OFF
	Note: Using the Cookie Verifier can introduce problems on clients where it is not supported. Therefore, this option should not be used if NFS V2 clients are used.		
Credentials Map Cache. The following fields are used to map the client user credentials to the HPSS user credentials.			
Object ID	The DCE UUID that identifies the credential mapping interface object	Valid DCE UUID.	NULL
Minimum Lifetime	The minimum time, in seconds, for which a credentials mapping is valid.	Any positive 32-bit integer value.	0 seconds
Maximum Lifetime	The maximum time, in seconds, for which a credential mapping is valid.	Any positive 32-bit integer value.	83200 seconds
Purge Interval	The time interval, in seconds, between credentials map purge operations.	Any positive 32-bit integer value.	60 seconds

Table 6-28 NFS Daemon Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Grace Interval	An interval, in seconds, to indicate how long after a credential's last use before it will be expired.	Any positive 32-bit integer value.	3600 seconds
Dump Interval	An interval, in seconds, to determine how often the credentials map cache is checkpointed to a UNIX file.	Any positive 32-bit integer value.	60 seconds
	Advice: This value should be based on how often entries are added/removed from the map. If this value is set too high, the user may not be able to access files after an NFS crash. If it is set too low, it may increase the system overhead. This field is used only if the UID option in the Export file is set.		
Privileged Callers (Principal Names)	The list of principal names of those callers permitted to change entries in the NFS credentials map other than their own.	Valid DCE principal name.	hpss_ssm
Credentials Dump File (Unix)	The name of a UNIX file to checkpoint the credentials map cache.	Valid, fully qualified UNIX file name.	/var/hpss/nfs/credmap.nfs
Check UUID	A flag that indicates whether users not on the privileged callers list may only make entries for their own DCE UID.	ON, OFF	OFF
Require Privilege	A flag that indicates whether only the users on the privileged callers list may change the credentials map.	ON, OFF	OFF
Read Map	A flag that indicates whether the credentials map will be read from a file at startup time.	ON, OFF	ON
	Advice: This flag should be set to OFF when the system is first configured and set to ON when the system is restarted after a crash.		
Dump Credentials	A flag that indicates whether the credentials map will be periodically checkpointed to a UNIX file.	ON, OFF	ON

Table 6-28 NFS Daemon Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Expire Credentials	A flag that indicates whether the credentials map entries will be expired and removed based on the grace and purge intervals.	ON, OFF	ON
Header Cache. The following fields are used to cache names and attributes of the HPSS file objects.			
Class of Service	The name of the COS used when creating the NFS files.	Any configured COS name from the pop-up list.	First configured COS name found in the SFS file.
	Advice: The hierarchy associated with this COS should have disk at its top storage level.		
Table Entries	The number of entries in a hash table used to hold cached attributes for HPSS bitfiles, directories, and symbolic links.	Any positive 32-bit integer value.	97
	Advice: This value should be set to a prime number greater than or equal to the number of HPSS objects used through NFS at any one time.		
LRU Maximum Length	The maximum number of HPSS objects to cache.	Any positive 32-bit integer value.	300
	Advice: This value should be equal to the number of HPSS objects in use through NFS for a period specified by the hold time. Generally, this value should not be greater than three times the number of table entries.		
Hold Time	The interval, in seconds, for which cached attributes remain valid.	Any positive 32-bit integer value.	99
	Advice: Smaller hold times reduce the effectiveness of the cache; larger hold times result in attributes not being updated as often.		
Directory Size	The size of the cache for directory name entries.	Any positive 32-bit integer value.	1024
	*Advice: This value should be based on average file size (fsize) and the average number of files (fnum). The value = 4(44 + fsize) * fnum.*		
Disk and Memory Data Cache. The following fields are used by the NFS Daemon to cache bitfile data.			

Table 6-28 NFS Daemon Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Buffer Size	The size of a single data cache entry.	Any positive 32-bit integer value.	500 KB
	Advice: This value is the amount of data read from and written to HPSS by the data cache layer at a time. Is is recommended that this value be a multiple of 8 KB and, if possible, a power of 2. When multiplied by the number of Memory Buffers, this value is the amount of memory that will be used by the data cache layer. When multiplied by the number of Cache Entries, it represents the amount of disk space used by the Cache file.		
Cache Entries	The number of data cache entries stored on local disk.	Any positive 32-bit integer value.	100
	Advice: When multiplied by the Buffer Size, this value represents the amount of disk used to hold the data cache entries locally in the Cache file.		
Memory Buffers	The number of buffers to allocate in memory.	Any positive 32-bit integer value.	10
	Advice: When multiplied by the Buffer Size, this value represents the RAM used to hold cache entries while they are being worked on. This value must be at least 3.		
Cache File (Unix)	The name of a local UNIX disk file that contains the cached entries.	Any valid, fully qualified UNIX file name.	/var/hpss/nfs/cachefile.nfs
	Advice: The size of this file can be determined by multiplying Buffer Size by Cache Entries		
CheckPoint File (Unix)	The name of a local UNIX disk file that contains the status information for the cached entries stored in the Cache file. Its size is related to the number of the cached entries, but it is much smaller than the Cache file.	Any valid, fully qualified UNIX file name.	/var/hpss/nfs/checkpoint.nfs
Cleanup Threads	The number of threads dedicated to writing dirty cache entries back to HPSS.	Any positive 32-bit integer value.	1
	Advice: This value should be small because extra, temporary cleanup threads are spawned as needed. Each non-temporary cleanup thread wakes up every Thread Interval seconds to look for dirty entries.		

Table 6-28 NFS Daemon Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Thread Interval	An interval, in seconds, that specifies how often the cleanup threads wake up to look for dirty entries.	Any positive 32-bit integer value.	30 seconds
	Advice: This value should not be too small because extra cleanup threads are spawned as needed when the cache starts to fill up.		
Dirty Threshold	The percentage of disk cache entries that must be dirty before extra cleanup threads are spawned.	Any integer value between 1 and 100.	80 percent
Touch Interval	The interval, in seconds, that a disk cache entry must not have been accessed in order for it to be considered a candidate to be written back to HPSS by a cleanup thread.	Any positive 32-bit integer value.	30 seconds
	Advice: This value is ignored when the Dirty Threshold is exceeded. It is recommended that this value be greater than the number of seconds the NFS client cache holds on to dirty data.		
Touch Weight	The importance of writing back to HPSS a dirty entry that has not been accessed too recently.	Any positive 32-bit integer value.	3
	Advice: The Touch Weight should be higher than the Dirty Weight to avoid locking out access to cached data that has not been flushed to HPSS. A three to one ratio is recommended (3 for Touch Weight, 1 for Dirty Weight).		
Dirty Weight	The importance of writing back to HPSS a dirty entry that has not been dirty for a long period of time.	Any positive 32-bit integer value.	1
	Advice: The Dirty Weight should be lower than the Touch Weight to avoid locking out access to cached data that has not been flushed to HPSS. A three to one ratio is recommended (3 for Touch Weight, 1 for Dirty Weight).		

Table 6-28 NFS Daemon Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Recover Cached Data	A flag that indicates whether the data cache layer will be forced to look for dirty cache entries in the CheckPoint File. When ON, the data cache layer looks for dirty entries. When OFF, it ignores them and resets the checkpoint file.	ON, OFF	ON
	Advice: This flag should be turned OFF temporarily only when there is a problem with the recovery process that inhibits the NFS Daemon from starting up because any dirty entries will be lost.		
Bypass Cache File	A flag that indicates whether the Cache file will be used. When ON, the Cache file will not be used. All transfers will go directly to HPSS.	ON, OFF	OFF

6.8.11 Non-DCE Client Gateway Specific Configuration

The Non-DCE Client Gateway specific configuration entry can be created using the Non-DCE Client Gateway Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

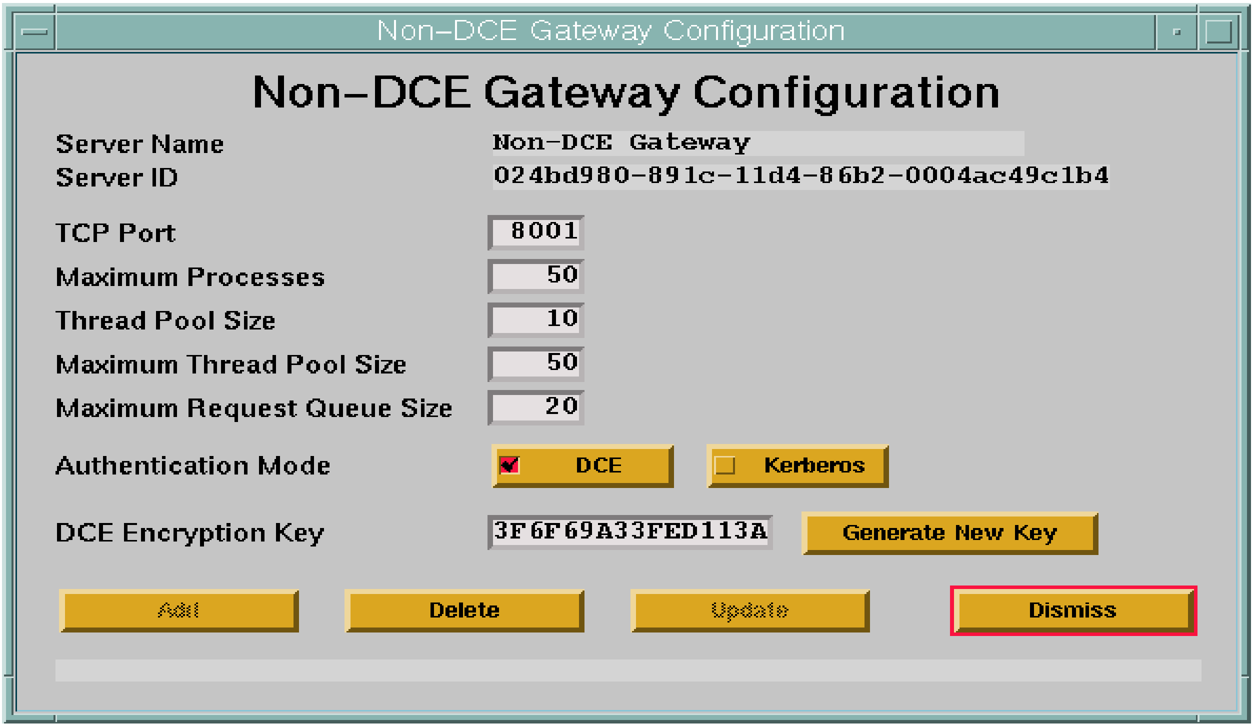
From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 6-5.

To add a new specific configuration, select the Non-DCE Client Gateway entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The Non-DCE Client Gateway Configuration window will be displayed as shown in Figure 6-31 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Non-DCE Client Gateway entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Non-DCE Client Gateway Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the Non-DCE Client Gateway entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Non-DCE Client Gateway Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.



Non-DCE Gateway Configuration

Server Name: Non-DCE Gateway
Server ID: 024bd980-891c-11d4-86b2-0004ac49c1b4
TCP Port: 8001
Maximum Processes: 50
Thread Pool Size: 10
Maximum Thread Pool Size: 50
Maximum Request Queue Size: 20
Authentication Mode: [✓ DCE] Kerberos
DCE Encryption Key: 3F6F69A33FED113A

Figure 6-31 Non-DCE Client Gateway Configuration Window

6.8.11.1 *Non-DCE Client Gateway Configuration Variables*

Table 6-29 lists the fields on the Non-DCE Client Gateway Configuration window and provides specific recommendations for configuring the Non-DCE Client Gateway for use by HPSS.

Table 6-29 Non-DCE Client Gateway Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the Non-DCE Client Gateway. This name is copied over from the Non-DCE Client Gateway general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected Non-DCE Client Gateway descriptive name.

Table 6-29 Non-DCE Client Gateway Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server ID	The UUID of the Non-DCE Client Gateway. This ID is copied over from the Non-DCE Client Gateway general configuration entry.	The UUID of the Non-DCE Client Gateway. This field cannot be modified. It is displayed for reference only.	Assigned by SSM when the Non-DCE Client Gateway general configuration entry is created.
TCP Port	Default port number to be used by the Non-DCE Client Gateway's listener process.	This number must be unique among all Non-DCE Client Gateways running on a single host	8001
Maximum Processes	The maximum number of request processes that can be active at any point in time. Effectively the maximum number of clients that can be connected to the Non-DCE Client Gateway at a time.	Any positive 32-bit integer value.	50
Thread Pool Size	The size of the request thread pool for each request process. If the number of simultaneous requests equals this number, the thread pool will be expanded up to its maximum defined value to handle additional requests. When request activity is then reduced, the size of the thread pool with be reduced back to this value.	Any positive 32-bit integer value.	10
Maximum Thread Pool Size	The maximum number of request threads per process active at any one time. If the number of simultaneous requests equals this number, any additional requests will be queued for processing by the next available request thread.	Any positive 32-bit integer value. This value must be at least as large as the value given for Thread Pool Size.	50

Table 6-29 Non-DCE Client Gateway Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Maximum Request Queue Size	The maximum number of requests to queue until a request thread becomes available. If this queue fills, no more requests will be processed for this particular Non-DCE client until there is more room in the queue.	Any positive 32-bit integer value.	20
Authentication Mode	These are the NDAPI client authentication mechanisms supported by the gateway.	DCE and/or Kerberos or None	DCE
	Advice: The NDAPI Client Library provides a means for a client to set the type of authentication mechanism it wishes to use. By default the NDAPI supports the DCE and None mechanisms (NDAPI Client Library and Gateway can be rebuilt to also support Kerberos authentication). During initial connection the client may request authentication using one of supported mechanism. If the gateway supports this mechanism and the client provides the correct authentication material, then the client is granted access to HPSS as the presented identity. Care should be take when setting this field, if neither DCE or Kerberos is specified, then NDAPI client will not be required to authenticate themselves.		
DCE Encryption Key	A key used to encrypt the DCE user name and password passed between the NDAPI client and gateway during the initial connection.	This value can only be changed by clicking on the *Generate New Key* button.	0
	Advice: This key should always be set if using DCE authentication. The value of this key should be placed in the NDAPI Client Library encryption key file (see Section 7.2: Non-DCE Client API Configuration on page 415 for details on configuring the NDAPI Client Library).		

6.8.12 Configure the PVL Specific Information

The PVL specific configuration entry can be created using the PVL Server Configuration window. After the configuration entry is created, it can be viewed, updated or deleted through the same window.

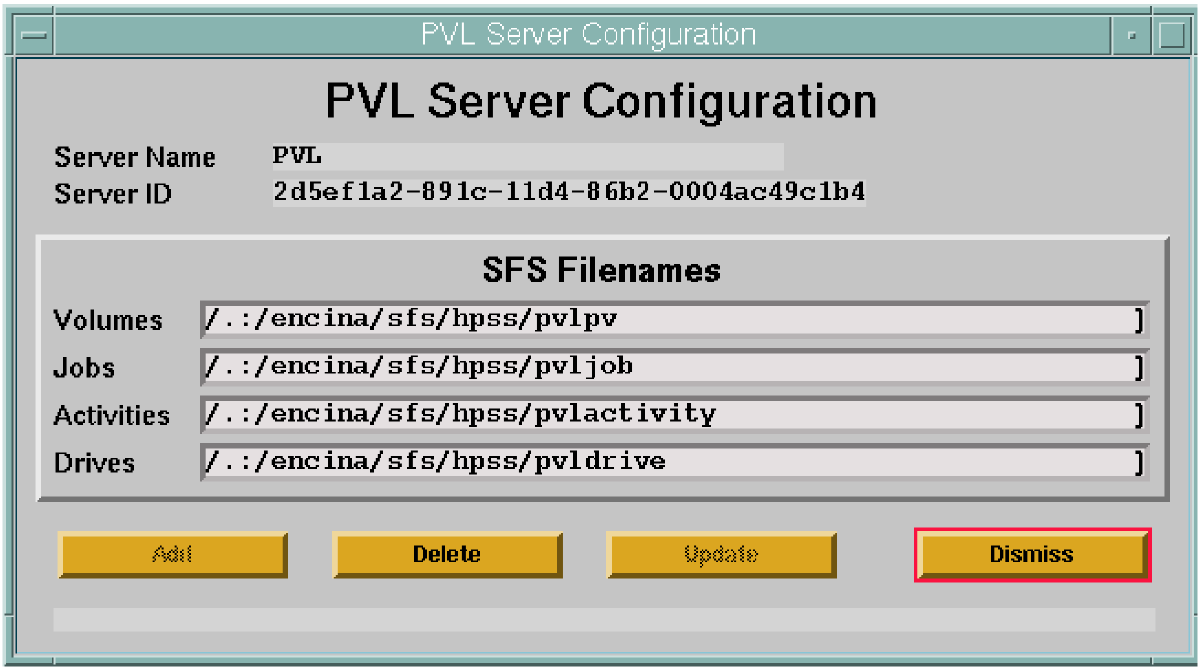
From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 6-5.

To add a new specific configuration, select the PVL Server entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The PVL Server Configuration window will be displayed as shown in Figure 6-32 with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the PVL Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The PVL Server Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the PVL Server entry on the HPSS Servers window and click on the **Delete** button from the **Configuration** button group. The PVL Server Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.



Server Name PVL
Server ID 2d5ef1a2-891c-11d4-86b2-0004ac49c1b4

Volumes /.:/encina/sfs/hpss/pvlpv
Jobs /.:/encina/sfs/hpss/pvljob
Activities /.:/encina/sfs/hpss/pvlactivity
Drives /.:/encina/sfs/hpss/pvldrive

Figure 6-32 PVL Server Configuration Window

6.8.12.1 Physical Volume Library Configuration Variables

Table 6-30 lists the fields on the PVL Server Configuration window and provides specific recommendations for configuring the PVL for use by HPSS.

Table 6-30 Physical Volume Library Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the PVL. This name is copied over from the PVL general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected PVL server descriptive name.
Server ID	The UUID of the PVL. This ID is copied over from the PVL general configuration entry.	The UUID of the PVL. This field cannot be modified. It is displayed for reference only.	Extracted from the PVL general server configuration entry.
SFS Filenames. The fields below list the names of the SFS files used by the PVL Server.			
Volumes	The name of the Encina SFS file that maintains volume metadata for the PVL.	Valid Encina file name.	/.:/encina/sfs/hpss/pvlpv
	Advice: Use a name that is meaningful to the type of metadata being stored.		
Jobs	The name of the Encina SFS file that maintains job metadata for the PVL.	Valid Encina file name.	/.:/encina/sfs/hpss/pvljob
	Advice: Use a name that is meaningful to the type of metadata being stored.		
Activities	The name of the Encina SFS file that maintains activity metadata for the PVL.	Valid Encina file name.	/.:/encina/sfs/hpss/pvlactivity
	Advice: Use a name that is meaningful to the type of metadata being stored.		
Drives	The name of the Encina SFS file that maintains drive metadata for the PVL.	Valid Encina file name.	/.:/encina/sfs/hpss/pvldrive
	Advice: Use a name that is meaningful to the type of metadata being stored.		

6.8.13 Configure the PVR Specific Information

The PVR specific configuration entry can be created using the PVR Server Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

If you are configuring a PVR for StorageTek, IBM 3494/3495/3584, or ADIC AML; before proceeding with PVR configuration you should read the PVR-specific section (Section 6.8.13.4: StorageTek PVR and RAIT PVR Information on page 389,Section 6.8.13.3: IBM 3494/3495 PVR Information on page 388, Section 6.8.13.2: LTO PVR Information on page 387, and Section 6.8.13.5: ADIC Automatic Media Library Storage Systems Information on page 392). These sections provide additional vendor-specific advice on PVR/robot configuration.

The STK RAIT PVR cannot be supported at this time since STK has not yet made RAIT generally available.

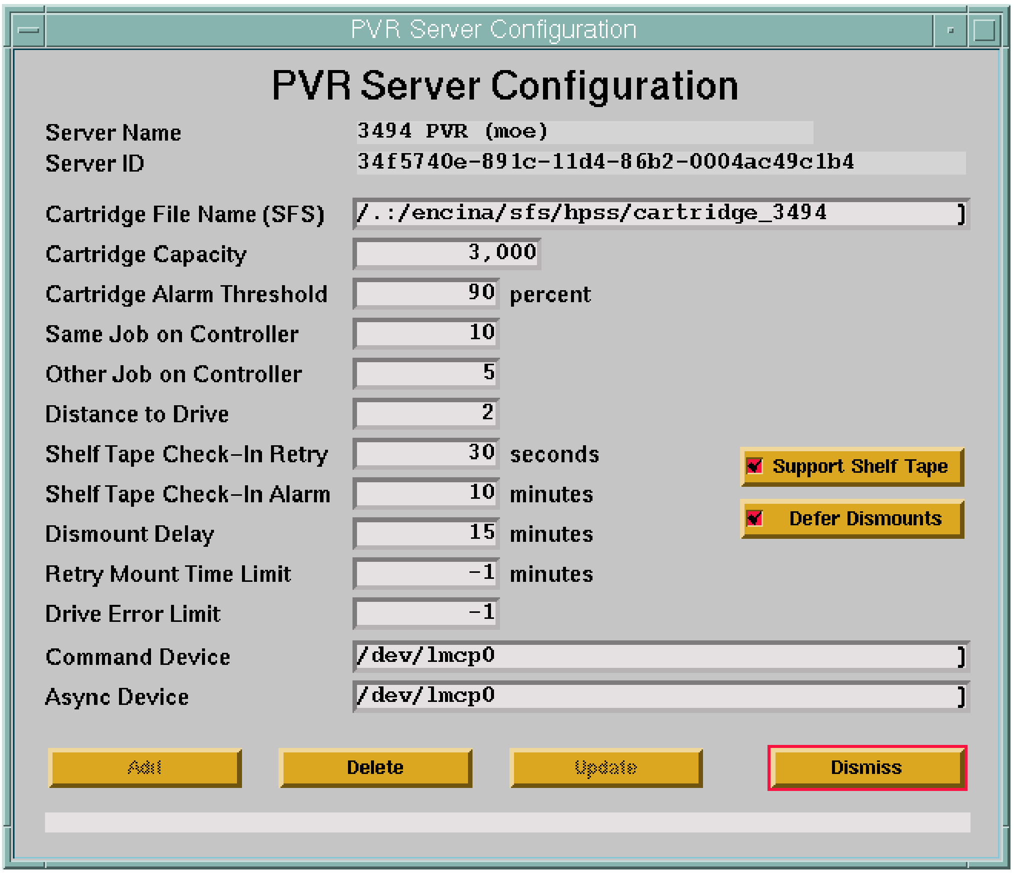
From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 6-5.

To add a new specific configuration, select the PVR Server entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The PVR Server Configuration window will be displayed (as shown in figures 6-33 through 6-39) depending on the selected PVR type) with default values. If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the PVR Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The PVR Server Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing configuration, select the PVR Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The PVR Server Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

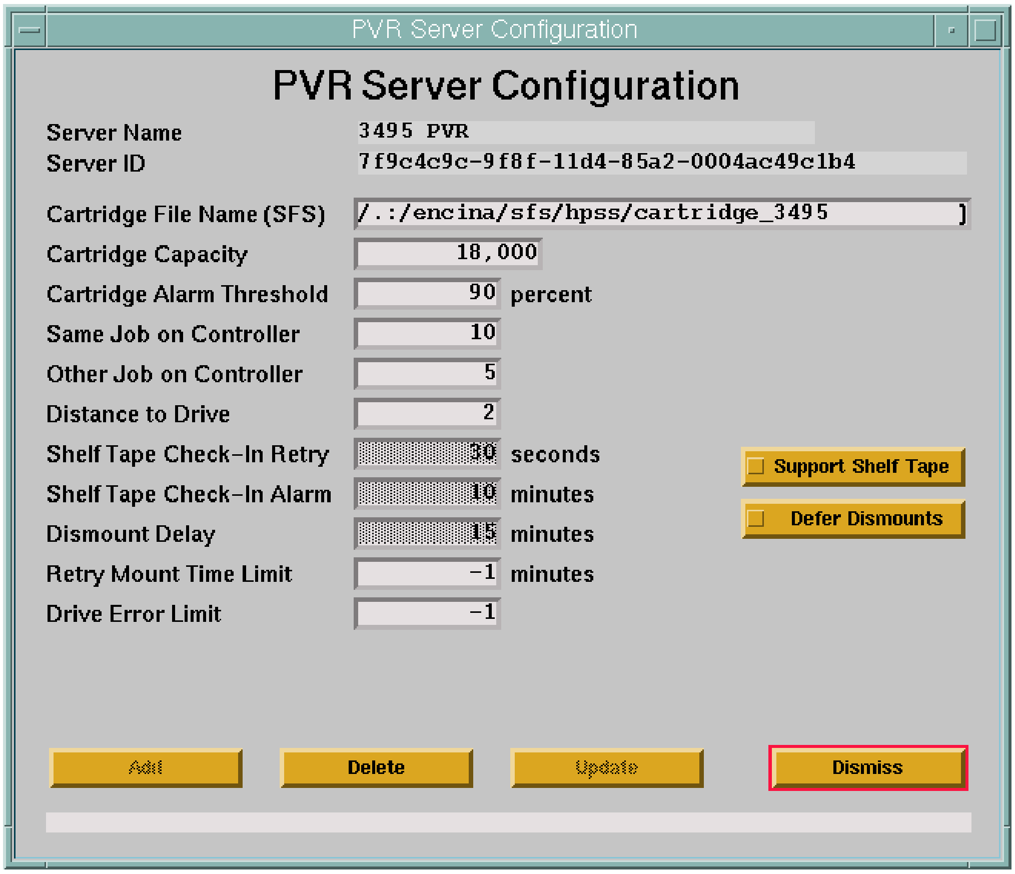


PVR Server Configuration

Server Name	3494 PVR (moe)
Server ID	34f5740e-891c-11d4-86b2-0004ac49c1b4
Cartridge File Name (SFS)	/.:/encina/sfs/hpss/cartridge_3494
Cartridge Capacity	3,000
Cartridge Alarm Threshold	90 percent
Same Job on Controller	10
Other Job on Controller	5
Distance to Drive	2
Shelf Tape Check-In Retry	30 seconds
Shelf Tape Check-In Alarm	10 minutes
Dismount Delay	15 minutes
Retry Mount Time Limit	-1 minutes
Drive Error Limit	-1
Command Device	/dev/lmcp0
Async Device	/dev/lmcp0

[✓ Support Shelf Tape]
[✓ Defer Dismounts]

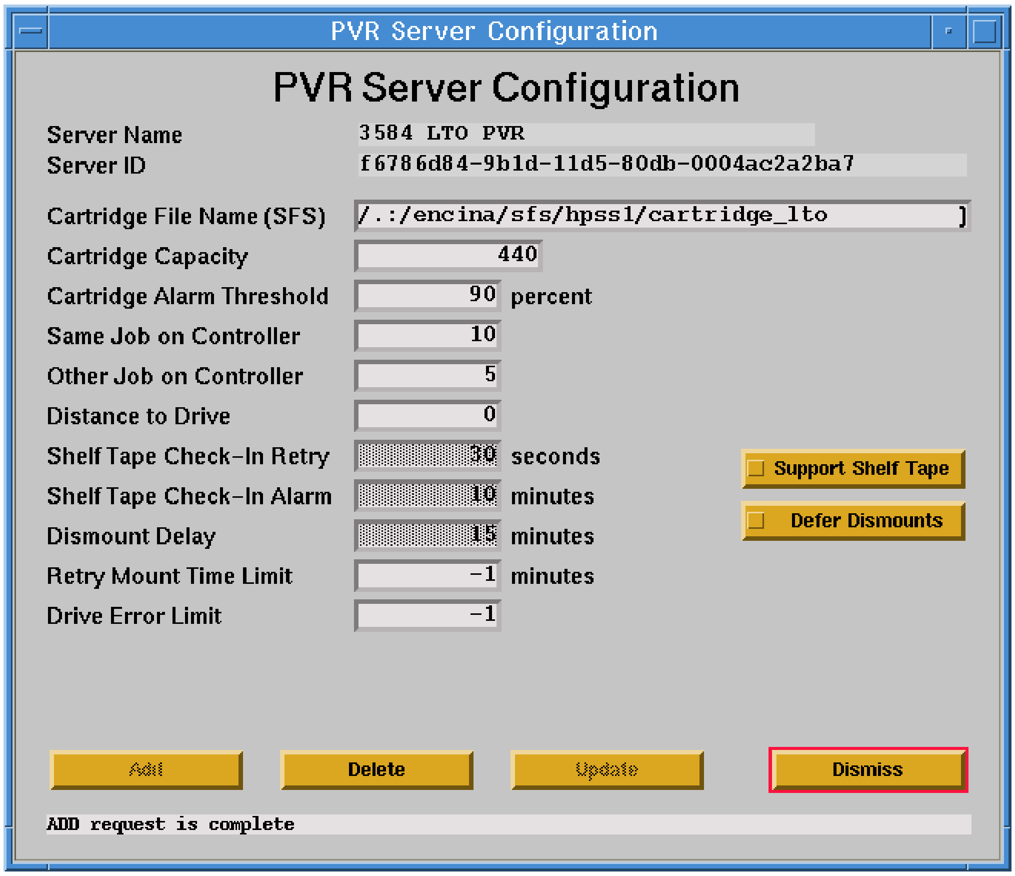
Figure 6-33 3494 PVR Server Configuration Window



PVR Server Configuration

Server Name	3495 PVR
Server ID	7f9c4c9c-9f8f-11d4-85a2-0004ac49c1b4
Cartridge File Name (SFS)	/.:/encina/sfs/hpss/cartridge_3495
Cartridge Capacity	18,000
Cartridge Alarm Threshold	90 percent
Same Job on Controller	10
Other Job on Controller	5
Distance to Drive	2
Shelf Tape Check-In Retry	30 seconds
Shelf Tape Check-In Alarm	10 minutes
Dismount Delay	15 minutes
Retry Mount Time Limit	-1 minutes
Drive Error Limit	-1

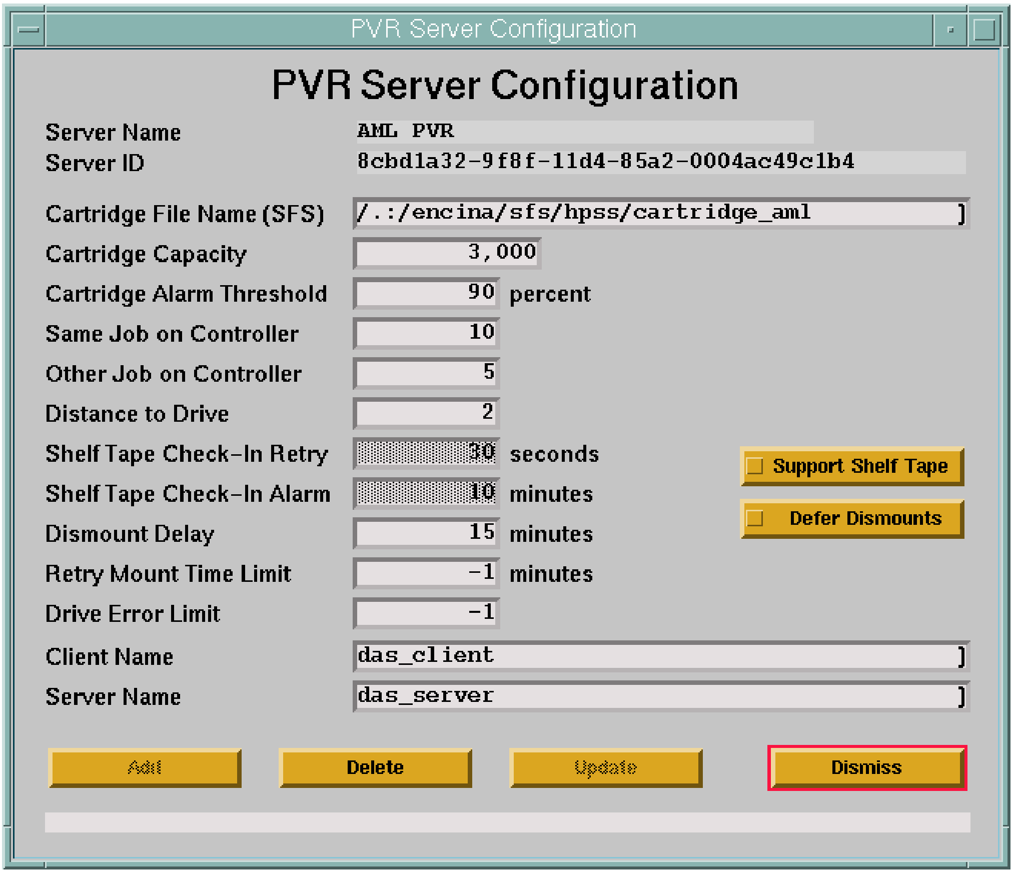
Figure 6-34 3495 PVR Server Configuration Window



Server Name	3584 LTO PVR
Server ID	f6786d84-9b1d-11d5-80db-0004ac2a2ba7
Cartridge File Name (SFS)	/.:/encina/sfs/hpss1/cartridge_lto
Cartridge Capacity	440
Cartridge Alarm Threshold	90 percent
Same Job on Controller	10
Other Job on Controller	5
Distance to Drive	0
Shelf Tape Check-In Retry	30 seconds
Shelf Tape Check-In Alarm	10 minutes
Dismount Delay	15 minutes
Retry Mount Time Limit	-1 minutes
Drive Error Limit	-1

```
ADD request is complete
```

Figure 6-35 3584 LTO PVR Server Configuration Window



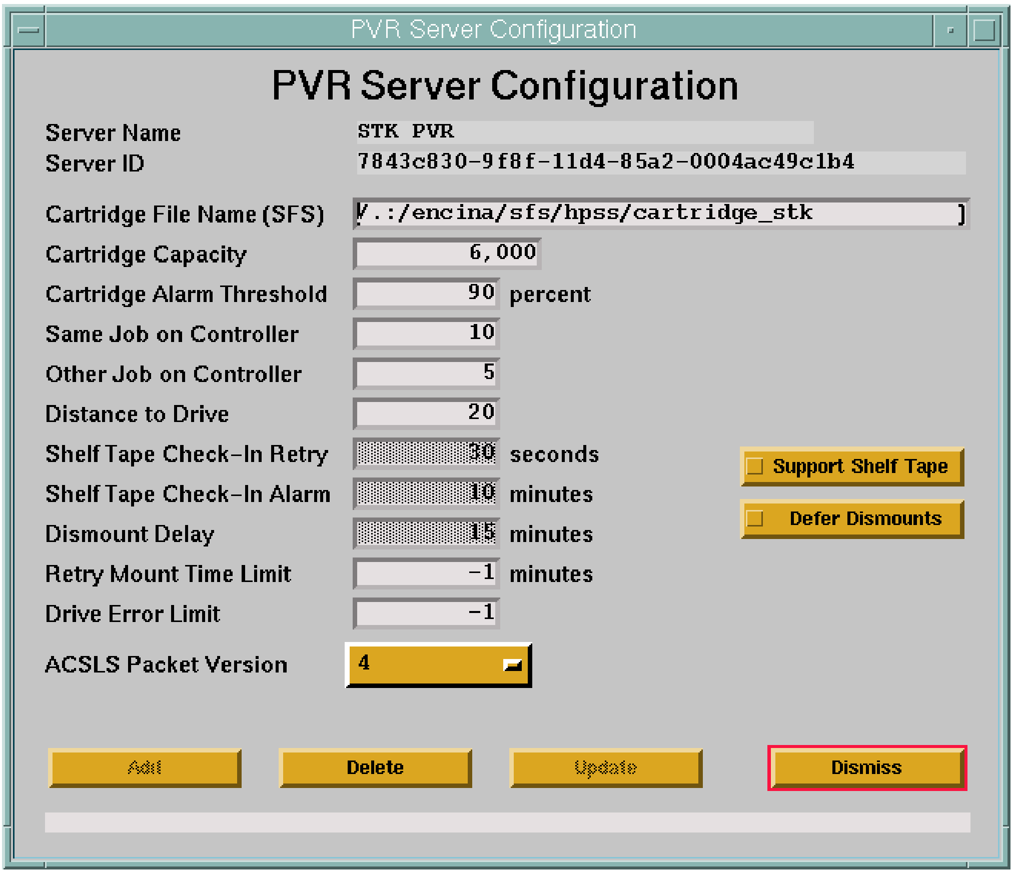
PVR Server Configuration

Server Name	AML PVR	
Server ID	8cbd1a32-9f8f-11d4-85a2-0004ac49c1b4	
Cartridge File Name (SFS)	/.:/encina/sfs/hpss/cartridge_aml	
Cartridge Capacity	3,000	
Cartridge Alarm Threshold	90	percent
Same Job on Controller	10	
Other Job on Controller	5	
Distance to Drive	2	
Shelf Tape Check-In Retry	30	seconds
Shelf Tape Check-In Alarm	10	minutes
Dismount Delay	15	minutes
Retry Mount Time Limit	-1	minutes
Drive Error Limit	-1	
Client Name	das_client	
Server Name	das_server	

☐ Support Shelf Tape

☐ Defer Dismounts

Figure 6-36 AML PVR Server Configuration Window

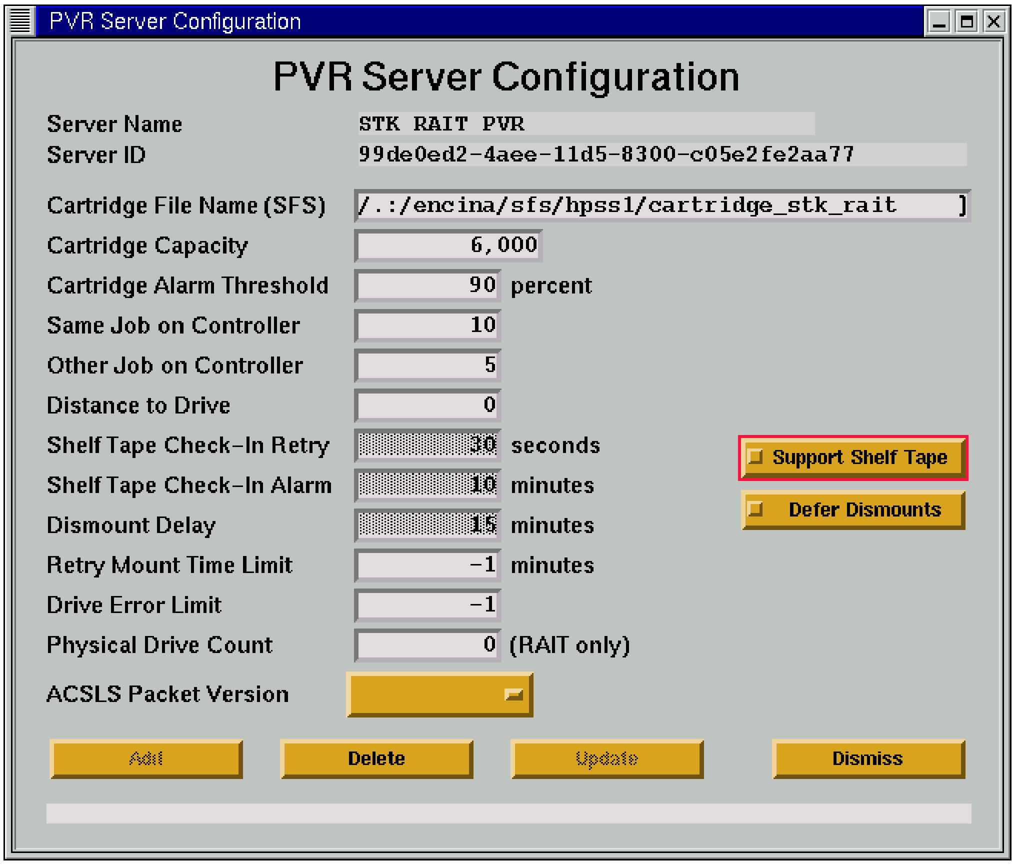


PVR Server Configuration

Server Name	STK PVR
Server ID	7843c830-9f8f-11d4-85a2-0004ac49c1b4
Cartridge File Name (SFS)	/.:/encina/sfs/hpss/cartridge_stk
Cartridge Capacity	6,000
Cartridge Alarm Threshold	90 percent
Same Job on Controller	10
Other Job on Controller	5
Distance to Drive	20
Shelf Tape Check-In Retry	30 seconds
Shelf Tape Check-In Alarm	10 minutes
Dismount Delay	15 minutes
Retry Mount Time Limit	-1 minutes
Drive Error Limit	-1
ACSLS Packet Version	4

☐ Support Shelf Tape
☐ Defer Dismounts

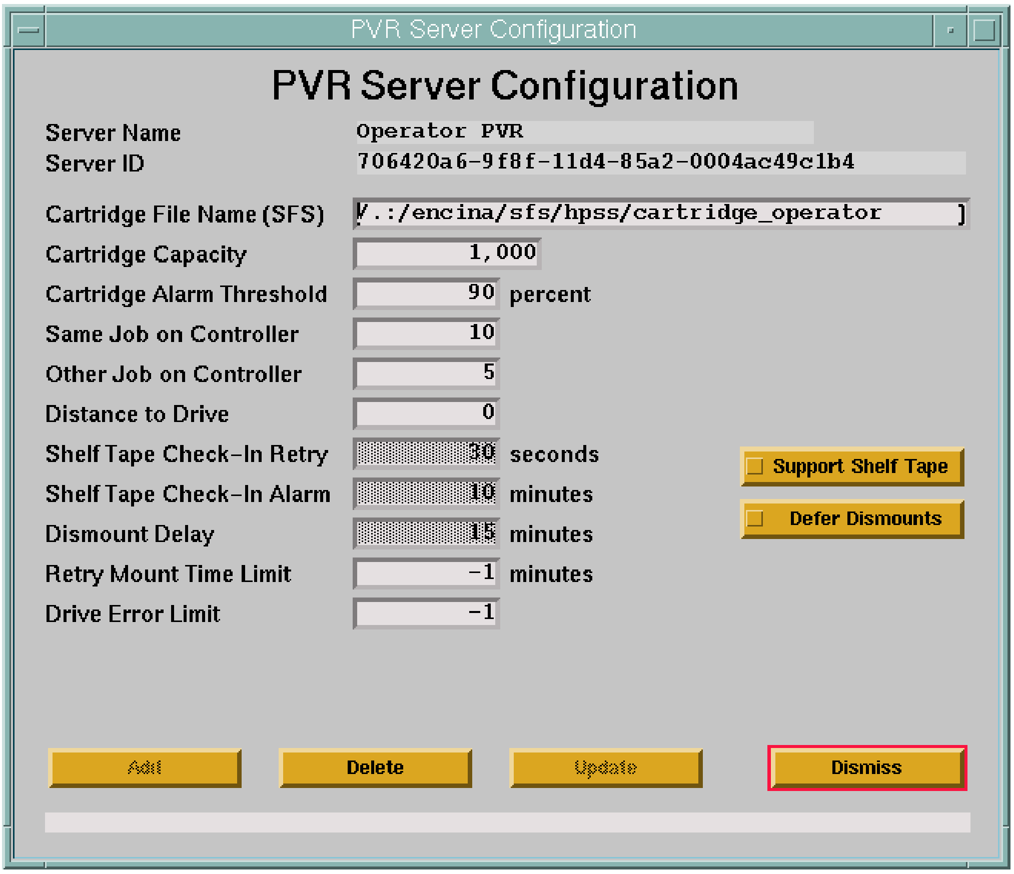
Figure 6-37 STK PVR Server Configuration Window



PVR Server Configuration

Server Name	STK RAIT PVR
Server ID	99de0ed2-4aee-11d5-8300-c05e2fe2aa77
Cartridge File Name (SFS)	/.:/encina/sfs/hpss1/cartridge_stk_rait
Cartridge Capacity	6,000
Cartridge Alarm Threshold	90 percent
Same Job on Controller	10
Other Job on Controller	5
Distance to Drive	0
Shelf Tape Check-In Retry	30 seconds
Shelf Tape Check-In Alarm	10 minutes
Dismount Delay	15 minutes
Retry Mount Time Limit	-1 minutes
Drive Error Limit	-1
Physical Drive Count	0 (RAIT only)
ACSLS Packet Version	

Figure 6-38 STK RAIT PVR Server Configuration Window



PVR Server Configuration

Server Name	Operator PVR	
Server ID	706420a6-9f8f-11d4-85a2-0004ac49c1b4	
Cartridge File Name (SFS)	/.:/encina/sfs/hpss/cartridge_operator	
Cartridge Capacity	1,000	
Cartridge Alarm Threshold	90	percent
Same Job on Controller	10	
Other Job on Controller	5	
Distance to Drive	0	
Shelf Tape Check-In Retry	30	seconds
Shelf Tape Check-In Alarm	10	minutes
Dismount Delay	15	minutes
Retry Mount Time Limit	-1	minutes
Drive Error Limit	-1	

Figure 6-39 Operator PVR Server Configuration Window

6.8.13.1 Physical Volume Repository Configuration Variables

Table 6-31 lists the fields on the PVR Server Configuration window and provides specific recommendations for configuring a PVR for use by HPSS.

Table 6-31 Physical Volume Repository Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the PVR. This name is copied over from the PVR general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected PVR server descriptive name.

Table 6-31 Physical Volume Repository Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Server ID	The UUID of this PVR. This ID is copied over from the PVR general configuration entry.	The UUID of this PVR. This field cannot be modified. It is displayed for reference only.	Extracted from the PVR general server configuration entry.
Cartridge File Name (SFS)	The name of the Encina file that maintains PVR cartridge metadata.	Valid Encina file name.	/.:/encina/sfs/ hpss/cartridge _<robot>
	Advice: The cartridge file name for each PVR must be unique. Multiple PVRs cannot share the same SFS file for cartridges. Suggested names are: *cartridge_3494 for IBM 3494 PVRs* *cartridge_aml for ADIC AML PVRs* *cartridge_lto for IBM 3584 LTO PVRs* *cartridge_stk for StorageTek-based PVRs* *cartridge_rait for StorageTek RAIT PVRs* *ccartridge_operator for Operator PVRs* *If two robots of the same type are managed by two different PVRs, be sure that each one has a different SFS file for cartridges.*		
Cartridge Capacity	The total number of HPSS cartridges to be stored in this PVR. This is not a hard limit, but rather the point at which major alarms will be generated.	A positive 32-bit integer.	3495 - 18000 3494 - 3000 3584 - 440 STK - 6000 RAIT - 6000 Operator - 1000
Cartridge Alarm Threshold	The maximum percentage of cartridge capacity allowed in a PVR before a minor alarm is generated.	Any integer value between 1 and 100.	90
	Advice: Depending on operational policy, in some sites it is wise to leave a small number of free slots in a PVR to allow room for a small number of cartridge injections without the need for corresponding ejects.		

Table 6-31 Physical Volume Repository Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Same Job On Controller	The number of cartridges from this job mounted on this drive's controller. The larger the number, the harder the PVR will try to avoid mounting two tapes in the same stripe set on drives attached to the same controller. See Advice below for more information.	Any positive 32-bit integer value.	10
Other Job On Controller	The number of cartridges from other jobs mounted on this drive's controller. The larger the number, the harder the PVR will try to avoid mounting any two tapes on drives attached to the same controller. See Advice below for more information.	Any positive 32-bit integer value.	5
Distance to Drive	The number of units of distance from the cartridge to the drive. The larger the number, the harder the PVR will try to mount tapes on the closest drive to the tape's current location. See Advice below for more information.	Any positive 32-bit integer value.	3495 - 2 3494 - 2 3584 - 0 STK - 20 RAIT - 0 Operator - 0

Table 6-31 Physical Volume Repository Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
*Advice: The Same Job on Controller, Other Job on Controller, and Distance to Drive values are used by the PVR when selecting a drive for a tape mount operation. The three values are essentially weights that are used to compute an overall score for each possible drive. After the score has been calculated, the drive with the lowest score is selected for the mount. If two or more drives tie for the lowest score, one drive is selected at random. The score is calculated as follows: Score = Weight 1 * Cartridges from this job mounted on this drive's controller + Weight 2 * Cartridges from other jobs mounted on this drive's controller + Weight 3 * Units of distance from the cartridge to the drive This method has the effect of distributing a striped tape mount across as many controllers as possible for the best performance. It also will try to pick controllers that are currently driving a minimum number of tapes. So, in an environment with many tape drives per controller, the best performance will be achieved by minimizing the load on any one controller.*			
Shelf Tape Check-In Retry	The number of seconds the PVR will wait before asking the robot if a requested shelf tape has been checked-in. The PVR will continue checking at this interval until the tape is checked-in. This field is only active if the Support Shelf Tape button is checked	Any positive 32-bit integer value	30 seconds
Shelf Tape Check-In Alarm	The PVR will periodically log alarm messages when a requested shelf tape has not been checked-in. This field specifies the number of minutes between alarms. This field is only active if the Support Shelf Tape button is checked.	Any positive 32-bit integer value	10 minutes
Dismount Delay	The number of minutes that dismounts are delayed after the last data access. This field is only active if the Defer Dismounts button is checked.	Any positive 32-bit integer value	15 minutes

Table 6-31 Physical Volume Repository Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Support Shelf Tape	A toggle button. If ON, the PVR will support the removal of cartridges from the tape library. If OFF, the PVR will not support the removal of cartridges from the tape library.	ON, OFF	OFF
Defer Dismounts	A flag to determine whether all tape dismounts in the drives manages by the PVR will be delayed when the associated PVL jobs are completed. If ON, the PVL will delay the dismounting of a tape cartridge until the drive is required by another job or until an approximate 15 minutes time limit is exceeded.	ON, OFF	OFF
Retry Mount Time Limit	This field controls the mount behavior of the PVR. If an error is encountered during a PVR mount operation, the mount will pend and be retried every 5 minutes. Setting a value in the field will change the mount behavior to periodically retry the mount until the specified time limit is exceeded. Once exceeded, an error is generated and the mount request is cancelled. If the mount request would have resulted in a write operation, the error returned will cause the Storage Server to set the Volume to EOM. Once at EOM, the volume will no longer be available for write operations.	-1 or 0 : Maintain persistent mount behavior, 1 - 3600 : Retry mount for this many minutes before failing	-1

Table 6-31 Physical Volume Repository Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Drive Error Limit	This field is intended to be used in conjunction with the PVR Server "Retry Mount Time Limit". If the number of consecutive mount errors which occur to any drive in this PVR equal or exceed this value, the drive is automatically locked by the PVL. This field is employed by the PVL, changing its value will not change drive disable behavior until the PVL is recycled.	-1 or 0 : Don't automatically disable the drive, 1 - 999999 : Automatically disable the drive after this number of consecutive errors is exceeded.	
Physical Drive Count	The number of physical drives associated with the PVR's RAIT drive pool. This value is read by the PVL on startup. Alterations to the value this field will not take effect until the PVL is recycled.	Any positive 32-bit integer value.	0
ACSLS Packet Version (STK Only)	The packet version used by STK's ACSLS software. See the *STK Automated Cartridge System Library Software (ACSLS) System Administrator's Guide* for details. The environment variable **ACSAPI _PACKET _VERSION** will override the value entered in this field. If neither are set, a default value of 4 is used.	4	4
	Advice: ACSLS 5.0 generally uses packet version 4.		

Table 6-31 Physical Volume Repository Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Command Device (3494/3495 Only)	The name of the device that the PVR can use to send commands to the 3494/3495 robot. The environment variable **HPSS_3494_COMMAND_DEVICE** will override the value entered in this field.	Any device; generally /dev/lmcpX for AIX systems; symbolic library name defined in **/etc/ibmatl.conf** for Solaris systems	/dev/lmcp0
Async Device (3494/3495 Only)	The name of the device that the PVR can use to receive replies from the 3494/3495 robot. The environment variable **HPSS_3494_ASYNC_DEVICE** will override the value entered in this field.	Any device; generally /dev/lmcpX for AIX systems; symbolic library name defined in **/etc/ibmatl.conf** for Solaris systems	/dev/lmcp0
	Advice: For Block Multiplexer Channel (BMUX)-attached IBM robots, the Async Device must be different from the Command Device. For TTY and LAN-attached robots, the devices can be the same.		
Client Name	The name of the client requesting authorization from the Distributed Automated Media Library Server.	Any alphanumeric string of length <= 64	none
	Additional Info: DAS software, which executes on the OS/2 controller PC, allows different clients to control the AML robotics system. DAS uses the client name to determine the access permission of the requesting client to the AML's storage positions, drives, and Insert/Eject units. Access configurations for clients are set in the configuration file C:\DAS\ETC\CONFIG on the OS/2 PC. The client name can be up to 64 alphanumeric characters in length and is case sensitive.		
Server Name	TCP/IP host name or IP address of the AML OS/2-PC DAS server.	Any alphanumeric string of length <= 64	none
	Additional Info: This value must be defined in the network domain server and must be resolvable during DAS start. The server name is set in the configuration file C:\CONFIG.SYS on the OS/2 PC. The server name can be up to 64 alphanumeric characters long and can include up to six dots ('.').		

6.8.13.2 LTO PVR Information

6.8.13.2.1 Vendor Software requirements

HPSS is designed to work with the AIX tape driver (Atape) software to talk to the IBM 3584 LTO Library over a SCSI channel. Currently HPSS is only supported for the AIX version of the Atape driver. Please note that the PVR must run on the same node that has the Atape interface and this node must have a direct SCSI connection to the library.

Please refer to *The 3584 UltraScalable Tape Library Planning and Operator Guide* and *IBM Ultrium Device Drivers Installation and User's Guide* for more information

6.8.13.2.2 Configuration Requirements

When Configuring an HPSS PVR to manage an LTO library, the **Drive Address** configuration entries (in the Devices and Drives configuration) should match the SCSI address (location) of the drive in the library. Running an inventory, either through the "**tapeutil**" utility that comes with the Atape driver or directly on the library console should give you this information. Typically the first drive is located at location 257 for a single frame library.

The LTO PVR must run on the same node that has the Atape interface and this node must have a direct SCSI connection to the library. Please note that the control channel for the library is shared with the data path for the first drive in the library over the same SCSI interface.

The LTO PVR requires an SMC device special file to access the robot. For AIX systems, this file is usually named **/dev/smc0** and can be created using the System Management Interface Tool (SMIT) or using **cfgmgr**.

The 3584 Library can be partitioned into multiple logical libraries which can each be controlled by a separate PVR. This is useful if you have a large library, since you can allocate a set of tapes to a specific set of drives under a logical library.

While HPSS can theoretically share an IBM robot with other tape management systems, this is not recommended. If a robot is shared, care must be taken to ensure that a drive is not used by any other tape management system while that drive is configured as unlocked in HPSS. This is important because HPSS periodically polls all of its unlocked drives even if they are not currently mounted. The SMC device special file is not available to other tape management programs. Additional device special files may be created for other programs.

6.8.13.2.3 Cartridge Import and Export

When importing new cartridges into HPSS, the cartridges must be entered into the 3584 Import/Export (I/E) station before any HPSS import operations re performed. The HPSS LTO PVR moves the cartridges into an available slot while importing into HPSS. When an HPSS tape export command is issued, the cartridges will be placed in the I/E ports and can then be manually removed. Tapes that are entered into the library without an HPSS import or tapes that are not removed after an HPSS export should not affect HPSS in any way, HPSS will simply continue to function as if these tapes did not exist in the library.

Cleaning cartridges are managed by the LTO library directly and should not be imported into the HPSS system.

6.8.13.2.4 Vendor Information

1. *3584 UltraScalable Tape Library Planning and Operator Guide* GA32-0408-01

2. *IBM Ultrium Device Drivers Installation and User's Guide* GA32-0430-00.1

3. *IBM Ultrium Device Drivers Programming Reference* WB1304-01

4. *3580 Ultrium Tape Drive Setup, Operator and Service Guide* GA32-0415-00

5. *3584 UltraScalable Tape Library SCSI Reference* WB1108-00

6.8.13.3 IBM 3494/3495 PVR Information

6.8.13.3.1 Vendor Software Requirements

HPSS is designed to work with IBM 3494 / 3495 robots attached to an HPSS server with either RS-232, Ethernet, or Block Multiplexer (BMUX) control connections. Data paths to the drives will be SCSI-2 with RS-232 and Ethernet control paths, or BMUX if BMUX control paths are used. The HPSS PVR must run on a machine with the appropriate version of Library Manager Control Point (LMCP) device drivers installed.

6.8.13.3.2 Configuration Requirements

When configuring an HPSS PVR to manage an IBM robot, the Drive Address configuration entries correspond to the device number of the drive. Determine the device number by running the HPSS tool **GetESANumbers** for each tape drive in the robot.

The HPSS PVR requires one or two LMCP device special files to access the robot. For AIX systems, these files are usually named **/dev/lmcpX** and can be created using the System Management Interface Tool (SMIT). For a BMUX connected robot, the files can be defined to be a *Command Port* or an *Asynchronous (Async) Port*. The HPSS PVR requires both. By default, the PVR will expect the Command and Async ports to be named **/dev/lmcp0** and **/dev/lmcp1** respectively. Control connections must be made prior to configuration of the **/dev/lmcpx** devices or undefined errors may result. For Solaris systems, the symbolic library name defined in **/etc/ibmatl.conf** (e.g., **3494a**) should be used.

For RS-232 and Ethernet connected robots, the device special files support both command and async capabilities. If only one device special file is created, the environment variables or configuration should be set so that both the command and async ports point to that one special file. It is also possible to create two device special files and arbitrarily select one to be the command port and the other to be the async port. The PVR can then be configured to recognize the ports as described above in the BMUX case.

HPSS can share an IBM robot with other tape management systems. If a robot is shared, care must be taken to make sure that a drive is not used by any other tape management system while that drive is configured as unlocked in the HPSS PVL. This is important because HPSS periodically polls all of its unlocked drives even if they are not currently mounted. The two LMCP device special files (or possibly one file in the case of an RS-232 or Ethernet controlled robot) are not available to other tape management programs. Additional device special files may be created for other programs.

Other programs can send commands to the robot at the same time as HPSS through the additional device special file.

If the robot is placed in *pause mode* by an operator, an alarm will appear on the HPSS operator screen. All subsequent robot operations will silently be suspended until the robot is put back in automatic mode.

6.8.13.3.3 Cartridge Import and Export

When importing new cartridges into HPSS, the cartridges must be entered into the IBM robot before any HPSS import operations are performed. Cartridges placed in the convenience I/O port will automatically be imported by the robot.

When ejecting cartridges, HPSS will send cartridges to the convenience I/O port if it is available. If the port is not available and a high capacity I/O region is defined, the cartridges will be placed in that region. If no locations are available for the cartridge to be ejected, an alarm will appear on the HPSS operator screen and HPSS will retry the eject operation periodically.

6.8.13.3.4 Vendor Information

1. *IBM 3494 Tape Library Dataserver Operator's Guide*, GA32-0280-02
2. Parallel and ESCON Channel Tape Attachment/6000 Installation and User's Guide, GA32-0311-02
3. IBM SCSI Device Drivers: Installation and User's Guide, GC35-0154-01
4. IBM 3495 Operator's Guide, GA32-0235-02
5. Parallel and ESCON Channel Tape Attachment/6000 Installation and User's Guide, GA32-0311-02

6.8.13.4 StorageTek PVR and RAIT PVR Information

6.8.13.4.1 Vendor Software Requirements

HPSS is designed to work with Storage Technology Corporation's (STK's) Automated Cartridge System Library Software (ACSLS) Version 5.0.

ACSLS should be running on the workstation directly connected to the STK Silo. The HPSS STK PVR or HPSS RAIT STK PVR can run on any workstation that has a TCP/IP connection to the ACSLS workstation. The workstation running the HPSS PVR must also be running STK's Storage Server Interface (SSI) software. This software will not be started by HPSS and should be running when HPSS is started. It is recommended that the SSI be started by the workstation's initialization scripts every time the workstation is booted.

The STK RAIT PVR cannot be supported at this time since STK has not yet made RAIT generally available.

The SSI requires that the system environment variables **CSI_HOSTNAME** and **ACSAPI_PACKET_VERSION** be correctly set. Note that due to limitations in the STK Developer's Toolkit, if the SSI is not running when the HPSS PVR is started, or if the SSI crashes while the HPSS PVR is running, the HPSS PVR will lock up and will have to be manually terminated by issuing a **kill -9** command.

6.8.13.4.2 Configuration Requirements

When configuring HPSS to manage an STK Silo, the Drive Address configuration entries correspond to the ACS, Unit, Panel, Drive number used by ACSLS to identify drives. For example, the first drive in a typical Silo configuration has the Drive Address (**0,0,10,0**).

HPSS can share an STK robot with other tape management systems. If a robot is shared, care must be taken to make sure that a drive is not used by any other tape management system while that drive is configured as unlocked in HPSS. This is important because HPSS can be configured to periodically poll all of its unlocked drives even if they are not currently mounted or in use by HPSS. If a drive is being used by another tape management system, it must be configured as locked in HPSS.

HPSS periodically polls all drives in its pool, so these drives cannot be used by other tape management software when HPSS is running even if HPSS is not currently mounting tapes to those drives.

HPSS will use any Cartridge Access Port (CAP) in the STK Silo that has a priority greater than zero. When a CAP is needed by HPSS, HPSS will pick the highest priority CAP that is currently available. At least one CAP must be assigned a non-zero priority. See the *STK Automated Cartridge System Library Software (ACSLS) System Administrator’s Guide* for procedures to set CAP priority.

6.8.13.4.3 Cartridge Import and Export

When importing new cartridges into HPSS, the cartridges must be entered into the STK Silo before any HPSS import operations are performed. See the *STK Automated Cartridge System Library Software (ACSLS) System Administrator’s Guide* for procedures to enter cartridges. When exporting HPSS cartridges, the cartridges will be placed in the CAP for removal.

Cleaning cartridges are managed by the ACSLS software and should not be imported into the HPSS system.

STK supports cartridges without bar code external labels. These cartridges can be entered into the Silo using the **venter** command at the ACSLS console. These cartridges are fully supported by HPSS, but the Silo will eject the cartridges if they are scanned during an audit.

6.8.13.4.4 Starting the STK Client Processes

In order for the HPSS PVR to communicate with the STK robot, it is required that STK client side processes be running on the node where the HPSS PVR is executing. These client side processes are the Server System Interface (ssi) and the Toolkit event logger. These binaries and associated script files are distributed with the HPSS, but are maintained by the STK Corporation.

The binaries and script files for starting the STK client side processes are located in the **$HPSS_PATH/stk/bin** directory. Documentation files describing the files in the bin directory are located in the **$HPSS_PATH/stk/doc** directory. Refer to these doc files for additional information.

The **t_startit.sh** file must be executed to start the STK client processes on those nodes where the HPSS PVR is executing. The script will prompt for the following options:

1. Multi-Host or Single-Host - specify m for multiple host.

2. Server side or Client side - Specify c for client side.

3. Remote Host Name - specify the name of the host where the ACSLS software is running.

4. Remote Host Version - specify one of the following (probably 4).

```
ACSLS Release Number    Remote Host Version Number
            5.x                        4
```

5. Whether processes created should be listed (Y or N) - specify either y or n.

6. Whether **t_cdriver** should be started - specify n.

To terminate the STK client processes, the script **t_killit.sh** may be executed.

Sample output from t_startit.sh follows:

```
 ***** Welcome to t_startit.sh, Version 2.01 *****
This is the automated tester and startit script for the TOOLKIT. Simply
answer the questions which follow.
Executables live in: /opt/hpss/stk/bin
Would you like Multi-Host or Single-Host testing?
Enter one of the following followed by ENTER:
   M Multi-host testing
   S Single-host testing
   X eXit this script
Enter choice: m
Would you like to define the server side or client side for Multi-Host
testing?
Enter one of the following followed by ENTER:
   S Server side
   C Client side
Enter choice: c
The Remote Host Name is the name of the server which has the ACSLS
software (or simulator) running on it.
Enter Remote Host Name (CSI_HOSTNAME): jeep
The Remote Host Version number is the ACSLS Packet Version level which
the server is expecting.
 Here are the valid choices:
ACSLS Release Number Remote      Host Version Number
```

```
                5.x                           4

Enter Remote Host Version (ACSAPI_PACKET_VERSION): 4
Starting /opt/hpss/stk/bin/mini_el... Attempting startup of /opt/hpss/
bin/mini_el ...
Starting /opt/hpss/bin/ssi... Attempting startup of PARENT for /opt/
hpss/bin/ssi... SIGHUP received Parent Process ID is: 17290 Attempting
startup of /opt/hpss/bin/ssi... SIGHUP received Parent Process #17290
EXITING NORMALLY
Initialization Done.
Do you want to see the processes created? (Y or N): y
17288 p4 S 0:00 /opt/hpss/stk/bin/mini_el
17292 p4 S 0:00 /opt/hpss/stk/bin/ssi 17290 50004 23
17295 p4 S 0:00 grep /opt/hpss/stk/bin
Do you want to start up t_cdriver? (Y or N): n
```

6.8.13.4.5 Vendor Information

1. *STK Automated Cartridge System Library Software (ACSLS) System Administrator's Guide*, PN 16716

2. STK Automated Cartridge System Library Software Programmer's Guide, PN 16718

6.8.13.5 ADIC Automatic Media Library Storage Systems Information

6.8.13.5.1 Vendor Software Requirements

HPSS is designed to work with ADIC Distributed Automated Media Library Server (DAS) software version 1.3 and the ABBA Management Unit (AMU) version 2.4.0. DAS is the ADIC software which consists of the Automated Media Library (AML) Client Interface (ACI) and the DAS server components. The AMU is the host computer software by which the ADIC Storage System manages the archive database, which is based on a DB/2 compatible database for an OS/2 system.

The AMU must run on a OS/2 PC host computer connected to the AML robot while the HPSS AML PVR can run on any RS/6000 workstation that has a TCP/IP connection to the OS/2 host computer. The workstation running the HPSS AML PVR must also contain the DAS/ACI software that is called by the HPSS AML PVR.

Refer to ADIC *DAS Installation and Administration Guide* and *Reference Guide AMU* for additional information.

6.8.13.5.2 Configuration Requirements

HPSS can share an AML robot with other tape management systems. If a robot is shared, care must be taken to make sure that a drive is not used by any other tape management system while that drive is configured as unlocked in HPSS. This is important because HPSS can be configured to periodically poll all of its unlocked drives even if they are not currently mounted or in use by HPSS. If a drive is being used by another tape management system, it must be configured as locked in HPSS. For robots that have more than one arm (such as the AML/2), users should configure the PVL Drive Information/Controller ID of each drive depending which arm is servicing it.

User needs to set the Server Name and Client Name, which are case sensitive, in the AML PVR Server Configuration panel to establish the connectivity between the HPSS software and the OS/2 controlling the robot. The Server Name is the name of the controller associated with the TCP/IP address, as defined in the TCP/IP HOST file, and the Client Name is the name of the OS/2 administrator client as defined in the DAS configuration.

Additionally users must configure the Insert/Eject ports for the AML PVR using the configuration files **/var/hpss/etc/AML_EjectPort.conf** and **/var/hpss/etc/AML_InsertPort.conf**. The reasons are that the AML robot can have multiple insert and eject ports, which have the capability to handle different media types. These two configuration files in conjunction with the AMU AMS configuration files specify which Insert/Eject areas or bins the tapes should be placed in for insertion into the archive and where they are ejected to when the HPSS export command is used.

Note that if multiple E/I/F bins are used for exporting cartridges, users must set the environment variable **DAS_EJECTAREAFULL=1** on the DAS Server PC by doing the following:

1. Edit the **config.sys**
2. Add a new line:

```
set DAS_EJECTAREAFULL=1
```

6.8.13.5.3 Cartridge Import and Export

When importing new cartridges into HPSS, the cartridges must be entered into the AML Insert/Eject/Foreign (I/E/F) port before any HPSS import operations are performed. The HPSS AML PVR moves the cartridges into the towers or shelves before importing into HPSS. Users can then issue the HPSS cartridge import command to import the tapes into HPSS; users do not need to issue the **dasadmin insert** command prior to importing. See the ADIC *AML Administrator's Guide* for procedures to enter cartridges into the I/E/F bins. When an HPSS tape export command is issued, the cartridges will be placed in the I/E/F ports and must be removed before issuing an HPSS import command.

Users must configure the Insert/Eject ports for the AML PVR using the configuration files **/var/hpss/etc/AML_EjectPort.conf** and **/var/hpss/etc/AML_InsertPort.conf**. The AML robot can have multiple insert and eject ports, which have the capability to handle different media types. These two configuration files in conjunction with the AMU AMS configuration files specify which Insert/Eject areas or bins the tapes should be placed in for insertion into the archive and where they are ejected to when the HPSS export command is used.

Cleaning cartridges are managed by the DAS software and should not be imported into the HPSS system.

6.8.13.5.4 Starting the DAS Server Processes

In order for the HPSS AML PVR to communicate with the AML robot, it is required that the DAS Server process be running.

Once the DAS, TCP/IP and AML configuration have been completed, the DAS server is initialized and started with the following steps:

1. Make sure the AMU archive management software is running and the hostname is resolved,

2. Select an OS/2 window from the Desktop and change the directory to C:\DAS,

   ```
   C:> cd \das
   ```

3. At the prompt, type **tcpstart** and make sure that TCP/IP gets configured and that the port mapper program is started,

   ```
   C:\das> tcpstart
   ```

4. Type **dasstart** to start the DAS server

   ```
   C:\das> dasstart
   ```

In most cases, a startup file like the following can be used to automatically start the DAS processor:

```
call tcpstart
das\tools\os2sleep 20
CD \AMU
START CON
START KRN
cd \das
tools\os2sleep 20
call dasstart
cd bin
start “DAS/2 AmuClient”
exit
```

6.8.13.5.5 Vendor Information

1. Administration Guide Distribution AML Server, Order no. DOC F00 010-B

2. Reference Guide AMU, Order no. DOC E00 005

3. Interfacing Guide DAS, Order no. DOC F00 011

6.8.14 Configure the Storage Server Specific Information

The Storage Server specific configuration entry can be created using the Storage Server Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Servers** option. The HPSS Servers window will be displayed as shown in Figure 6-5.

To add a new specific configuration, select the Storage Server entry and click on the **Type-specific...** button from the **Configuration** button group on the HPSS Servers window. The Storage Server Configuration window will be displayed with default values (Figure 6-40 and Figure 6-41 shown the screen image of the Disk and Tape Storage Server configuration window, respectively). If the default data is not desired, change the fields with the desired values. Click on the **Add** button to create the configuration entry.

To update an existing configuration, select the Storage Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Storage Server Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

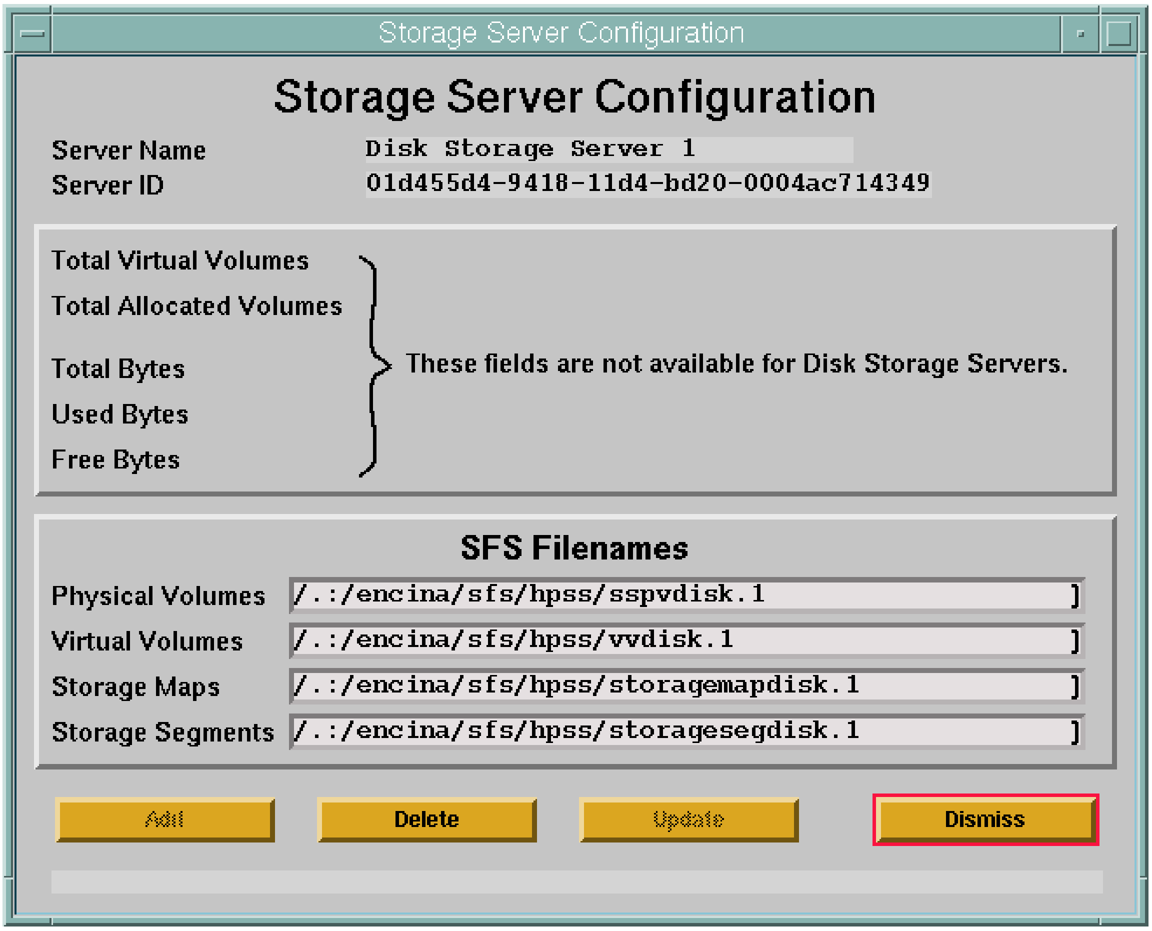
To delete an existing configuration, select the Storage Server entry on the HPSS Servers window and click on the **Type-specific...** button from the **Configuration** button group. The Storage Server Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the specific configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.

When configuring a Disk Storage Server, the statistics fields (Total Virtual Volumes, Total Allocated Volumes, Total Bytes, Used Bytes, and Free Bytes) in the Storage Server Configuration window will not be displayed. These fields cannot be set for Disk Storage Servers because the server computes these values each time it initializes and does not use the fields in the specific configuration record.

When configuring a Tape Storage Server, the initial values of the statistics fields are normally zero, but other values can be used. The server reads the configuration record when it initializes and updates the record as the statistics change. For this reason, SSM will not allow changes to be made to the Storage Server specific configuration while the server is running. Adjustments can only be made to the statistics by modifying the statistics fields when the server is not running.

The best way to set the initial values of the tape statistics is to use the **settapestats** utility program. After all of the tapes have been created in a tape storage server, take the server down and run **settapestats** with the **update_both** option. All of the statistics will be calculated and stored in the tape storage server metadata. See Section 12.2.56: *settapestats — Compute Tape Storage Server Statistics* on page 469 of the *HPSS Management Guide* for details.



Storage Server Configuration

Storage Server Configuration

Server Name	Disk Storage Server 1
Server ID	01d455d4-9418-11d4-bd20-0004ac714349

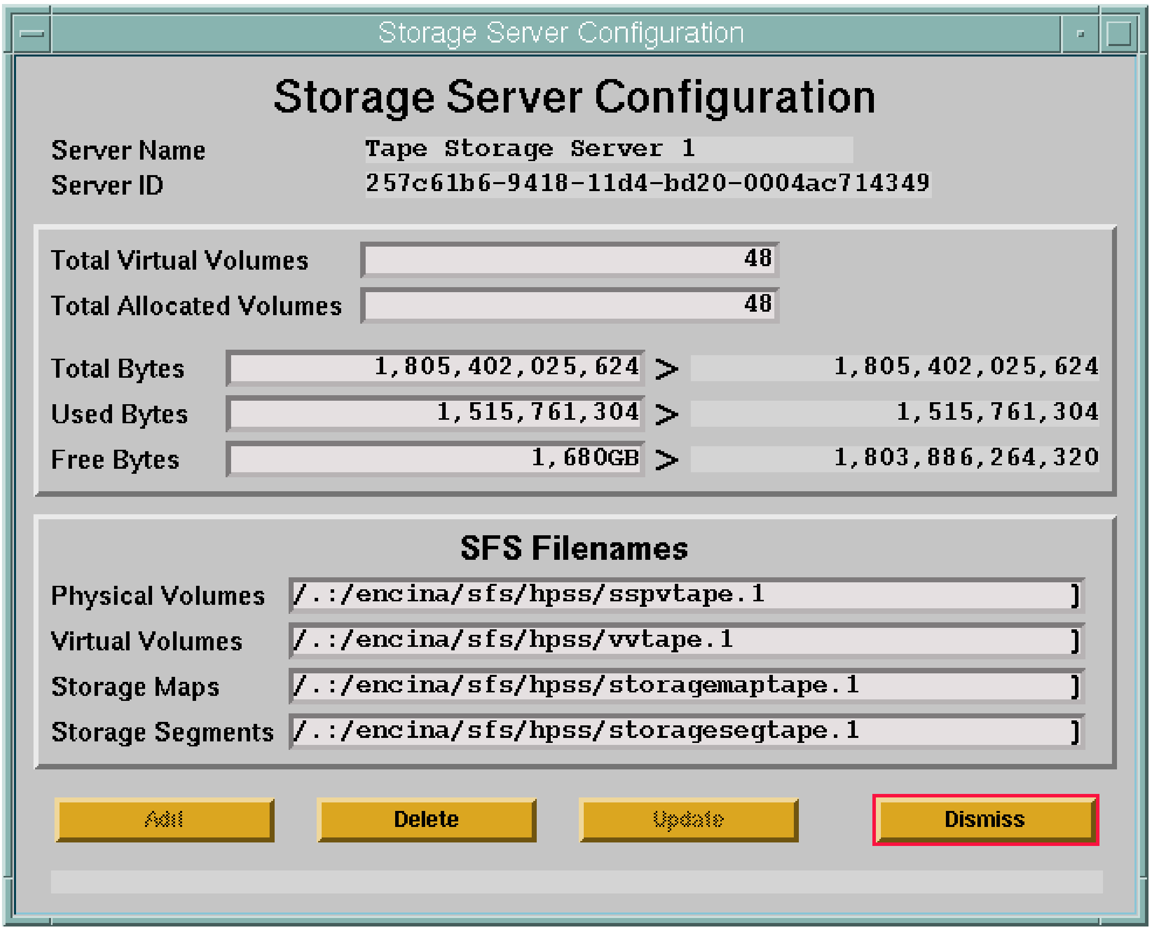
Total Virtual Volumes
Total Allocated Volumes
Total Bytes
Used Bytes
Free Bytes

These fields are not available for Disk Storage Servers.

SFS Filenames

Physical Volumes	/.:/encina/sfs/hpss/sspvdisk.1
Virtual Volumes	/.:/encina/sfs/hpss/vvdisk.1
Storage Maps	/.:/encina/sfs/hpss/storagemapdisk.1
Storage Segments	/.:/encina/sfs/hpss/storagesegdisk.1

Figure 6-40 Disk Storage Server Configuration Window


Storage Server Configuration

Storage Server Configuration

Server Name	Tape Storage Server 1
Server ID	257c61b6-9418-11d4-bd20-0004ac714349
Total Virtual Volumes	48
Total Allocated Volumes	48
Total Bytes	1,805,402,025,624 > 1,805,402,025,624
Used Bytes	1,515,761,304 > 1,515,761,304
Free Bytes	1,680GB > 1,803,886,264,320

SFS Filenames

Physical Volumes	/.:/encina/sfs/hpss/sspvtape.1
Virtual Volumes	/.:/encina/sfs/hpss/vvtape.1
Storage Maps	/.:/encina/sfs/hpss/storagemaptape.1
Storage Segments	/.:/encina/sfs/hpss/storagesegtape.1

Figure 6-41 Tape Storage Server Configuration Window

6.8.14.1 Storage Server Configuration Variables

Table 6-32 lists the fields on the Storage Server Configuration window and provides specific recommendations for configuring a Storage Server for use by HPSS.

Table 6-32 Storage Server Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Server Name	The descriptive name of the SS. This name is copied from the SS general configuration entry.	This field cannot be modified. It is displayed for reference only.	The selected SS descriptive name

Table 6-32 Storage Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Server ID	The UUID of this SS. This ID is copied from the SS general configuration entry.	This field cannot be modified. It is displayed for reference only.	Extracted from the SS general server configuration entry.
Statistics Fields. The following fields are displayed for Tape Storage Servers only.			
Total Virtual Volumes	The number of virtual volumes that are managed by this SS. This field is applicable to the Tape Storage Server only.	Any positive 64-bit integer value.	0
	Advice: This value should be set to zero when the specific configuration record is first created. The SS will calculate and update the configuration file as virtual volumes are added and deleted from the system. This value should only be changed when the SS is not running. See the documentation on the utility program "settapestats" for information about updating this value in established servers.		
Total Allocated Volumes	The number of virtual volumes that are either available for space allocation or full. This is the number of virtual volumes known to the server that are not in a "scratch" state. This field is applicable to the Tape Storage Server only.	Any positive 64-bit integer value.	0
	Advice: This value should be set to zero when the specific configuration record is first created. The SS will calculate and update the configuration file as virtual volumes are added and deleted from the system, and as volume states change. This value should only be changed when the SS is not running. See the documentation on the utility program "settapestats" for information about updating this value in established servers.		

Table 6-32 Storage Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Total Bytes	The total number of bytes of used and available storage known to the SS. This field is applicable to the Tape Storage Server only.	Any positive 64-bit integer value.	0
	Advice: This value should be set to zero when the specific configuration record is first created. The SS will calculate and update the configuration file as virtual volumes are added and deleted from the system. This number is an estimate of the amount of storage space, not an accurate number. This value should only be changed when the SS is not running. See the documentation on the utility program "settapestats" for information about updating this value in established servers.		
Used Bytes	The number of bytes written in storage segments. This field is applicable to the Tape Storage Server only.	Any positive 64-bit integer value.	0
	Advice: This value should be set to zero when the specific configuration record is first created. The SS will calculate and update the configuration file as storage segments are written, shortened, or deleted, and as volume state change. This value should only be changed when the SS is not running and only when the administrator is absolutely sure that the value is incorrect. See the documentation on the utility program "settapestats" for information about updating this value in established servers.		
Free Bytes	An estimate of the amount of unused storage space managed by the SS. This field is applicable to the Tape Storage Server only.	Any positive 64-bit integer value.	0
	Advice: This value should be set to zero when the specific configuration record is first created. The SS will calculate and update the configuration file as storage segments are written, shortened, or deleted (disk only), and as virtual volumes are added. This value should only be changed when the SS is not running. See the documentation on the utility program "settapestats" for information about updating this value in established servers.		
SFS Filenames. The fields below list the names of the SFS files used by the SS.			

Table 6-32 Storage Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Physical Volumes	The name of the SFS file that contains the physical volume metadata.	Valid SFS file name. The file must not be shared with other Storage Servers.	/.:/encina/sfs/hpss/sspvdisk.# for Disk SS /.:/encina/sfs/hpss/sspvtape.# for Tape SS
	Advice: Use a name that is meaningful to the type of SS and metadata being stored.		
Virtual Volumes	The name of the SFS file that contains the virtual volume metadata.	Valid SFS file name. The file must not be shared with other Storage Servers.	/.:/encina/sfs/hpss/vvdisk.# for Disk SS /.:/encina/sfs/hpss/vvtape.# for Tape SS
	Advice: Use a name that is meaningful to the type of SS and metadata being stored.		
Storage Maps	The name of the SFS file that contains the storage map metadata.	Valid SFS file name. The file must not be shared with other Storage Servers.	/.:/encina/sfs/hpss/storagemapdisk.# for Disk SS /.:/encina/sfs/hpss/storagemaptape.# for Tape SS
	Advice: Use a name that is meaningful to the type of SS and metadata being stored.		

Table 6-32 Storage Server Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Storage Segments	The name of the SFS file that contains the storage segment metadata.	Valid SFS file name. The file must not be shared with other Storage Servers.	/.:/encina/sfs/hpss/storagesegdisk.# for Disk SS /.:/encina/sfs/hpss/storagesegtape.# for Tape SS
	Advice: Use a name that is meaningful to the type of SS and metadata being stored.		

6.9 Configure MVR Devices and PVL Drives

The MVR Device and PVL Drive objects refer to the same physical drive, but are maintained in two separate SFS files based on what information is required by the PVL for drive management and what information is required by the Mover for device management. The two SFS files are managed by SSM from the same configuration window so that they can be synchronized with one another.

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

Before adding, deleting or modifying MVR Device and PVL Drive configuration, HPSS requires that the PVL and related the Mover be shutdown to ensure consistency. If a tape drive is being modified, then the corresponding PVR must also be shutdown. SSM will enforce this restriction.

To set the devices to use variable blocks on an AIX platform, use either the **chdev** command (substituting the appropriate device name for **rmt0**):

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

or **smitty**:

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

For Solaris, the method used to enable variable block sizes for a tape device is dependent on the type of driver used. Consult the tape device driver documentation for instructions on installation and configuration.

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

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

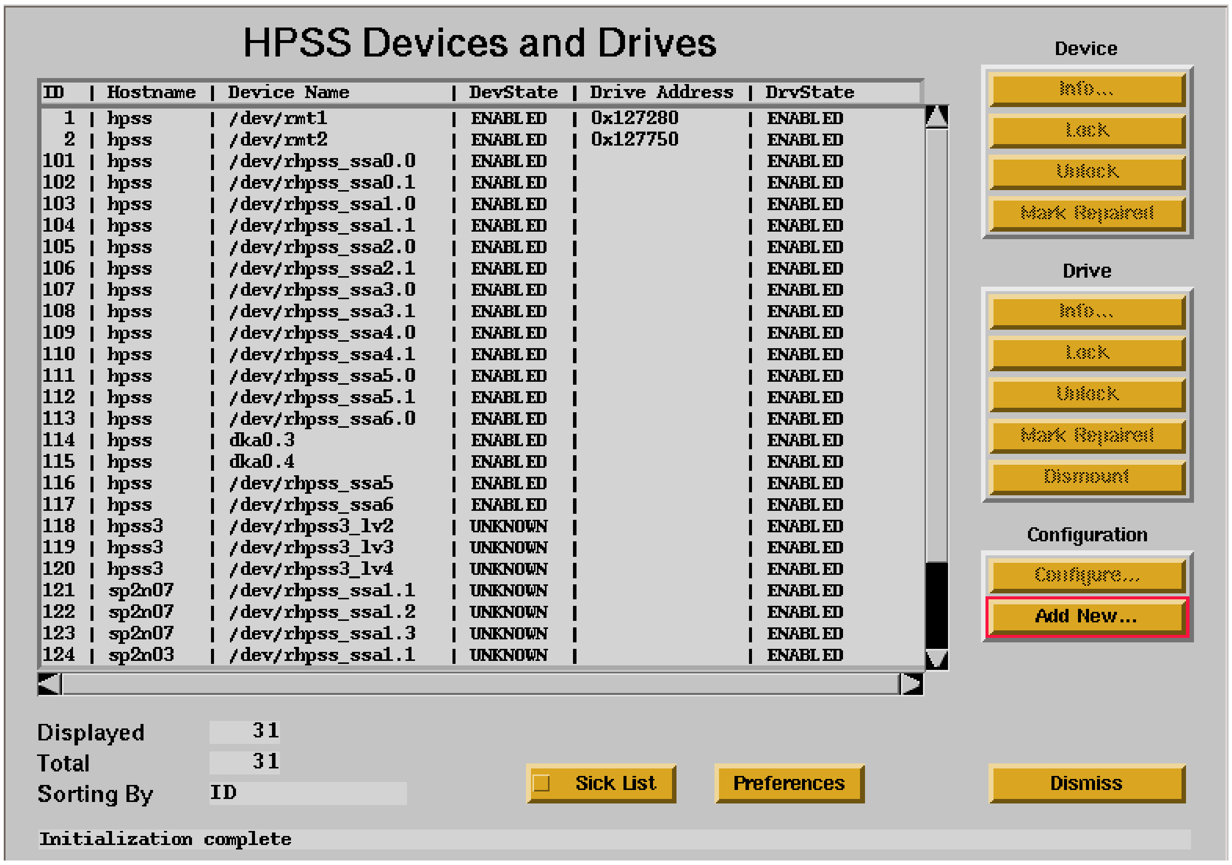
Before proceeding with device and drive configuration associated with StorageTek, or IBM robotics, refer to , Section 6.8.13.2: LTO PVR Information on page 387, Section 6.8.13.3: IBM 3494/3495 PVR Information on page 388, Section 6.8.13.4: StorageTek PVR and RAIT PVR Information on page 389, or Section 6.8.13.5: ADIC Automatic Media Library Storage Systems Information on page 392 as appropriate. These sections provide additional vendor-specific advice on robot and drive configuration.

A Device and Drive configuration entry can be created, updated and deleted only if the PVL and the associated Mover are not running.

The Device and Drive configuration entry can be created using the Mover Device and PVL Drive Configuration window. After the configuration entry is created, it can be viewed, updated, or deleted through the same window.

Before configuring a new device, you must shutdown the PVL and the Mover that will manage the device. If this will be a tape device, you must also shutdown the PVR that will manage the drive (see Section 1.8: *Shutting Down an HPSS Server* on page 32 of the *HPSS Management Guide* for details on shutting down HPSS servers).

From the HPSS Health and Status window (shown in Figure 6-1), click on the **Admin** menu, select the **Configure HPSS** option and click on the **Devices and Drives** option. The HPSS Devices and Drives window will be displayed as shown in Figure 6-42.



HPSS Devices and Drives

```
ID  | Hostname | Device Name        | DevState | Drive Address | DrvState
  1 | hpss     | /dev/rmt1          | ENABLED  | 0x127280      | ENABLED
  2 | hpss     | /dev/rmt2          | ENABLED  | 0x127750      | ENABLED
101 | hpss     | /dev/rhpss_ssa0.0  | ENABLED  |               | ENABLED
102 | hpss     | /dev/rhpss_ssa0.1  | ENABLED  |               | ENABLED
103 | hpss     | /dev/rhpss_ssa1.0  | ENABLED  |               | ENABLED
104 | hpss     | /dev/rhpss_ssa1.1  | ENABLED  |               | ENABLED
105 | hpss     | /dev/rhpss_ssa2.0  | ENABLED  |               | ENABLED
106 | hpss     | /dev/rhpss_ssa2.1  | ENABLED  |               | ENABLED
107 | hpss     | /dev/rhpss_ssa3.0  | ENABLED  |               | ENABLED
108 | hpss     | /dev/rhpss_ssa3.1  | ENABLED  |               | ENABLED
109 | hpss     | /dev/rhpss_ssa4.0  | ENABLED  |               | ENABLED
110 | hpss     | /dev/rhpss_ssa4.1  | ENABLED  |               | ENABLED
111 | hpss     | /dev/rhpss_ssa5.0  | ENABLED  |               | ENABLED
112 | hpss     | /dev/rhpss_ssa5.1  | ENABLED  |               | ENABLED
113 | hpss     | /dev/rhpss_ssa6.0  | ENABLED  |               | ENABLED
114 | hpss     | dka0.3             | ENABLED  |               | ENABLED
115 | hpss     | dka0.4             | ENABLED  |               | ENABLED
116 | hpss     | /dev/rhpss_ssa5    | ENABLED  |               | ENABLED
117 | hpss     | /dev/rhpss_ssa6    | ENABLED  |               | ENABLED
118 | hpss3    | /dev/rhpss3_lv2    | UNKNOWN  |               | ENABLED
119 | hpss3    | /dev/rhpss3_lv3    | UNKNOWN  |               | ENABLED
120 | hpss3    | /dev/rhpss3_lv4    | UNKNOWN  |               | ENABLED
121 | sp2n07   | /dev/rhpss_ssa1.1  | UNKNOWN  |               | ENABLED
122 | sp2n07   | /dev/rhpss_ssa1.2  | UNKNOWN  |               | ENABLED
123 | sp2n07   | /dev/rhpss_ssa1.3  | UNKNOWN  |               | ENABLED
124 | sp2n03   | /dev/rhpss_ssa1.1  | UNKNOWN  |               | ENABLED
```

Displayed	31
Total	31
Sorting By	ID

```
Initialization complete
```

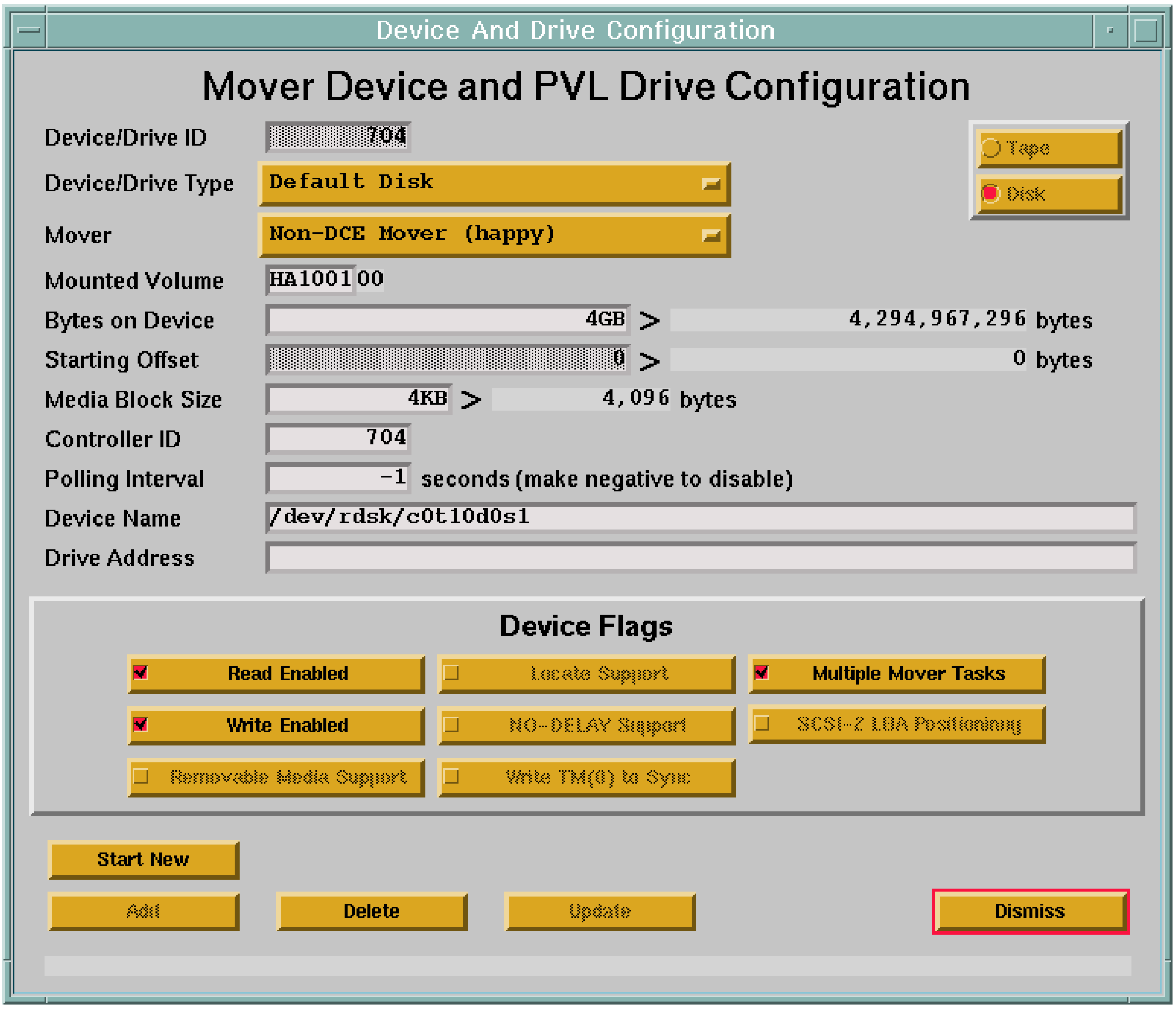
Figure 6-42 HPSS Devices and Drives Window

To configure a new device and drive, click on the **Add New...** button on the HPSS Devices and Drives window. The Mover Device and PVL Drive Configuration window will be displayed as shown in Figure 6-44 with default values for a new tape device / drive. If a disk device / drive is desired, click the **Disk** button to display the default disk data (as shown in Figure 6-43) before modifying any other fields. If the default data is not desired, change the field with the desired value. Click on the **Add** button to create the configuration entry.

To update an existing device and drive, select the desired device and drive entry on the HPSS Devices and Drives window and click on the **Configure...** button. The Mover Device and PVL Drive Configuration window will be displayed with the configured data. After modifying the configuration, click on the **Update** button to write the changes to the appropriate SFS file.

To delete an existing device and drive, select the desired device and drive on the HPSS Devices and Drives window and click on the **Configure...** button. The Mover Device and PVL Drive Configuration window will be displayed with the configured data. Click on the **Delete** button to delete the configuration entry.

Refer to the window's help file for more information on the individual fields and buttons as well as the supported operations available from the window.


Device And Drive Configuration

Mover Device and PVL Drive Configuration

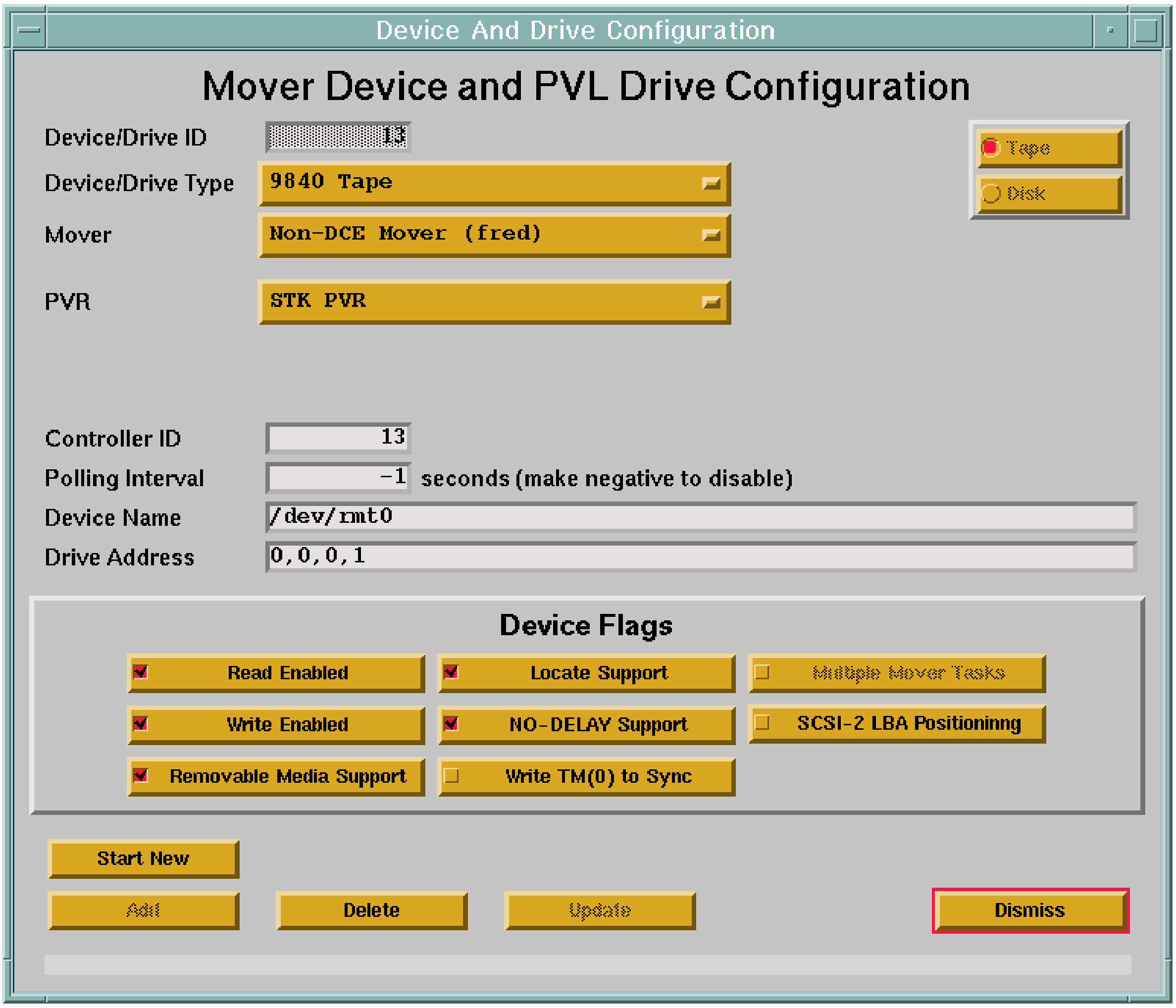
Device/Drive ID	704	
Device/Drive Type	Default Disk	
Mover	Non-DCE Mover (happy)	
Mounted Volume	HA100100	
Bytes on Device	4GB >	4,294,967,296 bytes
Starting Offset	0 >	0 bytes
Media Block Size	4KB >	4,096 bytes
Controller ID	704	
Polling Interval	-1	seconds (make negative to disable)
Device Name	/dev/rdsk/c0t10d0s1	
Drive Address		

Tape [✓ Disk]

Device Flags

- ✓ Read Enabled
- Locate Support
- ✓ Multiple Mover Tasks
- ✓ Write Enabled
- NO-DELAY Support
- SCSI-2 LBA Positioning
- Removable Media Support
- Write TM(0) to Sync

Figure 6-43 Disk Mover Device and PVL Drive Configuration Window


Device And Drive Configuration

Mover Device and PVL Drive Configuration

Device/Drive ID	13
Device/Drive Type	9840 Tape
Mover	Non-DCE Mover (fred)
PVR	STK PVR
Controller ID	13
Polling Interval	-1 seconds (make negative to disable)
Device Name	/dev/rmt0
Drive Address	0,0,0,1

[✓ Tape] Disk

Device Flags

- [✓] Read Enabled
- [✓] Locate Support
- [ ] Multiple Mover Tasks
- [✓] Write Enabled
- [✓] NO-DELAY Support
- [ ] SCSI-2 LBA Positioninng
- [✓] Removable Media Support
- [ ] Write TM(0) to Sync

Figure 6-44 Tape Mover Device and PVL Drive Configuration Window

6.9.1 Device and Drive Configuration Variables

Table 6-33 lists the fields on the Mover Device and PVL Drive Configuration window.

Table 6-33 Device/Drive Configuration Variables

Display Field Name	Description	Acceptable Values	Default Value
Device/Drive ID	The unique, numeric ID associated with this device/drive.	Any non-zero, positive 32-bit integer value.	Highest ID found in the SFS file plus 1.

Table 6-33 Device/Drive Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Device /Drive Type	The type of device over which data will move.	Any valid type from the pop-up list.	Default Tape or Default Disk, based on Tape/Disk toggle buttons.
Mover	The name of the MVR that controls the device.	Any configured MVR name from the pop-up list.	First configured MVR name found in the SFS file.
	Advice: There is a maximum of 64 devices that can be configured for a Mover.		
PVR	The name of the PVR that handles the removable media operations for the drive.	Any configured PVR name from the pop-up list.	First configured PVR name found in the SFS file.
	Advice: This field is meaningful for tape devices only.		
Mounted Volume	The 8-character name of the volume mounted on the disk drive, if any.	Valid 8-character name assigned to the disk volume.	None
	Advice: This value is meaningful for disk devices only.		
Bytes on Device	The size of device in bytes.	Any positive 64-bit integer value up to the system-imposed maximum device size.	None
	Advice: This value is used for disk devices only. If it is modified after the mounted disk volume has been imported into HPSS, the volume must be re-imported into HPSS. Note that the **PV Size** from Section 6.7 for the storage class must be less than or equal to the value of **Bytes on Device**. If the **Starting Offset** is non-zero, then the **Bytes on Device** value cannot be greater than the actual size of the underlying device less the **Starting Offset** value.		

Table 6-33 Device/Drive Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Starting Offset	The offset in bytes from the beginning of the disk logical volume at which the Mover will begin using the volume. The space preceding the offset will not be used by HPSS.	Zero, or a positive integer that is a multiple of the **Media Block Size**.	Zero
	Advice: This value is used for disk devices only.		
Media Block Size	The block size for the device.	Any positive 32-bit integer value.	None
	Advice: This value is used for disk devices only, and must be a multiple of the underlying disk block size.		
Controller ID	An indication of the adapter/bus that is in the data path to the device.	Any positive 32-bit integer value.	Set to Device/ Drive ID
	Advice: This field is used to attempt to mount separate volumes of a set of striped media on separate controllers, when possible, to prevent contention on the data path. It is recommended that the drives that are on the same adapter/bus have the same Controller ID. For drives in 3494 and 3495 robots, the Controller ID must be a valid ID. The valid IDs can be found in the Affinity list for any cartridge in the robot. Use the "mtlib -l <device name> -qV -V<volume name>" to obtain the Affinity list for a cartridge.		
Polling Interval	The number of seconds to wait between polling requests performed by the PVL to determine if any media is present in the drive.	Any positive 32-bit integer value for polling or any negative 32-bit integer value to disable polling.	60 seconds for tape; -1 for disk
	Advice: This field should be set to negative to disable polling for drives that do not support removable media. The value for drives located within robotic libraries should probably be set higher than for manually mounted drives or set to a negative value (preferred). We recommend that this field be set to 15 seconds for operator-mounted tape drives, 60 for tape robots, and -1 for disk drives.		

Table 6-33 Device/Drive Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Device Name	The name by which the MVR can access the device.	Any valid UNIX path name of a device file.	None
	Advice: This name is usually the path name of a device special file such as /dev/ rmt0/ For locally attached disk devices, the pathname should refer the raw/character special file (e.g., **/dev/rhpss_disk1**). For AIX systems, SCSI attached tape drives are typically referred to by pathnames of the form **/dev/rmtX**, where **X** begins at zero and is incremented for each tape drive detected. For IRIX systems, SCSI attached tape drives are typically referred to by pathnames of the form **/dev/rmt/tpsXdYns**, where **X** is the SCSI controller number, and **Y** is the SCSI ID of the drive. Note that for Ampex DST drives, the **tpsXdYnrns** name should be used (indicating that the driver should not attempt to rewind the drive upon close). For other drives on IRIX, the **tpsXdYnsvc** name should be used (indicating that the driver allow compression and variable block sizes). For Solaris systems, SCSI attached tape drives are typically referred to by pathnames of the form **/dev/rmt/Xc**, where **X** begins at zero and is incremented for each tape drive detected (the '**c**' indicates that compression is enabled). In particular note that the device that contains a '**b**' in the name should NOT be used, as this will change the behavior of the drive which will cause the HPSS Mover to fail. For Linux systems, this is the name that will be used to provide access to the SCSI raw device. The pathname will be in the form **/dev/raw/rawX**, where **X** specifies the raw device number. You can run the "**raw -q -a**" command to determine the correct raw device mappings. SCSI tape devices are referred to by pathnames of the form **/dev/stX**, where **X** begins at zero and is incremented for each LUN detected.		

Table 6-33 Device/Drive Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Drive Address	The name/address by which the PVR can access the drive.	Valid drive address (see Advice below).	None
	Advice: *For StorageTek robots:* Drive Address configuration entries correspond to the ACS,Unit,Panel,Drive Number used by ACSLS to identify drives. For example, the first drive in a typical configuration has Drive Address **0,0,10,0** (see Appendix K for information about configuration of StorageTek drives). *For 3584 LTO robots:* The Drive Address configuration entries correspond to the SCSI address (location) of the drive in the library. Determine the drive location by running an inventory on the library either through the **tapeutil** utility supplied with the Atape driver or using the library console. *For IBM 3494/3495 robots:* The Drive Address configuration entries correspond to the hexadecimal Library device number of the drive. Determine the Library device number by running the command "**/opt/hpss/bin/GetESANumbers /dev/rmtX**" for each tape drive in the robot. *For operator mounted drives:* For manually mounted drives, the drive address string is displayed to operators when an explicit drive is selected for mounting. The drive address should therefore be a string that easily communicates to an operator the drive in question (i.e., a name matching the label on the exterior of the drive). *For AML robots:* Leave the Drive Address field blank.		
Device Flags. The following fields are the device flags used by the MVR. Refer to Table for more information on the recommended settings for tape devices.			
Read Enabled	An indication of whether the device is available for reading.	ON, OFF	ON
Write Enabled	An indication of whether the device is available for writing.	ON, OFF	ON
Removable Media Support	An indication of whether the device supports removable media.	ON, OFF	ON for tape, OFF for disk

Table 6-33 Device/Drive Configuration Variables (Continued)

Display Field Name	Description	Acceptable Values	Default Value
Locate Support	An indication of whether the device supports a high speed (absolute) positioning operation.	ON, OFF	ON
	Advice: This option is supported for 3480, 3490, 3490E, 3590, 3580, Timberline, Redwood, 9840, 9940, DST-312, DST-314, and GY-8240 devices.		
NO-DELAY Support	An indication of whether the device supports opening the device with no delay flag set, while allowing tape I/O operation after the open.	ON, OFF	ON
	Advice: On some tape devices, this will allow for a quicker polling operation when no tape is presently loaded in the device. This field is meaningful for tape devices only. This option is not supported for the BMUX attached 3480, 3490, 3490E and 3590 devices.		
Write TM(0) to Sync	An indication of whether the device supports issuing a write tape mark request with a zero count to flush data written previously to the tape media.	ON, OFF	OFF
	Advice: On some devices, this may provide a higher performance method to ensure that data has been safely written to the media. This option is not used for the 3480, 3490, 3490E, 3590, Timberline, and Redwood devices, provided the HPSS supported device driver is used in accessing the device. Note that for Ampex DST-312 this field should be set to "ON".		
Multiple Mover Tasks	An indication of whether the MVR should allow multiple MVR tasks to simultaneously access the device.	ON, OFF	ON for disk, OFF for tape
	Advice: This value is meaningful for disk devices only.		
SCSI-2 LBA Positioning	If ON, the tape device supports SCSI-2 Logical Block Addresses.	ON, OFF	OFF
	Advice: If SCSI-2 LBAs and the SCSI LOCATE command (Locate Support) are supported by the device, HPSS will calculate destination LBAs based on known LBAs and relative addresses. LBA positioning provides for faster access of data residing on tape. The benefit will be realized for read requests with many source descriptors specifying locations spread sparsely down the tape. This is only supported with the IBM SCSI tape device driver.		

6.9.2 Supported Platform/Driver/Tape Drive Combinations

Table 6-34 Supported Platform/Driver/Tape Drive Combinations

Platform	Driver	Device(s)
AIX	IBM	3490, 3590, 3590E, 3590H, 3580
	AdvanTAPE	Redwood, Timberline, 9840, 9940, 9940B, RAIT 9840, RAIT 9940
	Native	9840, 9940, 9840B, 9940B
Solaris	IBM	3490, 3590, 3590E, 3590H, 3580
	Native	Redwood, Timberline, 9840, 9940, 9940B, RAIT 9840, RAIT 9940
IRIX	Native	9840, 9940, 9940B, RAIT 9840, RAIT 9940, GY-8240
	Ampex	DST-312, DST-314
Linux	Native	3580

In Table 6-34, the “Driver” column uses the following abbreviations:

AdvanTAPE - Gresham Tape Device Driver

Ampex - Ampex Tape Device Driver

IBM - IBM SCSI Tape Device Driver

Native - AIX, Solaris, IRIX, or Linux native SCSI Tape Device Driver

6.9.3 Recommended Settings for Tape Devices

Table 6-35 Recommended Settings for Tape Devices

Device Driver	Mover TCP Executable	NO-DELAY Support	Locate Support	Write TM(0) to Sync
Gresham	hpss_mvr_omi	ON	ON	OFF
Solaris Native	hpss_mvr_tcp	ON	OFF	OFF
IBM SCSI Tape	hpss_mvr_ssd	ON	ON	OFF
IRIX Native	hpss_mvr_tcp	ON	ON	OFF
Ampex	hpss_mvr_dd2	OFF	ON	ON
AIX Native	hpss_mvr_tcp	ON	OFF	OFF
Linux Native	hpss_mvr_tcp	ON	ON	OFF

Chapter 7

HPSS User Interface Configuration

7.1 Client API Configuration

The following environment variables can be used to define the Client API configuration:

The **HPSS_LS_NAME** defines the CDS name of the Location Server RPC Group entry for the HPSS system that the Client API will attempt to contact. The default is **/.:/hpss/ls/group**.

The **HPSS_MAX_CONN** defines the number of connections that are supported by the Client API within a single client process. If HPSS_MAX_CONN is set to zero, the number of connections is equal to the default supported by the HPSS connection management software - currently 150. If HPSS_MAX_CONN is nonzero, it is the number of connections to be used.

The **HPSS_KTAB_PATH** defines the name of the file containing the DCE security keys necessary for successfully initializing the Client API. The default is **/krb5/hpssclient.keytab**.

The **HPSS_HOSTNAME** environment variable is used to specify the hostname to be used for TCP/IP listen ports created by the Client API. The default value is the default hostname of the machine on which the Client API is running. This value can have a significant impact on data transfer performance for data transfers that are handled by the Client API (i.e., those that use the **hpss_Read** and **hpss_Write** interfaces).

The **HPSS_TCP_WRITESIZE** environment variable is used to specify the amount of data to be written with each individual request to write data to a network connection during a data transfer. For some networks, writing less than the entire size of the client buffer has resulted in improved throughput. This environment variable may not affect the actual value used, depending on the HPSS network options file - see Section 6.10.2 for further details.

The **HPSS_TRANSFER_TYPE** environment variable is used to specify the data transport mechanism to be used for data transfers handled by the Client API. The only valid value is **TCP** for TCP/IP transfer. The default value is **TCP**.

The **HPSS_PRINCIPAL** environment variable is used to specify the DCE principal to be used when initializing the HPSS security services. The default value is **hpss_client_api**. This variable is primarily intended for use by HPSS servers that use the Client API.

The **HPSS_SERVER_NAME** environment variable is used to specify the server name to be used when initializing the HPSS security services. The default value is **/.:/hpss/client**. This variable is primarily intended for use by HPSS servers that use the Client API.

The **HPSS_DESC_NAME** environment variable is used to control the descriptive name used in HPSS log messages if the logging feature of the Client API is enabled. The default value is "**Client Application**".

The Client API, if compiled with debugging enabled, uses two environment variables to control printing debug information. **HPSS_DEBUG**, if set to a non-zero value, will enable debug messages. By default, these messages will go to the standard output stream. If **HPSS_DEBUGPATH** is set, however, these messages will be directed to the file indicated by this environment variable. Two special cases for the debug path exist: **stdout** and **stderr**, which will use the standard output or standard error I/O streams, respectively.

The **HPSS_NUMRETRIES** environment variable is used to control the number of retries to attempt when an operation fails. Currently this class of operation includes library initialization and communications failures. A value of zero indicates that no retries are to be performed, and value of **-1** indicates that the operation will be retried until successful. The default value is **4**.

The **HPSS_BUSY_RETRIES** environment variable is used to control the number of retries to be performed when a request fails because the Bitfile Server does not currently have an available thread to handle that request. A value of zero indicates that no retries are to be performed, and a value of **-1** indicates that retries should be attempted until either the request succeeds or fails for another reason. The default value is **3**.

The **HPSS_BUSY_DELAY** environment variable is used to control the number of seconds to delay between retry attempts. Note that this value is used both for retrying initialization operations (see **HPSS_NUMRETRIES**) and Bitfile Server requests (See **HPSS_BUSY_RETRIES**). The default value is **15**.

The **HPSS_RETRY_STAGE_INP** environment variables is used to control whether retries are attempted on attempts to open files in a Class of Service that is configured for background staging on open. A non-zero value indicates that opens which would return **-EINPROGRESS** to indicate that the file is being staged will be retried (using the same control mechanisms described in the previous paragraph), while a value of zero indicates that the **-EINPROGRESS** return code will be returned to the client. The default value is non-zero.

The **HPSS_REUSE_CONNECTIONS** environment variable is used to control whether TCP/IP connections are to be left open as long as a file is open or are to be closed after each read or write request. A non-zero value will cause connections to remain open, while a value of zero will cause connections to be close. The default value is zero.

The **HPSS_USE_PORT_RANGE** environment variable is used to control whether the HPSS Mover(s) should use the configured port range when making TCP/IP connections for read and write requests. A non-zero value will cause the Mover(s) to use the port range, while a value of zero will cause the Mover(s) to allow the operating system to select the port number. The default value is zero.

The **HPSS_TOTAL_DELAY** environment variable is used to control the number of seconds to continue retrying requests. A value of zero indicates that no there is no time limit. The default value is zero.

The **HPSS_REGISTRY_SITE_NAME** environment variable is used to specify the name of the security registry used when inserting security information into connection binding handles. This is only needed when the client must support DFS in a cross-cell environment. The default registry is "**/.../dce.clearlake.ibm.com**".

The **HPSS_DMAP_WRITE_UPDATES** environment variable is used control the frequency of cache invalidates that are issued to the DMAPI file system while writing to a file that is mirror in HPSS. The default value is **20**.

The **HPSS_GKTOTAL_DELAY** environment variable is used to control the total number of seconds to continue retrying a Gatekeeper request before the request times out. A value of zero indicates that there is not time limit. The default value is **600**.

The **HPSS_LIMITED_RETRIES** environment variable is used to control the number of retry attempts before a limited retry error operation fails. The default value is **1**.

The **HPSS_DISABLE_CROSS_CELL** environment variable is used to control cross-cell traversal. When cross-cell traversal is disabled, a client will not be allowed to access directories which are located in another DCE cell. The default value is zero (cross-cell enabled).

The **HPSS_DISABLE_JUNCTIONS** environment variable is used to control junction traversal. When junction traversal is disabled, a client will not be allowed to access directories which require traversal to the directory via a junction. The default value is zero (junction traversal enabled).

7.2 Non-DCE Client API Configuration

7.2.1 Configuration Files

The following files are required by the Non-DCE Client API to perform the authentication functions correctly under the specified circumstances.

ndcl.keyconfig: This file is required on the client machine in order to use the DCE authentication mode. It contains the encryption key used to encrypt the DCE principal and the password (which are then shipped across to the Non DCE Gateway). By default this file is in the **/var/hpss/etc** directory

The keyfile must contain the same key as the **Non DCE Gateway Server Specific Configuration** screen in SSM. You must perform the following steps to set up the keyfile.

- Login on the client machine (the one running the Non DCE Client API) as user **hpss**.
- **cd /var/hpss/etc**
- Open the **Non-DCE Gateway Server Specific Configuration** screen in SSM.
- Use **vi** or your text editor of choice to create and open **ndcl.keyconfig**
- Copy the key from the NDCG SSM screen to the keyfile in the following format:

```
"0x[first 8 digits] 0x[next 8 digits]"
```

Thus if the key on the NDCG SSM screen is `0123456789ABCDEF` then the key in the **ndcl.keyconfig** file must look like the sample file shown below:

```
0x01234567 0x89ABCDEF
```

- Make sure you set the appropriate permissions on this file. Only users authorized to use the Non DCE Client API should have access to this file.

You can specify an alternate pathname for this file by setting the **HPSS_NDCL_KEY_CONFIG_FILE** environment variable.

7.2.2 Environment Variables

The following environment variables can be used to define the Non-DCE Client API configuration. Do not confuse references to the Client API with references to the Non-DCE Client API. When reference is made to the Client API, the document is referring to the actual Client API calls made by the Non-DCE Client Gateway. When reference is made to the Non-DCE Client API, the document is referring to calls made directly by the Non-DCE client application.

The **HPSS_NDCG_NAME** environment variable is used to specify the server name to be used when initializing the HPSS security services. The default value is **/.:/hpss/client**.

The **HPSS_NDCG_TCP_PORT** defines the default port location for the listener process of the Non-DCE Client Gateway with which the Non-DCE Client API will communicate. This value can be overridden by appropriate entries in the **HPSS_NDCG_SERVERS** environment variable. The default value is **8001**.

The **HPSS_NDCG_SERVERS** defines the name of the server on which the Non-DCE Client Gateway resides. Multiple servers may be separated by a colon (:). If multiple servers are specified, the NDAPI will randomly choose a listed server every time it establishes a new gateway connection. This way, multiple NDCGs may be configured for load-balancing. It is possible to explicitly set the TCP port on a per server basis by following the server name with a forward slash (/) and a port number. For example, a string **"hpss/8002:pluto"** would define two Non-DCE Client Gateways. One (hpss) uses an explicit port number, and the other (pluto) uses the value from **HPSS_NDCG_TCP_PORT**.

The **HPSS_LOGGING_PORT** defines the port number of the Log Client on the host running the connecting Non-DCE Client Gateway. The default value is **8101**.

The **HPSS_LOGGING_TYPE** defines the types of messages to log. It consists of a list of log types, separated by colons (:). For example **"CS_ALARM:CS_STATUS"** would log alarm and status messages. Valid log types are: **CS_ALARM**, **CS_EVENT**, **CS_REQUEST**, **CS_SECURITY**, **CS_ACCOUNTING**, **CS_DEBUG**, **CS_TRACE**, and **CS_STATUS**. The default value is **"CS_ALARM: CS_EVENT:CS_REQUEST:CS_SECURITY"**.

The **HPSS_MAX_CONN** defines the number of connections that are supported by the Client API within a single client process. The default is zero which is equal to the default supported by the HPSS connection management software - currently 150.

The **HPSS_KTAB_PATH** defines the name of the file containing the DCE security keys necessary for successfully initializing the Client API. The default is **/krb5/hpssclient.keytab**.

The **HPSS_HOSTNAME** environment variable is used to specify the hostname to be used for TCP/IP listen ports created by the Client API. The default value is the default hostname of the machine on which the Client API is running. This value can have a significant impact on data transfer performance for data transfers that are handled by the Client API (i.e., those that use the **hpss_Read** and **hpss_Write** interfaces).

The **HPSS_TCP_WRITESIZE** environment variable is used to specify the amount of data to be written with each individual request to write data to a network connection during a data transfer. For some networks, writing less than the entire size of the client buffer has resulted in improved throughput. This environment variable may not affect the actual value used, based on the contents of the HPSS network options file - see Section 6.10.2 for further details.

The **HPSS_TRANSFER_TYPE** environment variable is used to specify the data transport mechanism to be used for data transfers handled by the Client API. The only valid value is **TCP** for TCP/IP transfers. The default value is **TCP**.

The **HPSS_PRINCIPAL** environment variable is used to specify the DCE principal to be used when initializing the HPSS security services. The default value is **hpss_client_api**.

The **HPSS_DESC_NAME** environment variable is used to control the descriptive name used in HPSS log messages if the logging feature of the Client API is enabled. The default value is “Client Application”.

The Client API, if compiled with debugging enabled, uses two environment variables to control printing debug information. **HPSS_DEBUG**, if set to a non-zero value, will enable debug messages. By default, these messages will go to the standard output stream. If **HPSS_DEBUGPATH** is set, however, these messages will be directed to the file indicated by this environment variable. Two special cases for the debug path exist: **stdout** and **stderr**, which will use the standard output or standard error I/O streams, respectively.

The **HPSS_NUMRETRIES** environment variable is used to control the number of retries to attempt when an operation fails. Currently this class of operation includes library initialization and communications failures. A value of zero indicates that no retries are to be performed, and value of “-1” indicates that the operation will be retried until successful. The default value is 4

The **HPSS_BUSY_RETRIES** environment variable is used to control the number of retries to be performed when a request fails because the Bitfile Server does not currently have an available thread to handle that request. A value of zero indicates that no retries are to be performed, and a value of “-1” indicates that retries should be attempted until either the request succeeds or fails for another reason. The default value is 3.

The **HPSS_BUSY_DELAY** environment variable is used to control the number of seconds to delay between retry attempts. Note that this value is used both for retrying initialization operations (see **HPSS_NUMRETRIES**) and Bitfile Server requests (See **HPSS_BUSY_RETRIES**). The default value is 15.

The **HPSS_RETRY_STAGE_INP** environment variables is used to control whether retries are attempted on attempts to open files in a Class of Service that is configured for background staging on open. A non-zero value indicates that opens which would return **-EINPROGRESS** to indicate that the file is being staged will be retried (using the same control mechanisms described in the previous paragraph), while a value of zero indicates that the **-EINPROGRESS** return code will be returned to the client. The default value is non-zero.

The **HPSS_REUSE_CONNECTIONS** environment variable is used to control whether TCP/IP connections are to be left open as long as a file is open or are to be closed after each read or write request. A non-zero value will cause connections to remain open, while a value of zero will cause connections to be close. The default value is zero.

The **HPSS_USE_PORT_RANGE** environment variable is used to control whether the HPSS Mover(s) should use the configured port range when making TCP/IP connections for read and write requests. A non-zero value will cause the Mover(s) to use the port range, while a value of zero will cause the Mover(s) to allow the operating system to select the port number. The default value is zero.

The **HPSS_TOTAL_DELAY** environment variable is used to control the number of seconds to continue retrying requests. A value of zero indicates that no there is no time limit. The default value is 0.

The **HPSS_REGISTRY_SITE_NAME** environment variable is used to specify the name of the security registry used when inserting security information into connection binding handles. This is only needed when the client must support DFS in a cross-cell environment. The default registry is "**/.../dce.clearlake.ibm.com**".

The **HPSS_DMAP_WRITE_UPDATES** environment variable is used control the frequency of cache invalidates that are issued to the DMAPI file system while writing to a file that is mirror in HPSS. The default value is 20.

The **HPSS_NDCL_KEY_CONFIG_FILE** environment variable is used to specify an alternate file (default file is **/var/hpss/etc/ndcl.keyconfig**) that contains the key used to perform encryption in DCE authentication mode. This is set on the client machine.

The **HPSS_KRB_TO_DCE_FILE** is used to specify an alternate file (default file is **/var/hpss/etc/krb5_Realms_to_DCE_Cells**) that contains the mapping from kerberos realm names to dce cell names. This is set on the Non-DCE Gateway host.

HPSS_KCHILD_PATH is used to specify an alternate executable file (default is **/opt/hpss/bin/ndcg_kchild**) that handles kerberos authentication on the Gateway machine.

HPSS_NDCG_KRB5_SERVICENAME is used to specify the name of the Kerberos Service for the Non-DCE Gateway. Typically this should be something like **ndcg/<hostname>** where hostname is the name of the Non-DCE Gateway host machine.

HPSS_KCHILD_KEYTAB is used to specify an alternate keytab file (default is **/krb5/v5srvtab**) to be used to perform kerberos authentication. This is set on the Non-DCE Gateway host.

7.2.3 Authentication Setup

In order to enable the Non DCE Gateway and Client API to perform security in either DCE or Kerberos Modes the following setup needs to be done:

7.2.3.1 DCE Mode

This is the default security mode for the NDCG.

Perform the following on the NDCG server:

- If your OS supports it and you wish to use DES encryption to encrypt/decrypt your userid/password make sure you have the following line in **Makefile.macros** before compiling hpss.

  ```
  NDAPI_INTERNATIONAL_SUPPORT = off
  ```

  In case of international sites and for sites that don't have DES support, this flag can be set to **on** which will then use an alternate hashing mechanism to perform the encryption.

- Check the **DCE Authentication** box on the NDCG Gateway Type Specific config screen in SSM

Perform the following on the NDCL client:

- Copy the encryption key from the NDCG type specific config screen to the **ndcl.keyconfig** file (See Section 7.2.1: *Configuration Files* on page 415 for more information)

- Compile the client library (**ndcl**) with the **-DAUTH_TYPE_DCE** flag enabled in the **Makefile**

- Make sure you link the math libraries (**-lm**) with your client application when you build it. This is required for performing encryption.

7.2.3.2 Kerberos Mode

7.2.3.2.1 On both the Non DCE Client and Non DCE Gateway machines:

1. Create a kerberos realm that includes the client machine as well as the machine running the NDCG.

   This includes setting up the **/etc/krb5.conf** file on the client and the server.

   Sample **/etc/krb5.conf**:

   ```
   [libdefaults]
      default_realm = dopey_cell.clearlake.ibm.com
      default_keytab_name = /krb5/v5srvtab
      default_tkt_enctypes = des-cbc-crc
      default_tgs_enctypes = des-cbc-crc

   [realms]
      dopey_cell.clearlake.ibm.com = {
            kdc = dopey.clearlake.ibm.com:88
      }

   [domain_realm]
      dopey.clearlake.ibm.com = dopey_cell.clearlake.ibm.com
      happy.clearlake.ibm.com = dopey_cell.clearlake.ibm.com
   ```

Important fields in the **/etc/krb5.conf**:

[libdefaults] stanza:

The default_realm should map to the kerberos realm name you wish to use, which in most cases will be the same as the dce cell name.

The default_keytab_name (which is on the server host) is typically **/krb5/v5srvtab**. This is the keytab used by the ndcg kerberos service.

[realms] stanza:

This contains the information for the kerberos realm. It has the realm_name followed by information about the realm in the format shown in the sample file.

The kdc field is a required field. This must point to the fully qualified hostname on which the kdc (key distribution center) resides. Additionally one can specify the port used by the kdc (88)

[domain_realm] stanza:

This stanza contains mappings for the various hosts in the kerberos cell. At the very least it should have

- ❖ A mapping from the NDCG server host (**dopey.clearlake.ibm.com**) to the realm name (**dopey_cell.clearlake.ibm.com**)
- ❖ A mapping from the NDAPI client machine (**happy.clearlake.ibm.com**) to the realm name (**dopey_cell.clearlake.ibm.com**)

[Please note that this stanza requires fully qualified hostname.]

7.2.3.2.2 On the Non DCE Gateway machine:

2. Make sure you already have the **/etc/krb5.conf** and the **/krb5/v5srvtab** files
3. Add **ndcg** and **host** entries for target kerberos database.

   Thus if your hostname is **dopey.clearlake.ibm.com** your ndcg service will be called **ndcg/dopey.clearlake.ibm.com** and your host will be called **host/dopey.clearlake.ibm.com**

   Here is how to do it using **rgy_edit**:

   ➢ login as root and perform a **dce_login** as **cell_admin**

   ➢ run **rgy_edit** and issue the following commands:

```
rgy_edit=> domain principal
Domain changed to: principal
rgy_edit=> add host/dopey.clearlake.ibm.com -f \
                              "KRB5 host principal"
rgy_edit=> add ndcg/dopey.clearlake.ibm.com -f \
                              "KRB5 service principal"
```

```
rgy_edit=> domain account
Domain changed to: account
rgy_edit=> add host/dopey.clearlake.ibm.com
Enter account group [gname]: none
Enter account organization [oname]: none
Enter password: qwerty
Retype password: qwerty
Enter your password: <cell_admin_password>
Enter misc info: () KRB5 host account
[all other prompts can be answered with "Enter"]
...
rgy_edit=> add ndcg/dopey.clearlake.ibm.com
Enter account group [gname]: none
Enter account organization [oname]: none
Enter password:
Retype password:
Enter your password:
Enter misc info: () KRB5 service account
[all other prompts can be answered with "Enter"]
...
```

This would create host and ndcg entries for **dopey.clearlake.ibm.com**. Replace this host name with the actual name of the target. Use the fully domain-qualified name; Kerberos expects this!

The password represented as "**qwerty**" above should be any easily typed and easily remembered string, because it is going to get changed in the next step. None of the passwords you type are actually echoed to the screen.

Now Create keytab entries for the new principals on the target.

```
rgy_edit=> ktadd -p host/dopey.clearlake.ibm.com -pw qwerty
rgy_edit=> ktadd -p host/dopey.clearlake.ibm.com -r -a
rgy_edit=> ktadd -p ndcg/dopey.clearlake.ibm.com -pw qwerty
rgy_edit=> ktadd -p ndcg/dopey.clearlake.ibm.com -r -a
rgy_edit=> quit
```

4. Set the **HPSS_KCHILD_KEYTAB** environment variable if necessary. By default, the **/krb5/v5srvtab** keytab will be used.

5. Check the **kerberos** box in the **Non-DCE Gateway Configuration** SSM screen

6. Make sure the Non-DCE Gateway is set to run as **root**. This can be set through the Non-DCE Gateway's **Basic Server Configuration** SSM screen.

7.2.3.2.3 On the Non-DCE Client machine:

[Note: step 1 is in Section 7.2.3.2.1: *On both the Non DCE Client and Non DCE Gateway machines:* on page 419]

1. Compile the client library with kerberos mode enabled and link with the kerberos libraries. Use the following flags:

   ```
   -lkrb5 -lcrypto -lcom_err
   ```

   Make sure the linker knows where to find these libraries using the **-L** flag.

2. Call **hpss_SetAuthType** in your client program if other authentication modes have been enabled in the client

3. Run **kinit** to get the initial credentials for the desired principal

4. Make sure you use the fully qualified hostname when specifying the name of the target Non-DCE Gateway machine. This is usually set in the environment variable **HPSS_NDCG_SERVERS**.

7.3 FTP Daemon Configuration

There are four steps that must be performed prior to using the FTP user interface to transfer HPSS data:

1. Verifying the FTP Initial Configuration
2. Configuring the FTP Daemon Syslog
3. Defining the FTP Access
4. Creating FTP Users

These steps are described in more detail in the paragraphs that follow.

Step 1. Verifying the FTP Configuration

During the HPSS infrastructure configuration phase, the following files were modified by the **mkhpss** script to add the required FTP configuration:

- **/etc/services**—Verify that the **hpssftp** and **hpssftp-data** entries were added correctly. If possible, consider moving the ftp and ftp-data components to some other ports (4020/4021) and assign the hpssftp and **hpssftp-data** to ports 20/21. Reinitialize **inetd** after making these changes:

  ```
  % refresh –s inetd    (AIX only)
  ```

  or determine the pid for inetd and perform the following (all platforms):

  ```
  % kill –1 pid
  ```

- **/etc/inetd.conf—**Verify that the **hpssftp** entry was added correctly. The flags described below can be used in modifying this entry. Note that the flags are preceded by a dash (“**-”)** and that single character flags can be grouped in one argument (e.g., **`-adhvLSUX`**). If possible, use a "TCP Wrapper" application for initiating the HPSS FTP Daemon. This enhances on-the-fly changes to the startup of the HPSS FTP Daemon. Additionally, this provides for enhanced security. Several TCP Wrapper applications are available in the Public domain.

HPSS Parallel FTP Daemon options:

The only options which accept additional arguments are the **–p**, **-s**, **-D**, and **–F** options. The **–p** requires a port number (decimal value) while the other **D** and **F** options expect strings < **MAXPATHLEN** characters in length. The syntax on the –s option is explained below and should be handled carefully or indeterminate errors may occur! No space is allowed between the option and the value. **NOTE: that using long strings MAY cause severe problems for inetd!** Each of these four options MUST be preceded by a space and a "**-**". In general, it is recommended that these four options NOT be used!

All the other options should be specified by a single "**-**" at the start and the concatenation of options (e.g., "**-abdddhtuvzARSTUX**" – Not necessarily in that order)

NOTE the three consecutive "**d**"s imply to "**syslog**" as much debug information as possible. See the explanation below for the "**d**" option.

Table 7-1 Parallel FTP Daemon Options

Option	Description
a	Mandate Authentication with DCE/Kerberos Credentials - Disable{Username}/{Password} authentication. This also disables the **user** command.
b	Read the **hpss_option BUFSZ <value>** option from the {**ftpaccess**} file to obtain the path/name of the size specification to be used. Refer to documentation on the {**ftpaccess**} file for more details.
d	Turn on Debugging - Increments environment variable **HPSS_DEBUG** and sets **HPSS_DEBUGPATH=FTPBaseDir/adm/hpss_ftpd.log**. See option **-D** for details on **FTPBaseDir**. This option may be repeated up to 3 times. Each specification increases the detail of debug information sent to the syslog. **NOTE**: the information in the syslog is subject to the specifications in **/etc/syslog.conf**. For the most information, specify ***.debug** in **/etc/syslog.conf**.
h	Search the **ftpaccess** file for the **HOST** specifier - Defines the network interface to be used for data transferred between the PFTPD and the Movers when performing non-parallel transfers. **hpss_option HOST <interface name>** option sets the **HPSS_HOSTNAME** environment variable for the Client API. Refer to documentation on the **ftpaccess** file for details.
p#	Specify a port to be used by the HPSS PFTP Daemon. Used only when initiating the Daemon manually (rather than using **inetd**).

Table 7-1 Parallel FTP Daemon Options

Option	Description
s string	Specify the syslog facility for the HPSS PFTPD. The syntax on the **-s** option is **-slocal7**. The default syslog facility is **LOG_DAEMON** (reference: **/usr/include/sys/syslog.h**). Alternatives are **local0** - **local7**. Incorrect specification will default back to **LOG_DAEMON**. To make use of the alternates, modify **/etc/syslog.conf** to use the alternate facility. Note, the file specified in the **/etc/syslog.conf** must exist prior to initialization/refresh of the **syslogd**. No space is allowed between the **s** and the facility specifier.
t	Read the **hpss_option DTO <value>** option from the {ftpaccess} file to obtain the default timeout in seconds. Refer to documentation on the {ftpaccess} file for details.
u	Read the **hpss_option UMASK <value>** option from the {ftpaccess} file to obtain the default umask to be applied. Refer to documentation on the {ftpaccess} file for details
v	Increment the Client API Logging Specification. The Client API will perform logging specified by the **HPSS_DEBUG** environment variable in a file specified by the **HPSS_DEBUGPATH** environment variable (default name is **/var/hpss/ftp/adm/hpss_ftpd.log**.) The default value for the "**v**" option is **1**. (No "**v**" option required.) Future implementation may allow for increasing the value to either **2** or **3** to obtain additional Client API information. This would be accomplished by specifying **1** or **2** "**v**" options on the **hpss_pftp** startup line.
z	Sleep for 15 seconds at initialization. Useful when attempting to attach to the HPSS PFTP Daemon with the debugger.
A	Enable Passive Connections. Note: Not supported in the HPSS Parallel FTP Daemon for parallel operations. Client requests to use passive mode will be rejected.
C	Disable the ability for clients to explicitly set the Class of Service for new files (via the "**site setcos**" command).
Dstring	Set the **{FTPBaseDir}** path. Default: **/var/hpss/ftp**. This directory must contain several sub-directories including: **adm**, **bin**, **daemon**, and **etc**. Specific files/sub-directories are located in each of these subdirectories - **etc**: **ftpaccess**, [**ftpbanner**], **ftppasswd**, **ftpusers**, and [**trusted_hosts**]. **adm**: [**daemon.syslog**], [**hpss_ftpd.log**], [**xferlog**]. **daemon**: **ftpd/ftp.pids-hpss_class**. [ ] implies optional others are required. **etc/ftppassed** is optional if the **-S** or **-X** options are specified.
Fstring	Set the {**FTP_FtpAccessFile**}. Default: **ftpaccess**. Located in the directory **{FTPBaseDir}/etc**.
G	Read the **hpss_option AMGR <string>** from the **ftpaccess** file to obtain the path/name of the Authentication Manager to be used. Refer to documentation on the **ftpaccess** file for details.

Table 7-1 Parallel FTP Daemon Options

Option	Description
H	Used to disallow login for users whose home directory does not exist or is not properly configured. The default behavior (without the H option) is to put the user in the "/" directory.
I	Toggle the use of trusted hosts. Default is off. Note: this is not usually recommended.
K	Read the **hpss_option KTAB** from the **ftpaccess** file to obtain the {Path}/{Name} of the HPSS Keytab file containing the hpss_ftp principal. Default: **/krb5/hpss.keytabs**. Refer to documentation on the {ftpaccess} file for details.
L	Read the **hpss_option LS <string>** from the **ftpaccess** file to obtain the CDS name for the RPC group of Location Server. Refer to documentation on the **ftpaccess** file for details. This option is mandatory!
N	Read the **hpss_option SITE <string>** from the **ftpaccess** file to specify an alternate site. The **string** is the DCE cellname. This should rarely be necessary
P	Read the **hpss_option PRINC** principal from the **ftpaccess** file to obtain the DCE principal representing **hpss-ftp**.
R	Set the HPSS PFTP Daemon to use a specified port range for connections to the HPSS Movers for Parallel Transfer functions. The actual port range is specified in the Mover specific configuration record. Refer to Section 6.8.13: *Configure the PVR Specific Information* (page 373) for more information.
S	Use the DCE Registry for Authentication (bypassing the ftppasswd file). All HPSS FTP Customers MUST be in the DCE Registry since either the **hpss_XLoadThreadState**() or **hpss_LoadThreadState**{} call is invoked to obtain the end users identity.
T	Read the **hpss_option MTO <value>** option from the **ftpaccess** file to obtain the maximum timeout in seconds. Refer to documentation on the **ftpaccess** file for details.
U	Use UDP RPCs Only. Must be set! Sets the environment variable: **RPC_SUPPORTED_PROTSEQS=ncadg_ip_udp**. Unpredictable behavior and/or slow operation has been observed with DCE if this parameter is not set.
X	Use the DCE Registry for authentication (bypassing the ftppasswd file) and use the HPSS.homedir and HPSS.gecos Extended Registry Attributes (ERAs) for the users Home Directory and Accounting fields (if they are filled.)

The AIX (and others) inetd superdaemon will fail (without explanation or indication) if errors are encountered in the /etc/inetd.conf file. In particular, lines in excess of ~130 characters will cause this behavior to occur. It is recommended that HPSS Administrators utilize some form of "TCP wrapper" programs in conjunction with the inetd superdaemon (not provided with

HPSS.) Most of these applications, do not exhibit the line length limitation observed by the inetd superdaemon and they also allow "on the fly" modification of initialization parameters for network services; e.g., PFTP, telnet, etc., without having to refresh (kill -HUP) the inetd superdaemon.

The "**-D string**" option which will allow specification of the {**FTPBaseDir**} (Default: **/var/hpss/ftp**) and the "**-F string**" option which will specify the name of the **ftpaccess** file (Default: **ftpaccess** in the **{FTPBaseDir}/etc** directory.) The **ftpaccess** file will be in the directory specified by **{FTPBaseDir}/etc**. Default: **/var/hpss/ftp/etc**.

Two or more Parallel FTP Daemons (with different options) may be utilized by specifying a different {ftpaccess} file for each. This can be done by specifying different ports and using the **/etc/inetd.conf** file with multiple unique FTP entries or by using the same port but invoking a different option set based on the address(es) of the client (if the "TCP Wrapper" facilities are used.)

Step 2. Configuring the FTP Daemon Syslog

The FTP Daemon attempts to write to the system log (using **syslog()**). To enable this output, follow local system procedures for capturing this information. T he default syslog is **LOG_DAEMON**. This is modifiable to other options using the –s option on the startup line. NOTE the precautions above!

Step 3. Defining the FTP Access

The **{FTPBaseDir}/etc/{ftpaccess}** file defines: access, options, and restrictions for the server. An ftpaccess.template file is provided in the **$HPSS_DIR/config/template** directory for the HPSS Administrator to customize appropriately. The pound sign (#) at the beginning of a line is used to indicate a comment. Options are invoked by removing the comment symbol (#) and conversely disabled by introducing this symbol. Some options are specific to individual machines; therefore, each HPSS Parallel FTP Daemon should read its own {ftpaccess} file. It is important not to use NFS for the {**ftpaccess**} file if you use the "-h" option **(hpss_option HOST <interface name>**) in the HPSS PFTPD start options! If the HPSS PFTP Daemon specifies any of the "-L, -K, -P, or -h" options, the {**ftpaccess**} file must have the appropriate and correct **hpss_option** record or unpredictable failures may occur. Note that the -L option is mandatory.

The access options can be defined as follows:

```
# Define users as being guests (which means they cannot
# cd out of their own subdirectory).
guestgroup <group> [ <group> ... ]

# Set up user automatically into specified group.
# NOT CURRENTLY SUPPORTED.
autogroup <group> <class> [ <class> ... ]

# Set the maximum number of login attempts before closing the
# connection:
loginfails  <login_failure_threshold>

# Setup a private group file
# NOT CURRENTLY SUPPORTED.
```

```
private <private_group_pathname>

# Enable/Disable compression filter
# NOT CURRENTLY SUPPORTED.
compress [ yes | no ] <class> [ <class> ... ]

# Enable/Disable tar filter
# NOT CURRENTLY SUPPORTED.
tar [ yes | no ] <class> [ <class> ... ]

# Control for logging (sent to syslog()).
log [ commands ]  [ anonymous ]  [ { inbound } ]
[ transfers ]  [ guest ] [ { outbound } ] [ real ] [debug]

# Define a class of users, <user> is a full network address,
# including wildcarding, e.g., *.nsl.nersc.gov.
class  <class> { anonymous | guest | real } <user>
[ <user> ... ]

# Define a limit on the number of users from a certain class
# that can be using the FTP Daemon at specific times.
limit  <class> <n> <times> [ <msg_file> ]

# Deny access to a set of users, <addrglob> is full network address,
# including wildcarding, e.g., *.nsl.nersc.gov.
deny  <addrglob> [ <msg_file> ]

# Send shutdown message to current FTP users.
shutdown <shutdown_info_pathname>

# Send message to users, possible on access to a directory.
message <pathname> [ <when> ]

# Display prompt to read a file, possible on access to a directory.
readme <pathname> [ <when> ]

# Determine the appropriate behavior when needed data must be staged.
# <stage-code> can have several values:
# -1 wait forever
# 0 do not wait (default) - inform user that data needs staging and
#                           exit
# >0 wait for n seconds - if staging takes more than n seconds, inform
#                         the user and exit
# note: This option requires that the relevant storage class be
# configured in a compatible way, or behavior may not be as
# expected. For example, setting wait_on_stage to -1 (wait forever)
# for a SC with no-stage set would make little sense.
wait_on_stage <stage-code>

# Display a banner prior to the user prompt (before log in.) Very
# useful for informing the user that he/she is dealing with the
# HPSS file systems rather than the local file system of the
# host executing the FTP Daemon. This is highly recommended!
banner <pathname/filename>
```

The following “**hpss_options**” are read only if the corresponding flag (See the FTP Daemon Flags above) appears on the inetd.conf initialization line for the HPSS PFTP Daemon and may be left “active” (not commented out with the # symbol) even if the default value is desired.

```
# Define the Authentication Manager
hpss_option AMGR </opt/hpss/bin/auth_type>

# Define the default Blocksize in Bytes
hpss_option BUFSZ <value>

# Define the default timeout in Seconds
hpss_option DTO <value>

# Disallow HINTS processing, the default with this option is on
hpss_option HINTS off

# Define the Location Server RPC Group Entry (Mandatory)
hpss_option LS </.:/hpss/ls/group>

# Define an alternative Site Name
hpss_option SITE </.../dce.xyz.abc>

# Define the HPSS FTP Principal
hpss_option PRINC <hpss-ftp>

# Define the Maximum timeout in Seconds
hpss_option MTO <value>

# Define the default umask
hpss_option UMASK <value>

# Define the Keytab table name.
hpss_option KTAB <Path/Name of HPSS Keytab file containing the hpss_ftp
principal>

# Define the FTPD - Mover Network Interface to be used for
# non-parallel FTP transfers. Defaults to the interface associated
# with the “hostname” of the machine running the HPSS PFTP Daemon.
hpss_option HOST <myhost-fddi>
```

- **Message/readme/banner/shutdown** (message lines) are text files, with the following keywords (all one character in length) recognized, identified by a preceding **%**:

Table 7-2 Banner Keywords

Keyword	Description
M	Class limit
T	Current time on FTP Daemon
C	Current working directory
R	Remote hostname

Table 7-2 Banner Keywords

Keyword	Description
L	Local hostname
U	User name
s	Shutdown time
d	User disconnect time
r	Connection deny time
%	%

- The format of the **<shutdown_info_pathname>** file is:

  ```
  <year> <mon> <day> <hour> <min> <deny> <disc>
  ```

  Message lines contain keywords mentioned above. For example:

  ```
  1994 1 28 16 0 120 30
  System shutdown at %s (current time is %T)
  New FTP sessions denied at %r
  FTP users disconnected at %d
  ```

  indicates that shutdown will be 1/28/94 at 16:00, with users disconnected at 15:30 and sessions denied at 14:40 (the 120 indicates 1 hour, 20 minutes).

- The **<when>** clauses may be either **login** (in which case the file is displayed when the user logs in) or **cwd=[<directory_name> | *]** (in which case the file is displayed when the user changes to the named directory, with ***** wildcarding all directories).

- The **<times>** clause has the following format:

  ```
  <day><times>-<time> | <day><time>-<time> | . . .
  ```

  where **<day>** is one of **Su, Mo, Tu, We, Th, Fr, Sa, Su, Wk**, or **Any** and **<time>** is the time of day on a 24-hour clock. For example, to specify Tuesday between 8:00 am and 4:00 pm:

  ```
  Tu0800-1600
  ```

  The **Wk** entry for the **<day>** clause indicates a week day (Monday-Friday), and **Any** indicates all days of the week.

Step 4. Creating FTP Users

In order for an HPSS user to use FTP, a DCE **userid** and **password** must be created. Refer to Section 8.1.1: *Adding HPSS Users* (page 215) in the *HPSS Management Guide* for information on how to use the **hpssuser** utility to create the DCE **userid** and **password** and set up the necessary configuration for the user to use FTP. Note that this step should not be done until the NS and the BFS are running so that the **hpssuser** utility can create the home directory for the FTP user.

If desired (this is not recommended), the /bin/passwd_import and /bin/passwd_export utilities can be used to import/export the /etc/passwd file into/from the DCE Security Registry. However, caution should be used so that the /etc/passwd file is not overlaid. Also, note that the /bin/passwd_import and /bin/password_export utilities do not transfer the actual passwords in/out of DCE!

The **/opt/hpss/bin/hpss_ftppw** utility can be used to change the encrypted passwords in the / **var/hpss/ftp/etc/ftppasswd** file. The syntax for this utility is as follow:

```
hpss_ftppw <userid> [<password file pathname>]
```

The utility will prompt the user for the old and new passwords. The password file pathname argument can be used to specify a password file other than the default file, **/var/hpss/ftp/etc/ ftppasswd**.

If the HPSS PFTP Daemon utilizes the DCE Registry for authentication (**-S** or **-X** options), the {**ftpaccess**} file is superfluous and the rest of this section may be skipped! The HPSS home directory for the user must still be established and configured.

To enable anonymous FTP, the "**hpss_ftp**" user must be defined in either the HPSS FTP password file or in the DCE registry (depending on which authentication mechanism is enabled). In addition, the entry for the "**hpss_ftp**" user must contain a home directory defined to be a non-NULL value.

The home directory defined for the "hpss_ftp" user will be the root directory for the anonymous ftp session. The user will not be able to change out of the file tree with that directory as its root. Care must be taken if symlinks are created within this directory tree however - as it is possible that symlink will point out of this tree (and therefore allow an anonymous ftp user access outside of the directory tree).

To disable anonymous FTP, either:

1. Define the **hpss_ftp** user entry (in either the HPSS FTP password file or the DCE registry depending on which authentication mechanism is enabled) with a NULL home directory name, Set the shell for the **hpss_ftp** entry to "**/bin/FALSE**".

-and-

2. If the HPSS FTP password file is used for user authentication, do **not** define an entry for the "**hpss_ftp**" user.

or:

Add **hpss_ftp**, **anonymous** or **guest** to the HPSS FTP user file (normally "**/var/hpss/ftp/etc/ ftpusers**").

7.4 NFS Daemon Configuration

Before the HPSS NFS daemon can be started, any existing AIX or Solaris native NFS daemons must be stopped and prevented from restarting. This is important because the NFS protocol does not provide a way for clients to specify which of two daemons is wanted. When the system is set up correctly, there should be no 'nfsd' or 'mountd' processes running. If nfsd was running before, it may be necessary to free up the ports it allocated by using the 'rpcinfo -d' command. Otherwise the HPSS daemon may not start correctly. Also be sure to stop rpc.lockd and rpc.stad. The HPSS NFS daemon does not support file locking.

Additional HPSS NFS configuration information is specified through the HPSS **exports** file and through environment variables. If the default server name for the HPSS LS is not used, the environment variable for the LS, **HPSS_LS_NAME** should be changed in the **hpss_env** file. Similarly, if the default for the NFS Server descriptive name is not used, the environment variable **HPSS_NFS_DESC_NAME** should also be changed in the **hpss_env** file.

NFS allows you to traverse junctions, but be aware that certain functions may not behave as expected after a junction traversal (i.e. "`cd ..`"). By default, NFS junction traversal is disabled. To enable it, add **`HPSS_NFS_ENABLE_JUNCTIONS = "on"`** to the **hpss_env** file before bringing up the Startup Daemon. This environment setting will also allow junctions to be mounted (i.e. "**`mount hostname:/junction`**").

HPSS NFS does not support yellow pages (Sun MicroSystems' Network Information Services) to validate hosts. HPSS NFS does provide an option to validate the network address of hosts attempting to mount HPSS directories. The default configuration disables this check. To enable this check, define the variable **HPSS_MOUNTD_IPCHECK** in the **hpss_env** file.

Note: For users who have several NFS clients concurrently updating and reading the HPSS namespace and depend on having their namespace changes immediately reflected in all clients, it is necessary to perform NFS mounts with the noac option. Here is an example:

```
mount -orw,intr,timeo=30,noac hpssserver:/hpss /hpss
```

Using the **noac** option will degrade performance since it prohibits caching. Only use it if NFS clients rely on immediate HPSS namespace updates.

If people are using NFS to mount file systems from several hosts, a situation can develop where one NFS daemon gets bogged down and then users can't access the other file systems either. This is a common problem with all NFS file systems and is not peculiar to HPSS. However because the HPSS NFS daemon is prone to bogging down, it is especially likely to affect other file systems.

To avoid this problem, sites may want to advise their users on the best way to set up their NFS mount points. All of the mount points for a particular NFS host should be kept in a directory separate from the other hosts. (This is probably a good idea even if the clients are not mounting HPSS files.) One convention for doing this is to name the mount points by the name of the host that is exporting the file system. For example, if a host named **tardis** is exporting HPSS directories **/home** and **/public**, and a host named **jupiter** is exporting the Unix directory **/home**, the mount points might be: **/nfs/tardis/home**, **/nfs/tardis/public**, and **/nfs/jupiter/home**

If these names aren't considered user friendly, the site can use symlinks to establish friendlier names. For example, **/hpss** could point to **/nfs/tardis**, and **/users** could point to **/nfs/jupiter/home**.

The alternative approach would have been to mount **jupiter**'s files directly on **/users** and **tardis**' files directly on **/hpss**. But this is the layout of mount points that should be avoided; it can cause the two NFS daemons to interact badly with each other.

7.4.1 The HPSS Exports File

The HPSS exports file contains a list of HPSS directories and filesets that can be exported to the NFS clients. If this file is changed, the NFS and mount daemons must be restarted before the changes will take effect. For best results, the daemons should be restarted soon after the changes are made.

The **exports** file contains one line for each directory or fileset being exported. The format of an export line is:

```
[Fileset]:Directory -Option[,Option]...
```

where the fields are as follows:

- **Fileset**: Specifies an optional fileset name. If this field is missing, the root fileset is assumed and the **Directory** field refers a directory in that fileset. If the field is present, the directory refers to the named fileset.
- **Directory**: Specifies the complete HPSS directory pathname relative to the fileset given by Fileset.
- **Option**: Specifies optional characteristics for the directory being exported. More than one variable can be entered by separating them with commas. Choose from the following options:

Table 7-3 Directory Export Options

Option	Description
id=Export-id	Integer between 1 and 255 that specifies a unique identifier for this export entry. This option is required. If duplicate identifiers are found in the hpss exports file, an error message is logged and only the first entry will be included in the exports list.
ro	Exports the HPSS directory with read-only permission. Otherwise, if not specified, the directory is exported with read-write permission.
rw=Client[:Client]	Exports the HPSS directory with read-write per- mission to the machines/networks specified by the Client parameter and read-only to all others. The Client parameter can be either the host name or the network name. Network names are specified in the local system / etc/networks file. If a Client is not specified, the directory is exported with read-write permission to all.

Table 7-3 Directory Export Options

Option	Description
anon=UID	If a request comes from a client user with the HPSS root identity, use the UID value as the effective user ID. The default value for this option is -2.
root=HostName[:HostName,...]	Gives HPSS root access only to the HPSS root users from the specified host name. The default is for no hosts to be granted root access. Network names are not allowed for this option.
access=Client[:Client,...]	Gives mount access to each client listed. A client can be either a hostname or a network name. Each client in the list is first checked in the host's database and then in the /etc/networks file. The default value allows any machine to mount the given directory.
NOSUID	Causes the NFS server to mask off **setuid mode** bits. The default is for **setuid mode** bits to be allowed.
NOSGID	Causes the NFS server to mask off **setgid mode** bits. The default is for **setgid mode** bits to be allowed.
UIDMAP	Causes the NFS server to require an entry in the NFS credentials map cache for all non-mount related requests. The default is to not require UID mapping.
key=Key	Specifies an alpha numeric string up to 8 characters that will be used as an encryption key when the mount and NFS servers are configured with Encrypt File Handles enabled. This option is required if Encrypt File Handles is enabled. The default is for no key.

A # (pound sign) anywhere in the HPSS exports file indicates a comment that extends to the end of the line.

7.4.2 Examples

1. To export the HPSS directory **/usr** to network sandia.gov, enter:

```
/usr -id=1,access=sandia.gov
```

2. To export the HPSS **/usr/local** directory to the world, enter:

```
/usr/local -id =4
```

3. To export the HPSS directory **/usr2** only to these systems, enter:

```
/usr2 -id=5,access=hermes:zip:tutorial
```

4. To give root access only to these systems, enter:

```
/usr/tps -id=20,root=hermes:zip
```

5. To convert client root users to guest UID=100, enter:

```
/usr/new -id=10,anon=100
```

6. To export read-only to everyone, enter:

```
/usr/bin -id=15,ro
```

7. To allow several options on one line, enter:

```
/usr/stuff -id=255,access=zip,anon=-3,ro
```

8. To export files in a directory named **/sources** in a fileset named **project.fileset**:

```
project.fileset:/sources -id=20,access=sandia.gov
```

7.4.3 Files

/var/hpss/nfs/rmtab	Lists all currently mounted HPSS directories.
/etc/hosts	Contains an entry for each host on the network.
/etc/networks	Contains information about subnetworks on the network.

7.5 MPI-IO API Configuration

MPI-IO is not enabled in the install image distribution as it must be configured specifically for each site's host MPI. The HPSS administrator must determine if MPI-IO support is desired and, if so, which host MPI is to be used. A site-specific MPI-IO must be generated using make in the src/mpi directory, after the appropriate configuration changes have been made in Makefile.macros.

To enable MPI-IO, change the **MPIO_SUPPORT** in **Makefile.macros** to **on**. MPI-IO will be configured to match the **MPIO_MPI** setting in the **Makefile.macros**, which selects the host MPI. The HPSS administrator should choose the appropriate configuration to use when building the MPI-IO subsystem. To build the MPI-IO library, **cd** to the **$HPSS_DIR/src/mpi** and execute **make**. This will generate an appropriate **$HPSS_DIR/include/mpio_MPI_config.h**.

Only one MPI host is supported for each HPSS system. To change the MPI host, the administrator must edit the **Makefile.macros** to change the MPIO_MPI setting, and generate a new **include/mpio_MPI_config.h** file by doing a **make clean** followed by a **make** in the **src/mpi** directory.

If the MPI host version is updated, HPSS MPI-IO should be reconfigured and remade, to be sure any changes to the host interfaces are accurate. The administrator should follow the same instructions as above for changing the MPI host.

We currently support MPI-IO API for the following host MPIs:

- IBM's Parallel Operating Environment MPI, Version 3 Release 2
- Sun HPC MPI, version 4.1
- ANL MPICH, version 1.2

Other versions of MPI may be compatible with HPSS MPI-IO as well. The **mpio_MPI_config.h** file is dynamically generated from the host MPI's **mpi.h** file, making it possible to tailor the interaction of HPSS MPI-IO with the host MPI. Earlier versions of these MPI hosts are known to be compatible with HPSS MPI-IO, for example, but we recommend the use of the above versions for this release of HPSS.

This configuration results in the construction of the file **include/mpio_MPI_config.h.** Applications in C must be compiled with **`#include "mpio.h"`** which will also include the **mpio_MPI_config.h** to determine the host interface requirements. Fortran applications must include **mpiof.h** and C++ applications are further required to include **mpio.h** *before* including the host **mpi.h**

HPSS MPI-IO supports the Fortran77 and C++ interfaces for the MPI-IO API, as specified in the MPI-2 standard. These interfaces require compatible Fortran77 and C++ compilers as described in Section 2.3.1.6: *Miscellaneous* on page 49. The **Makefile.macros** allows tailoring of the MPI-IO environment to selectively disable support for Fortran or C++, if compatible compilers are not available, by setting one or both of **MPIO_CPLUSPLUS_SUPPORT** and **MPIO_FORTRAN_SUPPORT** to **off**.

The following environment variables can also be used to amend the MPI-IO API configuration:

The **MPIO_LOGIN_NAME** defines the DCE login name of the principal who will be executing the application. If specified along with **MPIO_KEYTAB_PATH**, which specifies the path name to a file containing the principal's security keys, each process in a distributed application will be able to create the credentials necessary for DCE authentication to HPSS. If either is not specified, the default is to use the DCE credentials for the current login session, if any.

The **MPIO_DEBUG** can be toggled to direct MPI-IO to give messages when errors are detected. A value of 0 (zero) will disable message reporting, and any nonzero value will enable reporting.

In addition to the MPI-IO-specific variables, any of the HPSS Client API variables can be used (see Section 7.1: *Client API Configuration* on page 413). If using MPI-IO with the non-DCE API, the **HPSS_NDCG_SERVERS** environment variable must be appropriately set, as described in Section 7.2.2: *Environment Variables* on page 416.

7.6 HDM Configuration

7.6.1 Introduction

HPSS provides distributed file system services by optionally allowing HPSS to interface with Transarc's DFS™. DFS is a scalable distributed file system that provides a uniform view of file data to all users through a global name space. DFS supports directory hierarchies, and an administrative entity called filesets. DFS also supports ACLs on both directories and files to allow granting different permissions to different users and groups accessing an object. DFS uses the DCE concept of cells to allow data access and authorization between clients and servers in different cells. DFS uses the Episode™ physical file system, although it can use other native file systems, such as UFS.

HPSS provides a similar interface through the Linux version of SGI's XFS file system.

A standard interface is used to couple DFS and XFS (the "managed" file systems) with HPSS. The integrated system provides transparent, automatic archiving and caching of data between the managed file system and HPSS. For DFS systems, HPSS supports partially resident file data in both DFS and HPSS and allows changes made to a file or directory through DFS interfaces to be visible through HPSS interfaces and vice versa. For XFS systems, access is available only via XFS interfaces, and file data is either wholly resident or wholly absent.

The standard selected to couple DFS and XFS with HPSS is The Open Group's Data Storage Management standard (XDSM). The API provided for in this standard is called the Data Management API (DMAPI), and the two acronyms, 'XDSM' and 'DMAPI', are often used interchangably. The XDSM standard originated in the mid-nineties from the Data Management Interfaces Group (DMIG). It is a low-level interface to the physical file system which provides the Data Management application (DMAP) with the ability to store important attribute information with a file. It also allows for the generation of notifications to the DMAP on occurrence of various file system operations. XDSM enables the DMAP to control disk storage by allowing the DMAP to move disk-resident file data to tertiary storage systems and vice-versa.

7.6.2 Filesets

DFS supports logical collections of files called *filesets*. A fileset is a directory subtree, administered as a unit, that can be mounted in the global name space. Multiple DFS filesets may reside on an *aggregate*, which is analogous to a disk partition. Filesets may be moved between DFS aggregates, either on the same or different servers to achieve load balancing.

XFS supports the classic filesystem which resides on and fills an assigned disk partition.

For the HPSS/DFS interface, two data management configuration options, archived filesets and mirrored filesets, are available which differ in the way the fileset is managed by HPSS.

For the HPSS/XFS interface, only the archived configuration option is supported.

7.6.2.1 Archived Filesets - (DFS or XFS)

HPSS is used strictly as an archive facility for the managed filesystem. Access to the name and data space is provided only through the managed file system's interfaces. Filesets and filesystems managed with this option are called 'archived filesets'. The files in an archived fileset contain copies of file data from any managed filesystem files that have migrated to HPSS. Pathnames to these HPSS files are generated by the HPSS Data Management Application (HDM) based on configuration policies. HPSS root may access objects in these filesets, but all client access is via the managed filesystem.

7.6.2.2 Mirrored Filesets - (DFS only)

Consistency between HPSS and DFS name and data spaces is maintained. DFS data is archived to HPSS. Access to the name and data space is available through both DFS and HPSS interfaces, with updates made through the DFS interface being visible through the HPSS interface and vice versa. Filesets managed with this option are called mirrored filesets. Objects in mirrored filesets have corresponding entries in both DFS and HPSS with identical names and attributes. A user may access data through DFS, at standard DFS rates, or when high performance I/O rates are important, use the HPSS interface.

7.6.2.3 Architectural Overview

To interface DFS with HPSS, modifications to both DFS and HPSS were required. The architecture used to integrate HPSS with the Episode file system is shown in Figure 7-1. The architecture to integrate HPSS with XFS is very similar. It is actually identical insofar as HPSS components are concerned; however, the 'DFS SMT', 'DFS File Server', 'DFS Client', and 'Episode' components in Figure 7-1 would be replaced with a single entity, 'XFS File System'.A brief description of the modifications and new components are discussed in the following sections.

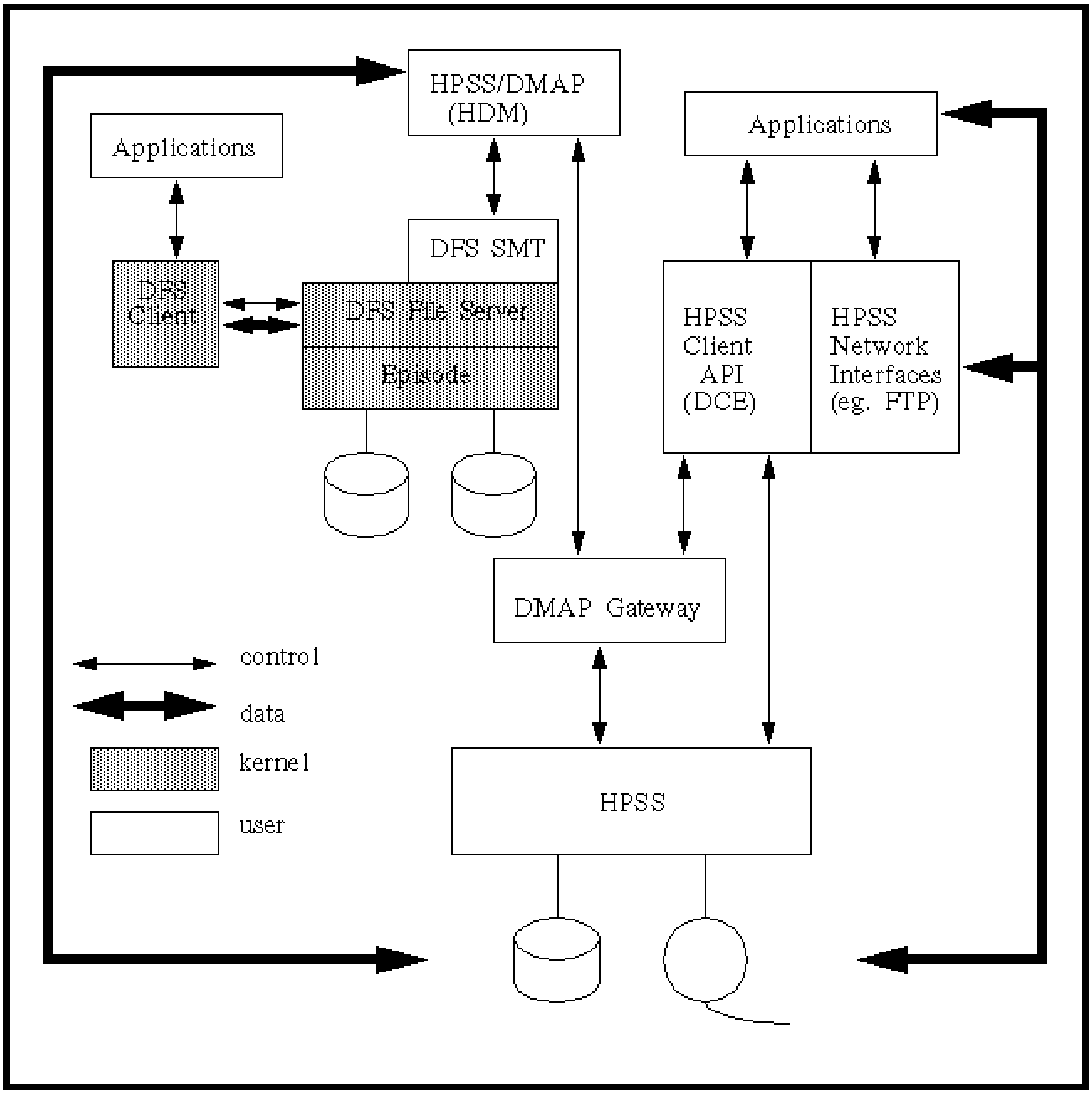


Figure 7-1 DFS / HPSS XDSM Architecture

7.6.2.4 XDSM Implementation for DFS

The XDSM implementation supported by Transarc is called the DFS Storage Management Toolkit (DFS SMT). It is fully compliant with the corresponding standard XDSM specification. In addition, it provides optional features: persistent opaque Data Management (DM) attributes, persistent event masks, persistent managed regions, non-blocking lock upgrades and the ability to scan for objects with a particular DM attribute.

The bulk of DFS SMT is implemented in the DFS file server, but there is also a user space shared library that implements all APIs in the XDSM specification. The kernel component maintains XDSM sessions, XDSM tokens, event queues, and the metadata which describes the events for which various file systems have registered. The kernel component is also responsible for receiving events and dispatching them to the DMAP.

The DFS File Server fetches and stores DM attributes, provides persistent managed regions and events, perform invisible I/O, purges data from files, and verifies file residency. This component determines if a DMAP has registered to receive a notification for an event related to a particular operation, then generates the event. If the event is synchronous, it causes the file system operation to wait for a response to the event before proceeding. It also provides for interlocking between DFS SMT requests and file system calls.

To support persistent DM related metadata, the Episode File System has an extended attribute facility. DM attributes, event masks, managed regions, and attribute change times (dtime values) are stored as extended attributes. These extended attributes are treated as file metadata. Changes to extended attributes are logged.

Episode was also modified to support files that become sparse by punching holes that release disk resources. With a conventional sparse file, reading from a hole returns zeroes. To assume these same semantics for a hole that exists because the DMAP migrated the data to tertiary storage would be incorrect. In this case, the DMAP must retrieve the data from tertiary storage. Hence, Episode marks these file blocks as being off-line (in tertiary store) instead of as a hole. This allows the file server to handle partially resident files.

To prevent blocking file server (kernel) threads while waiting for a response to an event, Episode has a mechanism to notify the client to retry after a specified interval of time.

The DFS fileset dump and restore capability includes extended attributes and migrated regions. Migrated data is not recalled when a dump is taken, producing an abbreviated dump.

The current XDSM release for Episode does not support cloning. Consequently, some fileset commands, such as, fts clone, fts move, etc., will not work for XDSM enabled DFS. As a result fileset backups that rely on cloning must be done with fts dump.

7.6.2.5 XDSM Implementation for XFS

The XDSM implementation supported by SGI for Linux XFS is compliant with the corresponding standard XDSM specification. In addition, it provides optional features: persistent opaque Data Management (DM) attributes, persistent event masks, and persistent managed regions.

There is a user space shared library that implements all APIs in the XDSM specification. A kernel component maintains XDSM sessions, XDSM tokens, event queues, and the metadata which describes the events for which various file systems have registered. The kernel component is also responsible for receiving events and dispatching them to the DMAP.

XFS fetches and stores DM attributes, provides persistent managed regions and events, performs invisible I/O, purges data from files, and verifies file residency. It determines if a DMAP has registered to receive a notification for an event related to a particular operation, then generates the event. If the event is synchronous, it causes the file system operation to wait for a response to the event before proceeding.

To support persistent DM-related metadata, XFS utilizes its standard extended attribute facility. DM attributes, event masks, managed regions, and attribute change times (dtime values) are stored as extended attributes. These extended attributes are treated as file metadata.

The **xfsdump** and **xfsrestore** utilities include extended attributes and migrated regions. Migrated data is not recalled when a dump is taken, producing an abbreviated dump.

7.6.2.6 HPSS Components Required to Support DFS and XFS

An HPSS Data Management Application (HPSS / DMAP or HDM) and a DMAP Gateway must be configured into the HPSS system if an integrated HPSS / DFS or HPSS / XFS system is to be supported at your site.

7.6.2.6.1 HPSS/DMAP (HDM) Server

HDM is responsible for initiating and coordinating name space and data interactions between the managed file system and HPSS. It catches and processes desired name and data space events generated by the managed file system, migrates file data to HPSS, purges unneeded file data from the managed file system after migration, and processes requests originating in HPSS interfaces that require either name or data resources from the managed file system. HDM resides on the same machine where the managed file system is running. HDM communicates with HPSS components via XDR over TCP sockets.

HDM registers to receive name and data space events originating in the managed file system. After catching a name space event involving a mirrored fileset, the appropriate requests are made to HPSS to keep the name spaces synchronized. However, for archived filesets, only create and destroy name space events are processed, but no HPSS resources are utilized until the file is migrated to HPSS. HDM receives data space events when a file is read, written, or truncated and the data region involved is registered via XDSM. HDM is responsible for caching data to the managed file system that is not present; invalidating HPSS data, if necessary; and manipulating data regions to minimize the occurrence of events involving the same region. HDM can cache partial files with DFS but must cache whole files for XFS due to a constraint on the number of managed regions supported by XFS.

HDM provides an interface for HPSS to request that an action occur in DFS to keep the HPSS and DFS name and data spaces synchronized. This mechanism is only used with mirrored filesets when an HPSS client requests to create, delete or modify a name space object. This interface is also used by HPSS to migrate or purge data from Episode disks. Before file data can be altered through HPSS interfaces, the data must first be purged from Episode disks. The capability to forward DCE credentials is provided, enabling HDM to make DFS requests on behalf of the end user.

HDM migrates file data from the managed file system to HPSS. It also purges file data from the managed file system to free disk resources. For DFS, only data that has been modified on the managed file system is migrated to HPSS, thus, a minor modification to a very large file will not result in re-migrating the entire file. Because policies for migrating and purging data are separately configurable, file data migrated from a managed file system is not automatically purged. In many cases, data for a given file will be present in both the managed file system and HPSS and can be read from either interface without any data migration or staging.

7.6.2.6.2 DMAP Gateway Server

The DMAP Gateway is a conduit and a translator between HDM and HPSS. HPSS servers use DCE/RPCs to communicate, however the DMAP Gateway encodes requests using XDR and sends these requests via sockets to HDM. In addition, it translates XDR from the HDM to DCE/TRPC/Encina calls to the appropriate HPSS servers. When a connection between the HDM and Gateway is made, mutual authentication occurs.

The DMAP Gateway keeps a record of all the managed filesystems and HPSS filesets it manages. For scalability, multiple DMAP Gateways are supported. However, a given DMAP Gateway will only operate on the filesets it manages. At this time, the DMAP Gateway only supports filesets managed by DFS and filesystems managed by XFS. Additionally, a DMAP Gateway can only interact with the Name Server(s) within its HPSS site.

In addition to translating requests between HDM and HPSS servers, the DMAP Gateway keeps fileset request statistics and internal DMAP Gateway resource utilization statistics. Thus, heavily used filesets can be identified and management of these filesets could be distributed across multiple DMAP Gateways to improve performance.

7.6.2.7 HPSS/DFS Usage Guide and Limitations

This section of the document provides information to help system administrators determine the best way to use the HPSS/DFS interface. Both DFS and HPSS have limitations that will impact performance when using the HPSS/DFS interface. Some of these limitations are the result of implementation decisions, others are hardware dependent. This section will address these limitations, explain why they exist, when and if the limitation will change, and how to configure your system to minimize the impact of these limitations.

7.6.2.7.1 Fileset Anode Limitations

The current implementation of Episode allows a maximum of 2GB of space per fileset for anode and file attribute storage. This space includes anodes (file headers), space allocation, ACL's, and space used to keep track of DMAPI attributes (regions and extended attributes). In a DFS fileset, that is not also managed by HPSS, this provides storage space for approximately 4 million files. The number of files in a fileset is greatly reduced when the fileset is also managed by HPSS.

The information stored to keep track of migration generally uses an extra fragment of anode space. Since we recommend using a fragment size equivalent to the block size (8K), each file migrated into HPSS will use a little more than 8K (assuming no extra ACL space). This puts the upper limit on the number of files per fileset at about 250,000.

This limit may change in the future. If the fragment size can be reduced to 1K then the number of files increases to about 1.5 million. However, using a 1K fragment size is not currently recommended for two reasons:

1. It is possible for an aggregate to appear to have plenty of space but that space could be allocated as fragments that cannot be used for files greater than the block size. This leads to problems when trying to determine how much data needs to be purged to free up DFS space.

2. Fragments can not be purged. Only files using blocks can be purged.

Meetings with Transarc and IBM Austin have taken place to discuss the issue. The above restriction may be fixed in the future.

7.6.2.7.2 *Migration and Purge Algorithms*

Currently, the HDM reads through all the anodes in an aggregate to determine migration and purge candidates. Using empirical data we determined that the HDM reads approximately 70 entries per second (this is disk hardware dependent). The optimum number of objects stored on an aggregate for migration purposes can be determined using the following parameters:

- The amount of time it takes to read through all objects on the aggregate.
- The frequency of migration runs (purge runs when space is needed).
- The amount of time it takes to move files from DFS to HPSS.

If migration runs once per day, and data can be moved in 4 hours, the current suggestion is to allow approximately 500,000 objects on an aggregate. This means that migration will take approximately 6 hours (7142 seconds = ~2 hours for header rumble + 4 hours for data movement). Obviously, these numbers vary dramatically and are dependent on file size and the number of files that need to be migrated. Remember, only new files and files whose data has been altered need to be migrated into HPSS. Files that have already been migrated into HPSS, but have been read need to update XDSM attributes, but this can be done quickly (approximately 60 per second).

Episode XDSM(DMAPI) changes have been discussed between HPSS and Transarc. It is believed that with these changes, the Episode XDSM(DMAPI) rate to read all objects in an aggregate would be increased to approximately 400 per second. This would greatly reduce the time taken to determine migration and purge candidates. Then the suggested limit for the number of objects on an aggregate could be raised to 2 million objects.

7.6.2.7.3 *Fileset Quotas*

The Episode fileset quota is currently stored in a 32-bit field (actually only 2^31 – 1 can be stored). The quota represents 1K allocations. So, the current maximum quota per fileset is about 2TB. The Episode quota reflects the amount of data stored in the fileset plus the amount of anode space. The quota can not be increased beyond 2TB. When the quota limit is reached the only way to store more files is to remove old files.

When the HDM purges data from Episode the quota is not affected. The quota reflects all data ever stored through the DFS interface. If a file is written to a mirrored fileset through the HPSS interfaces, the quota is not altered (except for anode space) until the data is actually read through the DFS interface. So, it is possible to write more than 2TB to a mirrored fileset, but only if the data is written through a non-DFS interface and never read through the DFS interface (This may occur if large data dumps are made directly to HPSS, but the client wishes to use DFS to manage the name space).

There are no current plans to change the Episode fileset quota implementation.

7.6.2.7.4 Mirrored Fileset Recovery Speed

Mirrored fileset recovery time must be considered when configuring the system. Mirrored fileset recovery can be tedious and slow. The DFS fileset is recovered using HPSS metadata, which requires many SFS accesses.

The three time consuming steps when recovering mirrored filesets are:

- Dumping the HPSS fileset information. The rate for this step is approximately 60 entries per second.
- Restoring the fileset to Episode. The rate for this step is approximately 60 entries per second.
- Loading Episode DMAPI handles into the HPSS metadata. This is by far the most time consuming step, and is highly dependent on SFS speeds. The rate for this step is approximately 12 entries per second.

The above rates can be used to approximate the recovery times for mirrored filesets. For example, the recovery time for a mirrored fileset with 100,000 objects is approximately 3.2 hours (1666 seconds for metadata dump + 1666 seconds for Episode fileset recovery + 8333 seconds for HPSS metadata updates = 11665 seconds = ~3.2 hours).

So, it can be seen that recovery of a lost Episode fileset can take a very long time. If multiple filesets need to be recovered, many of these steps can be done in parallel, but disk and SFS limitations will quickly outweigh any amount of parallelism.

7.6.2.7.5 Cloning, Replication, and Fileset Backup

Aggregates managed by the Episode XDSM (DMAPI) do not support cloning. Fileset replication and movement can not be done because they require cloning. DFS recommends the use of cloning to backup filesets. These mechanisms cannot be used with HPSS/DFS filesets.

HPSS/DFS filesets that are mirrored do not have to be backed up since they can be restored from HPSS metadata. The restored mirrored fileset will only be able to restore data that HPSS knows about. Any file deleted by mistake cannot be recovered on mirrored filesets. Users should be made aware of this fact and the proper precautions taken.

HPSS/DFS archived filesets must be backed up because not all data is migrated into HPSS and the name space information is not recoverable from HPSS metadata. The only way to back up archived filesets is with the DFS utility '**`fts dump`**'. The problem is, the fileset is locked for updates while the dump is taken. Filesets with a large amount of objects and data may be locked from updates for a long period of time.

7.6.2.8 HPSS/XFS Usage Guide and Limitations

This section of the document provides information to help system administrators determine the best way to use the HPSS/XFS interface. Both XFS and HPSS have limitations that will impact performance when using the HPSS/XFS interface. Some of these limitations are the result of implementation decisions, others are hardware dependent. This section will address these limitations, explain why they exist, when and if the limitation will change, and how to configure your system to minimize the impact of these limitations.

7.6.2.8.1 Migration/Purge Algorithms and the MPQueue

Migration and purge are handled differently in an HPSS / XFS system than in an HPSS / DFS system. In an HPSS / XFS system, a queue of migration and purge candidates are kept in shared memory using the Migration / Purge Queue or MPQueue. When migration and purge run, they simply step through the migration and purge candidates in the MPQueue. This eliminates the need to query every file in the filesystem for each migration and purge run and will result in much more expedient migration and purge runs than for a comparable DFS filesystem.

However, since the MPQueue resides in shared memory it must have a fixed size. This size is decided when the filesystem is initially configured and should be planned carefully, taking into consideration the migration and purge policies to be used and expected filesystem activity. It will be necessary to have one MPQueue entry available for each unmigrated file as well as each file that has been migrated but not yet purged. In other words, any file which is not completely migrated and purged from XFS will take up an entry in the MPQueue. It is recommended to use an MPQueue size that is at least twice the maximum number of “active” files expected for the file system.

7.6.2.9 Other Limitations

These limitations apply equally to HPSS / DFS and HPSS / XFS systems.

7.6.2.9.1 HPSS/DFS Activity

The system administrator must consider what level of HDM / DMG activity the core HPSS servers can support. HPSS speed is limited by hardware, SFS, disk and tape speeds. If HDM / DMG activity causes millions of HPSS file creations and file migrations into HPSS, then the system may have problems.

Users of managed filesystems should be educated about the limitations of such configurations. Though the HDM / DMG interface allows for normal Unix type file activity, it can not be forgotten that these interfaces access data from the archive. The HDM is implemented to migrate individual files into HPSS, so there is a potential that each file access may incur a tape delay when the file is read back from HPSS. If an HDM / DMG user stores related data into larger files, the total number of tape accesses can be decreased, increasing the user’s performance. This will also decrease the load on HPSS.

Users and administrators should be educated in the use of managed filesystems, so that informed decisions can be made about their use. A given fileset type does not optimally handle all types of file usage patterns. Standard DFS and XFS filesystems run out of disk space. Archived filesets incur delays for data retrieval if the data has to be staged back to the managed filesystem. HPSS / DFS mirrored filesets incur delays for data retrieval and also are slower for name space updates, but they permit access to the data and name spaces through all HPSS interfaces.

7.6.3 Configuration

For AIX systems, it is assumed that the system on which the DFS Server is running has been configured with the appropriate DCE and DFS versions and PTFs as given in Section 2.3.2.1: *HPSS Server/Mover Machine - AIX* on page 50.

For Linux systems, it is assumed that the system on which XFS is running has been configured with the appropriate kernel and XFS versions as given in Section 2.3.5.1: *HPSS/XFS HDM Machine* on page 52.

For DFS HDMs, two additional steps must be performed before continuing to the configuration of the HDM:

1. Configure DFS SMT Kernel Extensions (AIX)
2. Configure DCE DFS

The following sections describe these extra steps in more detail. For XFS HDMs, skip ahead to Section 7.6.3.3: *Configuring an HDM Server (DFS & XFS)* on page 448.

7.6.3.1 Configuring DFS SMT Kernel Extensions on AIX (DFS Only)

The DFS SMT kernel extension software should be configured to use the shortest possible timeout parameter for **delay**. This parameter determines the interval at which the kernel backs off when HDM response is slow, which can happen if a file is being staged to Episode from HPSS and the data is not yet available on Episode. The timeout parameter is set with **-delay**. The default value should not be used, since it can cause the system to take up to 17 minutes to process a single end-user request. It is recommended that **delay** be set to 1. The parameter is expressed as an exponent that is applied to a base of 4 seconds. Hence, setting the parameter to 1 causes a 4 second delay. Configuring this parameter is described in the next section.

7.6.3.2 Configuring DCE DFS (DFS Only)

To avoid data and name space inconsistencies, the HDM should be started before the aggregates it manages are exported. The safest way to ensure this is to modify the startup scripts.

The first thing to do is to create the following new scripts:

```
/opt/dcelocal/tcl/user_cmd.tcl
/var/hpss/hdm/hdm1/pre_start_dfs
/var/hpss/hdm/hdm1/pre_stop_dfs
/var/hpss/hdm/hdm1/post_stop_dfs
```

The new scripts take care of loading the DFS SMT Kernel extensions and starting the HDM. For purposes of discussion, the last three scripts are assumed to be in **/var/hpss/hdm/hdm1**; but any suitable directory can be used as long as **user_cmd.tcl** has been set up to point to that directory.

The example scripts below assume that a site runs more than one HDM on a machine, and that these HDMs obey certain conventions. The scripts will have to be modified if different conventions are used, and can be simplified on machines that have only one HDM. The following convention makes it particularly easy to write the scripts and to use **hdm_admin** to administer the resulting system. In this convention, an HDM whose **ServerID** parameter is **<N>** works with data files in the directory named **/var/hpss/hdm/hdm<N>** and uses a shared memory key given by **3788+<N>**.

For more information on the **ServerID** and shared memory key concepts, refer to Section 7.6.3.3. For more information on hdm_admin, refer to Section 6.6: *Using hdm_admin* (page 108) in the *HPSS Management Guide.*

Here is a sample **user_cmd.tcl**:

```
#!/bin/ksh
set pre_start_dfs   "/var/hpss/hdm/hdm1/pre_start_dfs"
set pre_start_dfs_fail_on_error $TRUE
set pre_stop_dfs    "/var/hpss/hdm/hdm1/pre_stop_dfs"
set post_stop_dfs   "/var/hpss/hdm/hdm1/post_stop_dfs"
```

The **pre_start_dfs** Korn shell script will break before trying to start DFS and export DFS files. The script ensures that the HDMs are all running. If there is any problem doing that DFS will not be started, giving the system administrator a chance to fix the problem.

If this script needs to be modified, it is important to make sure that incidental messages that would normally be written to stderr get rerouted to stdout or **/dev/null**. Otherwise the TCL procedures that call **pre_start_dfs** will assume the script has had an error, even if the script eventually calls exit 0. It is also important to redirect all output from the **hpss_hdm** command; otherwise TCL will wait for the HDM to stop before going ahead with the startup, with the result that DFS will never start.

In this script, be sure to set the delay parameter on **cfgdmepi** to 1. This parameter controls the maximum delay time between client retries after an operation fails. In general, a delay parameter of N causes a maximum delay interval of 4 raised to the Nth power. If N is zero, retries will be done once a second which may cause the system to thrash. If N is omitted, the default value of 5 will be used, which can result in delay times as long 1024 seconds (roughly 17 minutes).

7.6.3.2.1 DFS Configuration Scripts

The **pre_start_dfs** script is executed before DFS is started.

An example of the **pre_start_dfs** for AIX is as follows:

```
#!/bin/ksh
export HPSS_PATH_BIN=/opt/hpss/bin
echo "   loading dfscore, dfssvr, and dcelfs"
/usr/sbin/cfgdfs -a /usr/lib/drivers/dfscore.ext
/usr/sbin/cfgdfs -a /usr/lib/drivers/dfssvr.ext
/usr/sbin/cfgdfs -a /usr/lib/drivers/dcelfs.ext
echo "   configuring dmepi"
/usr/sbin/cfgdmepi -delay 1 -a /usr/lib/drivers/dmlfs.ext
if [ $? != 0 ]; then
  exit 1
fi
# Start the servers (two of them in this example):
for id in 0 1; do
  key=`expr 3788 + $id`
  var=/var/hpss/hdm/hdm$id
  $HPSS_PATH_BIN/hdm_admin -k $key -s $id -v $var ps >/dev/null 2>&1
  if [ $? != 0 ]; then
    echo "   starting hdm$id"
    rm -f $var/hdm.out
    $HPSS_PATH_BIN/hpss_hdm $var/config.dat $id > $var/hdm.out 2>&1
    status=$?
    if [ $status != 0 ]; then
```

```
        echo "   could not start hdm$id, status = $status"
        exit $status
      fi
    else
      echo "   hdm$id is already running"
    fi
  done
  echo "   all hdm servers are running"
  exit 0
```

An example of the **pre_start_dfs** for Solaris is as follows:

```
#!/bin/ksh
# Start the servers (two of them in this example):
for id in 0 1; do
  key=`expr 3788 + $id`
  var=/var/hpss/hdm/hdm$id
  $HPSS_PATH_BIN/hdm_admin -k $key -s $id -v $var ps >/dev/null 2>&1
  if [ $? != 0 ]; then
    echo "   starting hdm$id"
    rm -f $var/hdm.out
    $HPSS_PATH_BIN/hpss_hdm $var/config.dat $id > $var/hdm.out 2>&1
    status=$?
    if [ $status != 0 ]; then
      echo "   could not start hdm$id, status = $status"
      exit $status
    fi
  else
    echo "   hdm$id is already running"
  fi
done
echo "   all hdm servers are running"
exit 0
```

The **pre_stop_dfs** script is executed just before shutting down DFS. As part of the shutdown procedure, AIX will try to unexport or "detach" all DFS aggregates. This ensures that they are left in a consistent state and don't need to be salvaged before they can be used again. In order to detach an aggregate, it is important that the HDMs be running first, so this script tends to that. The script also ensures that no filesets are locally mounted, which would prevent the aggregates from being detached. As with the other scripts, incidental messages to stderr should be rerouted to stdout.

Here is sample for **pre_stop_dfs**:

```
#!/bin/ksh
export HPSS_PATH_BIN=/opt/hpss/bin
for id in 0 1; do
  key=`expr 3788 + $id`
  var=/var/hpss/hdm/hdm$id
  $HPSS_PATH_BIN/hdm_admin -k $key -s $id -v $var ps >/dev/null 2>&1
  if [ $? != 0 ]; then
    echo "   starting hdm$id for cleanup"
    $HPSS_PATH_BIN/hpss_hdm $var/config.dat $id > $var/hdm.out 2>&1
  fi
  echo "   unmounting locally mounted filesets for hdm$id"
```

```
    $HPSS_PATH_BIN/hdm_admin -k $key -s $id -v $var tcp \
                                              disable >/dev/null 2>&1
  done
  exit 0
```

The **post_stop_dfs** script is executed once it has completed detaching the DFS aggregates and shutting down DFS. The script stops any HDMs that are still running.

Here is a sample for **post_stop_dfs**:

```
#!/bin/ksh
export HPSS_PATH_BIN=/opt/hpss/bin
# Stop any hdm's that are still running
for id in 0 1; do
  key=`expr 3788 + $id`
  var=/var/hpss/hdm/hdm$id
  $HPSS_PATH_BIN/hdm_admin -k $key -s $id -v $var ps >/dev/null 2>&1
  if [ $? = 0 ]; then
    echo "   stopping hdm$id"
    $HPSS_PATH_BIN/hdm_admin -k $key -s $id -v $var \
                                          -y stop >/dev/null 2>&1
  fi
done
exit 0
```

To start and stop DFS, be sure to use the commands start.dfs and stop.dfs. In particular, do not stop DFS using dfs.clean, since it does not handle HDM correctly.

7.6.3.3 Configuring an HDM Server (DFS & XFS)

The HDM server is configured with five configuration files, usually kept in **/var/hpss/hdm/hdm<id>.** Although any directory can be used, all five files must be kept together. By convention, the event and message logs are also kept in this same directory. The names of the configuration files are:

config.dat	basic configuration file
filesys.dat	description of file systems and filesets
gateways.dat	gateways permitted to access HDM
policy.dat	migration and purge policies
security.dat	configuration for Security Server (DFS systems only)
nshandle.dat	Name Server handle data (XFS systems only)

Four of the files (**config.dat**, **gateways.dat**, **policy.dat**, and **security.dat**) may be created using a text editor. The format of these files is described below. The files must be present before HDM can be started. If any one of these files is edited, HDM must be restarted before the change will take effect.

The fifth file (**filesys.dat**) is automatically updated by HDM as new aggregates and filesets are created. Therefore, this file should not ordinary be edited by the administrator. HDM cannot be started if this file is missing or does not contain correct information. Before starting HDM for the first time, a special version of **filesys.dat** must be created so that HDM will recognize that the file is correct.

It is possible to run multiple instances of HDM on one machine. For example, one HDM might be used to handle mirrored aggregates, while another might be used to handle archived aggregates. To set up a system in this manner, be sure to keep the associated configuration and log files in separate directories. For example, **/var/hpss/hdm/hdm1** and **/var/hpss/hdm/hdm2**.

Since logs necessary for HDM are heavily used and vital, it is recommended that configuration directories and the directories where the logs are stored be placed on file systems that are mirrored and have low latencies. Also, the log files defined in the **config.dat** must exist before the HDM is run. Simply use the **touch** command to create these files if they do not exist already. For XFS, an additional file, **nshandle.dat**, must also exist before the HDM is run. Create it using the **touch** command.

7.6.3.3.1 config.dat File

The basic configuration file, **config.dat**, is a text file that defines the configuration of an HDM server. It is recommended that the file be kept in the directory **/var/hpss/hdm/hdm<id>**.

The file consists of a series of lines, where each line defines one parameter. The first field on the line specifies the name of the parameter and the second field is the value for that parameter. The first line in the file must define **ServerID**. The following lines define the rest of the parameters. **ServerID** must begin in column one, while the other parameters must be indented by at least one tab character. The file may contain comments that start with a '**#**' character and continue until the end of the line.

<u>ServerID</u> is an arbitrary number used to distinguish different HDM servers defined in the configuration file. Once **ServerID** has been established, it should not be changed because it is used during event recovery whenever HDM is restarted. A good choice for **ServerID** is 1, since this is the default used by **hdm_admin**. HDM can be started with the command:

```
hdm_admin start /var/hpss/hdm/hdm<id>/config.dat <ServerID>
```

where **ServerID** is a number identifying which part of the configuration file HDM should read.

Following is an extract from a typical configuration file:

```
# An example HDM configuration file

ServerID 1
   DescName          Production configuration
   RegisterBitMap    0.0         #A comment
   [...]
```

Refer to the template file, **/opt/hpss/config/templates/hdm_config.dat.template** when constructing a **config.dat** file.

The following paragraphs discuss each parameter found in the file. Except as noted, each parameter must be specified. HDM will **not** start if a mandatory parameter is omitted. The configuration parameters can be specified in any order. The keywords must be spelled correctly, using the specified upper and lower case letters. For example, DescName, not descName or descname.

AclLogName specifies the name of the file used for the ACL log. Typically, this will be **/var/hpss/hdm/hdm<id>/hdm_acl_log**. This file contains a record of pending ACL change requests made through the HPSS interface. The file must exist before HDM is started, but can be empty. The size of the file is unbounded, but typically will be small. This log is very important and should be stored on a reliable disk.

DestroyLogName specifies the name of the file used for the destroy log. Typically, this will be **/var/hpss/hdm/hdm<id>/hdm_destroy_log**. This file contains a record of all files on mirrored filesets that need to be destroyed. The file must exist before HDM is started, but can be empty. If **DestroyLogSize** is changed, HDM automatically adjusts the size of the file. This log is very important and should be stored on a reliable disk.

DestroyLogSize specifies the total number of files in mirrored filesets that are waiting to be destroyed, thus determining the size of the log file. Because of limitations in DFS SMT, it is not possible to destroy HPSS files immediately when a destroy event is received. For example, after a recursive remove, the number of pending file destroys can become quite large. Once the log is full, attempts to remove file names and delete the file data will be delayed until the destroy process clears the log. Another consideration occurs when a user's program creates a file, opens it, and then unlinks it, expecting that when the program exits, the file will go away. HDM must keep an entry for this file in the destroy log until the user's process exits. This ties up destroy log entries. For these reasons, a fairly large **DestroyLogSize** should be used. On the other hand, avoid using an excessively large value because that causes more overhead when deleting files. Also, if the system has a heavily loaded archived file system, it may take a while for HDM to get around to destroying files on the mirrored file systems. Using a smaller **DestroyLogSize** tends to fix this problem. A good starting value for **DestroyLogSize** is 200. The name of this log file is specified by **DestroyLogName**.

If necessary, **DestroyLogSize** can be decreased by editing **config.dat** and restarting HDM. However, this only works if the new value is large enough to accommodate all of the outstanding entries in the old destroy log.

EventQueueSize specifies the maximum number of events HDM can queue for processing. Ideally, the number should be the sum of **NumDataProcesses**, **NumNamespProcesses**, and **NumAdminProcesses**, but it may be a good idea to use a slightly larger number. If the value is too small, some subprocesses could lie idle. For example, if the queue happens to fill with data events, then name space processes will lie idle until some of the data events have finished processing. A value in the range 20-50 is a good starting point.

ExecPath specifies the path name of the directory where HDM executables are located. Typically, this will be **/opt/hpss/bin**.

Flags defines special flags that control the operation of HDM. The parameter is specified as a series of keywords, separated by white space. Currently there are two keywords defined: "**permissiveMount**" and "**stdout**".

When a DFS aggregate is exported and "**permissiveMount**" is specified, HDM will check its tables to see if it manages that aggregate. If not, it assumes that some other HDM manages the aggregate and relays the event forward. If no other HDMs are prepared to manage the aggregate, it will be mounted but will not be kept in sync with HPSS. This flag is required when several HDM servers are run on one machine, but leads to the possibility that an aggregate will be overlooked and not kept properly synchronized.

On the other hand, if "**permissiveMount**" is not specified, HDM will abort mount events for aggregates it does not manage. While this is safer, it cannot be used on a machine where multiple copies of HDM are running.

The "permissiveMount" flag must be specified if several HDM servers are to be run on one machine, and should never be specified when only one is to be run.

If the "**stdout**" keyword is present on the Flags line, HDM will write all messages to standard output. While this option can help with debugging, its use is generally discouraged.

HPSSDMAPHostName specifies the host name for the machine where the HDM will be run. This should be a fully qualified host name. For example, **tardis.ca.sandia.gov**. Be sure to use the host name where HDM will be run, not the host name where the DMAP Gateway will be run.

HPSSDMAPTCPPort specifies the TCP port used by the HDM to listen for requests from the DMAP Gateway. Conventionally, this will be 6002. Check **/etc/services** to ensure that this port is not already being used by another program. Do not confuse this port number with the port used by the DMAP Gateway to receive requests from HDM, whose value is conventionally 7001. Make sure both port numbers mentioned here are consistent between HDM and DMAP Gateway configurations.

When multiple HDM servers are configured, each HDM must have a unique TCP port.

LogRecordMask specifies the type of messages recorded in the message log. A series of keywords, separated by white space, define the messages that are to be recorded. For example:

```
LogRecordMask alarm event trace debug
```

Legal keywords and their meanings are:

Table 7-4 LogRecordMask Keywords

Keyword	Description
accounting	accounting information; not currently used
alarm	error conditions of interest to the administrator
debug	low level error message for troubleshooting
event	informational messages (e. g., system starting)
request	request-specific messages; not currently used
security	security related events; not currently used
status	status messages; not currently used
trace	program flow and other low level messages

In normal operation, only alarm and event messages need to be enabled. Trace and debug messages should be enabled when it is necessary to track down the root cause of a problem. Logging too many different types of messages will impact HDM performance.

MainLogName specifies the name of the file used for the main event log. Typically, this will be **/var/hpss/hdm/hdm<id>/hdm_main_log**. This file contains a record of all name space events for which processing has not yet completed. The file must exist before the HDM is started, but can be empty. If **MainLogSize** is changed, HDM automatically adjusts the size of the file as needed. This log is very important and should be stored on a reliable disk. Also, since the file is frequently updated by HDM, it should be stored on a low latency disk.

MainLogSize specifies the total number of name space events that can be handled concurrently. While an event is being processed, information about the event is stored in a log file. When HDM restarts after a crash, it reads the log to determine what must be done to synchronize the DFS and HPSS name spaces. If too small a value is chosen for the log size, HDM may occasionally have to wait for a log entry to become available before it can process a name space event. There is no reason to select a value much larger than **NumNamespProcesses**. A value between 10-25 is a good starting point. If necessary, the value of this parameter can be decreased by editing **config.dat** and restarting HDM. However, this only works if the new value of **MainLogSize** is large enough to accommodate all of the outstanding events in the old event log. The name of this log file is specified by **MainLogName**.

If both the event log and event queue become full, event processing can not be done, and HDM will return an error to the DFS SMT.

MaxFilesets specifies the maximum number of filesets HDM can support. The number must be at least as large as the number of filesets on the **dmlfs** aggregates at the site. A larger value can be used if adding new filesets is anticipated. However, using a value that is too large adds overhead to event processing. For XFS, the value of this parameter should be identical to the value defined for **MaxFileSystems**.

MaxFileSystems specifies the maximum number of DFS aggregates or XFS filesystems the HDM can support. For DFS, this number must be at least as large as the number of **dmlfs** aggregates on the DFS File Server. A larger value can be used if adding new aggregates is anticipated. For XFS, it should be large enough to accomodate the planned number of XFS filesystems, and **MaxFilesets** must be set to the same value. If the value must be changed, HDM must be restarted before the change takes effect.

MaxGateways determines the maximum number of entries allowed in the **gateways.dat**. Since it is recommended to have two entries for each DMG (one with the short name and one with the fully-qualified name) and one entry for each administrative machine, this value should be at least 3. If this value is too small, the HDM will not start, so try to use the same number as there are lines in the **gateways.dat**.

MaxMsgFileSize specifies the maximum size in bytes of the message log files. When writing to a log file will exceed this size, the message logger automatically switches to the other log file. If the value for **MaxMsgFileSize** is too small, potentially useful information will be lost as older messages are overwritten with new ones. A good starting value is 5000000.

MaxPolicies specifies the maximum number of migration and purge policies that can be handled by HDM. The value should be large enough to accommodate all of the purge and migration policies defined in **policy.dat**. (There is no need to make this value larger than the sum of these policies.)

MaxStages specifies the maximum number of data event processes that can concurrently stage files from HPSS to Episode. When this limit is reached, further transfers from HPSS are deferred until one of the stages completes. This value must be less than **NumDataProcesses**. A value in the range 1-3 is a good starting point.

MaxTcpConnects specifics the maximum number of simultaneous requests to mirrored filesets managed by this HDM allowed through the HPSS interface. If the value for this parameter is too small, an HPSS user request may occasionally fail. A good starting value is 200.

MinArchiveMigrationSize defines the minimum file size that will be migrated into an archived fileset. If the migration process is taking too much CPU time, use this parameter to prevent small files from being migrated.

MsgFileDirectory specifies the path name of the directory where HDM error message files are located. Typically, this will be **/var/hpss/hdm/hdm<id>**. The log files will be named **hdm_message.0**, **hdm_message.1**, etc. The number of these files is controlled by **NumMessageFiles**. If **MsgFileDirectory** is left blank, the HDM will not write messages to the message files. In this situation, the **stdout** flag should be defined so that messages appear somewhere. But remember that using the **stdout** flag is discouraged, so it is better to provide a value for **MsgFileDirectory**.

NumAdminProcesses specifies the number of HDM subprocesses that will be assigned to handle administration events, such as mounting and unmounting file systems. The value, 2, is a good starting point.

NumDataProcesses specifies the number of HDM subprocesses that will be assigned to handle events involving data requests, mainly read and write events. The value selected must be large enough to allow a reasonable amount of I/O overlap. Remember that, at any given time, some number of these processes may be busy staging data from HPSS (**MaxStages**). The value, 8, is a good starting point.

NumMessageFiles specifies the number of files that HDM will use to write error messages. Unlike the other parameters, this parameter is optional. If it is not supplied, error messages will be written to two files. Message files will appear in the directory named by **MsgFileDirectory**, and the files will be named **hdm_message.0**, **hdm_message.1**, **hdm_message.2**, etc.

NumNamespProcesses specifies the number of HDM subprocesses that will be assigned to handle events involving changes to the name space. These events include file creates, deletes, renames, and permission changes. The value, 8, is a good starting point.

RegisterBitmap is a 64-bit number, expressed in the form "<high>.<low>", where <high> and <low> are integers providing the high and low 32 bits of the number. This field is not currently used, and should be set to 0.0.

SharedMemoryKey is an integer that defines a key used by HDM subprocesses to attach to the shared memory segment initialized by the main process. The key is also used to identify the HDM semaphore set. If a value of zero is listed in the file, a default key with the value, 3789, is substituted. If only one HDM is running on a machine, use zero for this parameter; this ensures that HDM and **hdm_admin** use the same key. The default setting can be changed with the environment variable, HPSS_HDM_SHMEM_KEY.

When multiple HDM servers are to be run on the same machine, each HDM must have a unique SharedMemoryKey. HDM servers cannot share memory or logs without serious consequences.

ZapLogName specifies the name of the file used for the zap log. Typically, this will be **/var/hpss/hdm/hdm<id>/hdm_zap_log**. This file contains a record of the archived fileset files that need to be destroyed. The file must exist before HDM is started, but can be empty. The size of the file is unbounded, but typically will be small. This log is very important and should be stored on a reliable disk.

7.6.3.3.2 filesys.dat File

The filesystem configuration file, filesys.dat, is a text file that defines each aggregate and fileset HDM manages. If HDM receives an event related to an aggregate or fileset not listed in the file, the event will be aborted and the end user will receive an error. The file must be located in the same directory as **config.dat**, typically **/var/hpss/hdm/hdm<id>**.

This file is maintained by HDM and should not need to be edited by an administrator. When HDM modifies the file, a copy of the original file is saved using a name like filesys.YYMMDDhhmmss, where YYMMDDhhmmss gives the date and time that the original file was saved (not the date when the original file was first created.) If many changes are made in a short period of time, HDM will not save a copy of each file, but rather will save the most recent file in filesys.bak, overwriting the previous backups in the process. For example, if the system administrator creates a large number of filesets at 10:00 AM on 4/1/99, when the operation is complete, three files will be created: a file named filesys.990401100000 containing a copy of the **filesys.dat** that was in effect before the new filesets were created; filesys.bak describing all but the last fileset; and filesys.dat describing all of the filesets.

When HDM is first started, **filesys.dat** must be present. It is not possible to start HDM if **filesys.dat** is empty or does not have the right format. To make a suitable file for starting HDM the first time, use a text editor to create a file whose first and last lines are identical and contain the string "**`# HDM filesys.dat version 1`**". Alternatively, create the file by copying a template file using commands such as these:

```
cd /opt/hpss/config/templates
cp hdm_filesys.dat.template /var/hpss/hdm/hdm<id>/filesys.dat
```

where **/var/hpss/hdm/hdm<id>** is the directory where **filesys.dat** is to beplaced.

Although HDM is responsible for maintaining **filesys.dat**, it may occasionally be necessary for the administrator to edit the file manually to correct problems. This section describes the format of the file.

It is not safe to edit the file any time XDSM managed filesets or aggregates are being created or deleted. Once the file has been edited, HDM must be restarted for the changes to take effect. Remember, until HDM is restarted, it can overwrite any changes.

Comments may appear in this file, but they will not be preserved when HDM updates the file. Comments begin with a '**#**' character and continue to the end of the line. HDM requires that the first and last lines in the file contain the comment "**`# HDM filesys.dat version 1`**". If these lines are absent or incorrect, HDM assumes the file is corrupt; so be careful not to alter these lines. With the exception of these two lines, HDM ignores comments.

For DFS, this file consists of a number of lines that describe the aggregates and the filesets that reside on that aggregate. Each aggregate is described by a line that begins in the first column. After the line for each aggregate are the lines that describe the filesets that reside on that aggregate. Fileset lines begin with a TAB character. Any line that begins in column one is treated as the definition for an aggregate.

An aggregate configuration line contains the following information:

```
      path media fsid option migratePolicy purgePolicy stageType
```

and the format for a fileset configuration line is:

```
      <TAB> ftname global local ftid gateway port
```

For XFS, each filesystem is listed on a single line beginning in column one. The format of this line is:

```
      name global media option migratePolicy purgePolicy MPQueueSize \
      ftid gateway port
```

The following is an example HPSS / DFS **filesys.dat**:

```
# HDM filesys.dat version 1
# Sample filesys.dat
/opt/dcelocal/var/dfs/aggrs/bkup1 /dev/bkup1 2 archive/delete wait \
run partial
   new.bkup ? ? 0.292 ? 0 # We haven't specified this yet
   hpss.bkup NO_MOUNT_POINT NO_MOUNT_POINT 0.232 tardis 7001
/opt/dcelocal/var/dfs/aggrs/mirror1 /dev/mirror1 4 mirrored run wait \
partial
   new.mirror ? ? 0.295 ? 0
   hpss.mirror /:/hpss/mirror /var/hpss/hdm/hdm1/aggr/mirror1 \
hpss.mirror 0.361 server.domain.ibm.com 7001
# HDM filesys.dat version 1
```

The following is an example HPSS / XFS **filesys.dat**:

```
# HDM filesys.dat version 1
# Sample filesys.dat
xfsbackup1 /mnt/xfsbackup1 ide0(3,74) archive/rename run wait 10000 \
1079914467.1018724031 server.domain.ibm.com 7001
xfsbackup2 /mnt/xfsbackup2 ide0(3,75) archive/delete run wait 10000 \
1079914467.1019508245 server.domain.ibm.com 7001
# HDM filesys.dat version 1
```

Following is a description of the parameters for a DFS aggregate:

Path specifies the path name where DFS mounts the aggregate. Typically, this is **/opt/dcelocal/var/dfs/aggrs/aggrname**, where **aggrname** is the name of the aggregate.

Media specifies the device file for the aggregate. Typically, this is **/dev/aggrname**, where **aggrname** is the name of the aggregate.

Fsid specifies the file system id for this aggregate. The value is defined by **dfstab**.

Option specifies how the filesets on the aggregate will be managed by HPSS. The parameter may be either **archive/delete**, **archive/rename**, or **mirror**. If **mirror** is selected, the name and data space will be mirrored by HPSS, and the end user can access the name and data space from either DFS or HPSS. Otherwise, only files are archived by HPSS, and the files can only be accessed from DFS. If **archive/delete** is selected, any files deleted from DFS will also be deleted from HPSS. If **archive/rename** is selected, any file deleted from DFS is not deleted from HPSS, but renamed instead. This allows for the possibility of restoring a file later, if it was accidentally deleted.

MigratePolicy and **PurgePolicy** specify the name of the policies that will be used to migrate and purge files on this aggregate. The name of the migrate policy must appear in the **MigratePolicy** section of **policy.dat**, and the name for the purge policy must appear in the **PurgePolicy** section of the file. In the example above, the names of the policies are **wait** and **run**, but these names have no special meaning to HDM. Section 7.6.3.3.4 discusses **policy.dat** in detail.

StageType specifies the type of staging HDM will use when it is necessary to stage a file from HPSS to DFS. Legal choices are **whole** and **partial**. If **whole** is specified, the whole file will be staged; otherwise only that part of the file necessary to satisfy a request will be staged. With **partial**, the amount of data staged will be at least the size of the data access, or if the data access is small and the file is large, a 16MB chunk of the file surrounding the data being accessed will be staged.

Following is a description of the parameters for a DFS fileset:

Ftname specifies the name of the fileset. The name should be the same as the name for the DFS fileset, which is also the name of the HPSS fileset.

Global specifies the global mount point for the fileset. This name will be a DFS style path name. For example, /:/hpss/mirror.

Local specifies the local mount point for the fileset. This name will be a UNIX style path name. Typically, the mount point will be in the directory, /**var/hpss/hdm/hdm<id>/aggr/<fileset name>**, where **fileset name** will be the same as the fileset name. For example, **hpss.mirror**.

Ftid specifies the fileset Id. The parameter is specified in the form <high>.<low>, where <high> and <low> are numbers representing the high and low 32 bits for the fileset Id. This Id should be the same as the DFS fileset Id, which is also the fileset Id of the HPSS fileset.

If a fileset uses one of the archive options, the global and local mount points will typically not be set and **filesys.dat** will show NO_MOUNT_POINT for these parameters. If a mirrored fileset will only be accessed by users local to the cell, the global mount point can also be specified as NO_MOUNT_POINT.

When a fileset is only partially configured, the global and local mount points are each represented by a '?'. While this condition exists, DFS users cannot access the fileset. Typically, this happens only for a short period of time while the administrator is setting up the HPSS fileset. To complete the configuration, an administrator will use SSM to create the HPSS fileset.

An administrator must not edit filesys.dat to "fix" the "?"s! This data will be assigned when the HPSS fileset is created.

Gateway specifies the fully qualified name of the host where the DMAP Gateway that will manage this fileset runs. To keep the example above short, **Gateway** is shown as tardis, but in practice, the name should be tardis.ca.sandia.gov. If the fileset is partially configured, the host name is represented by a '?'. That prevents end users from accessing the DFS fileset. To complete the configuration, an administrator will use SSM to create the HPSS fileset.

Port specifies the TCP port that the DMAP Gateway uses to listen for requests from HDM. Conventionally, this is 7001. Do not confuse this port number with the port number used by HDM to listen for requests from the DMAP Gateway (6002). Until the fileset is fully configured, **Port** will be shown as zero.

When multiple HDM Servers and DMAP Gateways are running, they must use different TCP ports.

Following is a description of the parameters for an XFS filesystem:

Name specifies the name of the filesystem. This is user-defined and should be descriptive.

Media specifies the device for the aggregate in the form of <adapter>(<major>,<minor>). To determine the media descriptor for your filesystem, you will need to follow these steps:

1. Find the <major> and <minor> numbers for the disk partition containing your filesystem by looking at a long listing of the **/dev** directory. For example, if your filesystem is on partition **/dev/hdb7**, do the following:

   ```
   % ls -l /dev/hdb7
   brw-rw----    1 root     disk       3,  71 Mar 23  2001 /dev/hdb7
   ```

   The listing shows that the major number is 3 and the minor number is 71.

2. Next, determine the <adapter> that your partition is managed through. Use the major number to lookup the appropriate block device in the **/proc/devices** file. Here is an example:

   ```
   % cat /proc/devices
   Character devices:
     1 mem
     2 pty
     3 ttyp
     4 ttyS
     5 cua
     7 vcs
    10 misc
    14 sound
    21 sg
   128 ptm
   136 pts
   162 raw
   180 usb
   226 drm
   ```

```
Block devices:
  2 fd
  3 ide0
  8 sd
 22 ide1
 65 sd
 66 sd
```

The block device that matches our major number (3) is **ide0**.

3. Now put this information together to form the media descriptor. Since the format is <adapter>(<major>,<minor>), for our example the media descriptor is `ide0(3,71)`.

Option specifies how the filesets on the filesystem will be managed by HPSS. The parameter may be either **archive/delete**, or **archive/rename**. In either case, only files are archived by HPSS (not directories), and the files can only be accessed from XFS. If **archive/delete** is selected, any files deleted from XFS will also be deleted from HPSS. If **archive/rename** is selected, any file deleted from XFS is not deleted from HPSS, but renamed instead. This allows for the possibility of restoring a file later, if it was accidentally deleted.

MigratePolicy and **PurgePolicy** specify the name of the policies that will be used to migrate and purge files on this aggregate. The name of the migrate policy must appear in the **MigratePolicy** section of **policy.dat**, and the name for the purge policy must appear in the **PurgePolicy** section of the file. In the example above, the names of the policies are **wait** and **run**, but these names have no special meaning to HDM. Section 7.6.3.3.4: *policy.dat File* on page 459 discusses **policy.dat** in detail.

MPQueueSize specifies the number of entries the HDM will allocate in shared memory for tracking migration and purge candidates. Refer to Section 7.6.2.8.1: *Migration/Purge Algorithms and the MPQueue* on page 444 for information and advice regarding this parameter.

Ftid specifies the fileset ID. The parameter is specified in the form <high>.<low>, where <high> and <low> are numbers representing the high and low 32 bits for the fileset ID. This ID should be the same as the fileset Id of the HPSS fileset.

When a fileset is only partially configured, the global mount point is represented by a '**?**'. Typically, this happens only for a short period of time while the administrator is setting up the HPSS fileset. To complete the configuration, an administrator will create the HPSS fileset using either SSM or the **create_fset** utility.

An administrator must not edit filesys.dat to "fix" the "?"s! This data will be assigned when the HPSS fileset is created.

Gateway specifies the fully qualified name of the host where the DMAP Gateway that will manage this fileset runs. If the fileset is partially configured, the host name is represented by a '**?**'. That prevents end users from accessing the XFS filesystem. To complete the configuration, an administrator will use SSM to create the HPSS fileset.

Port specifies the TCP port that the DMAP Gateway uses to listen for requests from HDM. Conventionally, this is 7001. Do not confuse this port number with the port number used by HDM to listen for requests from the DMAP Gateway (6002). Until the fileset is fully configured, the port will be shown as zero.

When multiple HDM Servers and DMAP Gateways are running, they must use different TCP ports.

7.6.3.3.3 *gateways.dat File*

The gateway configuration file, **gateways.dat**, is a text file identifying DMAP gateways that will communicate with HDM. The file must be located in the same directory as **config.dat**, typically **/var/hpss/hdm/hdm<id>**.

The file consists of a number of entries, each containing a host name, port, and encryption key. The host name is the name of the machine where a gateway runs or the name of a host where a system administrator will run sensitive commands such as **create_fsys**. The host name should be a fully qualified name, which includes the domain name. However, two entries may be necessary in some cases, one using the fully qualified name and one using the abbreviated name. Be sure to make entries in this file for every DMAP Gateway that HDM uses. For security reasons, do not name hosts that do not communicate with HDM.

Be sure to use the DMAP Gateway port (typically 7001) and not HDM port (typically 6002). If the machine is an administrative machine, use 0 for the port.

The encryption key is used to secure communications between HDM and DMAP Gateway. The value is expressed as a 16 digit hexadecimal number. The number must agree with the number entered on the DMAP Gateway's server specific configuration screen. The **gateways.dat** file should be protected to prevent unauthorized users from discovering the key.

For backward compatibility, the port and encryption key fields are optional. If the encryption key is missing, HDM will use an encryption key of zero. If the port is missing, a value of zero will be used. In this case, the named machine will not be able to act as a gateway, but will still be able to act as an administrative machine.

The following is a sample **gateways.dat** file:

```
##################################################################
# gateways.dat: sample gateway configuration file
#
# Normal entry has full name
tardis.ca.sandia.gov 7001  0123456789abcdef
# Short name may be needed
tardis               7001  0123456789abcdef
# An administrative machine
admin.daleks.com     0     fedcba9876543210
# Port = 0; key = 0
k9.master.com
##################################################################
```

7.6.3.3.4 *policy.dat File*

The policy configuration file, **policy.dat**, is a text file that describes the parameters used to control the migration and purge processes. The file must be located in the same directory as **config.dat**, typically, **/var/hpss/hdm/hdm<id>**.

The file consists of a number of sections, where each section defines a migration or purge policy. Each section begins with a line that identifies the type of policy being defined (a migration or purge policy) and gives it a name. Comments can appear in the file, starting with a '**#**' character and continuing to the end of the line. Following is an example policy file that will be used in the rest of the discussion:

```
##################################################################
# policy.dat: sample migration and purge policy definition file

# Migration policies

MigratePolicy  wait
   MigrationDelayTime            0
   LastAccessTimeBeforeMigration 3000

MigratePolicy  run
   MigrationDelayTime            86400
   LastAccessTimeBeforeMigration 3000

# Purge policies

PurgePolicy    wait
   PurgeDelayTime                0
   LastAccessTimeBeforePurge     4000
   UpperBound                    80
   LowerBound                    60

PurgePolicy    run
   PurgeDelayTime                300
   LastAccessTimeBeforePurge     4000
   UpperBound                    80
   LowerBound                    60
##################################################################
```

A migration policy is specified by a line that begins with **MigratePolicy**, and a purge policy is specified by a line that begins with **PurgePolicy**. The keyword is followed by the name of the policy being defined, such as, **wait** and **run**, in the example. The choice of policy names has no significance to HDM. If desired, the same name could be used to describe a migrate and a purge policy; HDM does not assume these policies are related. The configuration parameters that define a policy immediately follow the line that names the policy. The parameters can be in any order, as long as they are all provided. Each parameter line contains a keyword, preceded by a TAB character.

After the **MigratePolicy** line, there must be two lines defining the following parameters:

LastAccessTimeBeforeMigration specifies the number of seconds that must elapse after a file is accessed before the file becomes eligible for migration.

MigrationDelayTime specifies the time, in seconds, that the migration process waits between passes in which it looks for files to migrate. If the time is set to zero, HDM waits an infinite amount of time, meaning the migration process waits for a signal before looking for files to migrate. **hdm_admin** can be used to send the signal.

After the **PurgePolicy** line, there must be four lines defining the following parameters:

LastAccessTimeBeforePurge specifies the number of seconds that must elapse after a file is accessed before the file becomes eligible for purging.

PurgeDelayTime is the time, in seconds, that the purge process waits between passes in which it looks for files to purge. If this time is set to zero, HDM waits an infinite amount of time, meaning the purge process waits for a signal before looking for files to purge. **hdm_admin** can be used to send the signal.

UpperBound and **LowerBound** are integers between 0 to 100, representing percentages. The purge process initiates a purge cycle when the percentage of space used on the aggregate exceeds **UpperBound** and stops the cycle when the percentage drops below **LowerBound**. Needless to say, **LowerBound** should be less than **UpperBound**.

The choice of values for policy parameters is site specific. Some sites may choose to have an administrator to determine when to initiate migration and purge cycles. In this case, the infinite wait policies mentioned above would be appropriate. Other sites may choose to have the migration and purge cycles to run automatically, and for those sites, the run policies, also described above, would be better. In the second **MigratePolicy** in the above example, the migration process is set to run once every 24 hours, which may work well for a lightly loaded site. For other sites, it may be necessary to run the migration process more frequently to keep up with the load. The purge process should be run often enough to ensure that free space on the system is available. In the example above, **UpperBound** and **LowerBound** were chosen to keep between 60% and 80% of DFS on-line. Using too low of a value for **LowerBound** could lead to thrashing. Using too high of a value for **UpperBound** could cause end users to wait whenever the system needs to free space.

7.6.3.3.5 security.dat File (DFS Systems Only)

The security configuration file, **security.dat**, is a text file used by the DFS Security Server component of HDM for cross cell access through HPSS interfaces on mirrored filesets. The file must be located in the same directory as **config.dat**, typically, **/var/hpss/hdm/hdm<id>**.

The format of this file is similar to config.dat. Like, config.dat, the first line in security.dat contains the keyword **ServerID**, starting in column one, and provides its value. The lines for the remaining parameters immediately follow and must begin with a TAB character. All of the parameters must be defined. Following is an extract from a sample security.dat file:

```
######################################################
# security.dat: sample security configuration file
ServerID 1
   ServerName        /.:/hpss/hdm
   Principal         hpss_hdm
   KeyTabFile        /krb5/hpss.keytabs
   RecoveryFile      /var/hpss/hdm/hdm1/pag.dat
   ObjectID          7622b5a2-226e-11d2-9cdd-08005a4726ef
######################################################
```

ServerID is an arbitrary number used to distinguish different HDM servers defined in the security configuration file. Once **ServerID** has been established, it should not be changed because it is used during event recovery whenever HDM is restarted. Both **config.dat** and **security.dat** must contain a section for each **ServerID** that is defined. (Recall that this is one of the parameters entered on the **hdm_admin** execute line.)

KeyTabFile specifies the name of a UNIX file containing a copy of the DCE key for the HDM Security Server component. The file must exist and must contain an entry for the given **Principal**.

ObjectID specifies the DCE object UUID for an HDM. ObjectID is used by the endpoint mapper to distinguish between different instantiations of HDM servers, so a unique value must be used. **uuidgen** can be used to generate a unique UUID.

Principal specifies the name of the DCE principal that the HDM Security Server component will use.

RecoveryFile specifies the name of a UNIX file containing a copy of the Process Activation Group (PAG) records that the HDM Security Server maintains. The file must exist, but can be empty.

ServerName specifies the name of a CDS entry HDM uses to register its bindings.

7.6.3.3.6 nshandle.dat File (XFS Systems Only)

This file will be used by the XFS HDM to store the Name Server Handles of the HPSS filesets that are linked to the XFS file systems it manages. These handles are stored in binary form to reduce the risk of tampering. It is vital that this file never be directly edited or lost!

When first configuring the XFS HDM, this file will need to be created in the same directory as the configuration files by using the **touch** command.

Chapter 8

Initial Startup and Verification

8.1 Overview

This chapter provides instructions for starting up the HPSS servers, performing post-startup configuration, and verifying that the system is configured as desired. Briefly, here are the steps involved:

1. Start up the HPSS servers (Section 8.2: *Starting the HPSS Servers* (page 463))
2. Unlock the PVL drives (Section 8.3: *Unlocking the PVL Drives* on page 465)
3. Create HPSS storage space (Section 8.4: *Creating HPSS Storage Space* on page 465)
   - Import volumes into HPSS (Section 3.1.1: *Importing Volumes into HPSS* on page 53 of the *HPSS Management Guide*)
   - Create Storage Server resources (Section 3.1.2: *Creating the Storage Server Resources* on page 61 of the *HPSS Management Guide*)
4. Create additional HPSS Users (Section 8.5: *Create Additional HPSS Users* on page 465)
5. Create File Families (Section 8.6: *Creating File Families* on page 465)
6. Create Filesets and Junctions (Section 8.7: *Creating Filesets and Junctions* on page 465)
7. Create HPSS Directories (Section 8.8: *Creating HPSS directories* on page 466)
8. Verify HPSS Configuration (Section 8.9: *Verifying HPSS Configuration* on page 466)

8.2 Starting the HPSS Servers

After the HPSS servers and their associated data are configured, they can be started up using SSM. When they are started for the first time after HPSS configuration, it is recommended that the servers be brought up one at a time, even though it is possible to start up all the configured servers at once. Doing so will simplify the diagnostics of any server startup problems due to configuration errors.

SSM can be used to start up the following types of HPSS server:

- Bitfile Server
- DMAP Gateway
- Gatekeeper Server
- Location Server
- Log Client
- Log Daemon
- Metadata Monitor
- Migration / Purge Server
- Mover
- Name Server
- NFS Daemon
- NFS Mount Daemon
- Non-DCE Client Gateway
- Physical Volume Library
- Physical Volume Repository
- Storage Server

Before starting up the HPSS servers, ensure that all configured HPSS Startup Daemons are up and running. As part of the HPSS initial configuration, the Startup Daemon was configured to be invoked at the operating system startup. However, after configuration, it will need to be manually started. Invoke the **/etc/rc.hpss** script as **root** to start the HPSS Startup Daemon. As a default, the **rc.hpss** script will redirect all servers' output to **/dev/console**. The **rc.hpss** script can also be invoked with the **-o** option if redirection of the output to **/dev/console** is not desired.

If the servers' output is to be redirected to /dev/console, ensure that the HPSSLOG variable in the hpss_env file is not set to stdout *since this combination may severely degrade the HPSS performance.*

It is recommended that the Log Daemon and the Log Client(s) be started up before the other HPSS servers so that the HPSS servers' startup messages are logged to aid in the startup problem determinations.

Ensure that any required PVR controllers are up and running before bringing up the PVRs.

Refer to Section 1.6: *Starting HPSS Servers* on page 30 of the *HPSS Management Guide* for more information on server startup.

8.3 Unlocking the PVL Drives

As a default, all newly configured drives are locked. They must be unlocked before the PVL can use them. Refer to Section Section 5.5.1: *Unlocking a Drive* on page 102 of the *HPSS Management Guide* for more information.

8.4 Creating HPSS Storage Space

Adding storage space in HPSS is done in two distinct phases: import and create. The first phase, import, involves physically introducing the tape cartridges and the disk volumes into the PVL and labeling them with HPSS volume labels. The second phase, create, defines the Storage Server data structures that will describe the data stored on the imported volumes.

Refer to Section 3.1: *Creating HPSS Storage Space* (page 53) in the *HPSS Management Guide* for more information on creating HPSS storage space.

8.5 Create Additional HPSS Users

Refer to Section 8.1.1: *Adding HPSS Users* on page 215 of the *HPSS Management Guide* for more information on how to create additional HPSS users.

8.6 Creating File Families

Refer to Section 6.7.4: *File Family Configuration* (page 322) for more information on how to create file families.

8.7 Creating Filesets and Junctions

SSM can be used to create XDSM / HPSS (which includes both DFS / HPSS and XFS / HPSS) and HPSS-only filesets. SSM can also be used to create the junctions that link the filesets to the HPSS name space.

Before creating XDSM/HPSS filesets, DFS must already be configured (if it is to be used), and the procedures to set up the DFS and XFS filesets and file systems must already be performed. Refer to Section 7.6: *HDM Configuration on page 435 for more information.*

Ensure that the Name Server and the DMAP Gateway are up and running before creating the filesets and junctions. Also ensure that SSM is connected to both of the servers.

8.8 Creating HPSS directories

If **Log Archiving** is enabled, use an HPSS namespace tool such as **scrub** or **pftp**, create the **/log** directory in HPSS. This directory must be owned by **hpss_log** and have permissions **rwxr-xr-x**.

8.9 Verifying HPSS Configuration

After HPSS is up and running, the administrator should use the following checklist to verify that HPSS was configured correctly:

8.9.1 Global Configuration

- Verify that a **Default Class of Service** has been selected.
- Verify that a **Root Name Server** has been selected.

8.9.2 Storage Subsystem Configuration

- Verify that a **Default Class of Service Override** has been selected if desired.
- Verify that a **Gatekeeper** has been selected if gatekeeping or account validation is required.
- Verify that the **Allowed Classes of Service** list has been filled in correctly.
- Verify that a Name Server, Bitfile Server, Migration/Purge Server, and appropriate Disk and/or Tape Storage Servers have been configured for easy storage subsystem.
- Verify that each storage subsystem is accessible by using **lsjunctions** and ensuring that there is at least one junction to the Root fileset of each subsystem. (The root fileset for a given subsystem can be found in the specific configuration for the subsystem's Name Server)

8.9.3 Servers

- Verify that all required HPSS servers are configured properly and the servers are up and running.
- Verify that a log policy is configured for each server. The HPSS log should be reviewed periodically to verify that the desired level of information is logged. In addition, verify that all Mover log policies have the **DEBUG** flag turned on to aid in the diagnostics of future data transfer problems.

8.9.4 Devices and Drives

- Verify that all devices/drives are configured and each is assigned to an appropriate PVR/Mover.

- For tape devices, verify that the "**Locate Support**" option is enabled (unless there are unusual circumstances why this functionally is not or cannot be supported).
- For tape devices, verify that the "**NO-DELAY**" option is enabled (unless there are unusual circumstances why this functionally is not or cannot be supported).
- For disk devices, verify that the "**Multiple Mover Tasks**" flag is enabled.
- For disk devices, verify that the "**Bytes on Device**" and "**Starting Offset**" values are correct, and the sum of the two values does not exceed the actual size of the underlying device.
- Verify that all configured drives are unlocked.

8.9.5 Storage Classes

- Verify that all storage classes are defined and each has sufficient storage space created to store HPSS files.
- Each storage class that will support migration and purge are configured with the appropriate migration and purge policy.
- A storage class at the lowest level in a hierarchy must not be configured with a migration or purge policy.
- To support repack and recover of tape volumes, the stripe width of a each tape storage class must not be more than half of the number of available drives of the same drive type.

8.9.6 Storage Hierarchies

- Verify that all storage hierarchies are defined properly.

8.9.7 Classes of Service

- Verify that all classes of service are defined properly.
- Verify that each COS is associated with the appropriate storage hierarchy
- Verify that the COS is intended to use the characteristics of the hierarchy and the underlying storage classes. In addition, verify that the classes of service have the correct **Minimum File Size** and **Maximum File Size** values. If these sizes overlap, the file placement may be indeterminate when the user creates a file using the size hints. For classes of services which are not to be used as part of standard file placement, set their **Force Selection** flag to ON so that they will only be chosen if specified by their COS ID.
- Verify that classes of service with multiple copies have the **Retry Stage Failures from Secondary Copy** flag enabled.

8.9.8 File Families, Filesets, and Junctions

- Verify that file families and filesets are created according to the site's requirement.
- Verify that each fileset is associated with the appropriate file family and/or COS.
- Verify that each fileset has an associated junction.

8.9.9 User Interfaces

- Verify that the desired HPSS user interfaces (FTP, NFS, DFS etc.) are properly configured.

8.9.10 Operational Checklist

8.9.10.1 Use each configured user interface

- Create files of various sizes on each defined COS.
- Verify that the files are created on the expected storage class with acceptable transfer rates.
- If necessary, redefine the associated storage class definition to enhance the throughput performance.
- The characteristics fields (**Transfer Rate**, **Latency**, etc.) in the storage class and class of service definition should be updated to reflect actual performance results.
- After the files are created on the correct storage class, verify that the files are created with correct file ownerships and permissions.
- Verify that other file operations (**delete**, **chmod**, **copy**, etc.) work properly.
- If accounting is configured, verify that the files are created with the correct accounting indices.
- If file families, filesets and junctions are configured, verify that they work as intended.

8.9.10.2 Verify Storage Management

- Verify that migration and purge operations work as intended.
- Force Migration and Force Purge operations to verify that files are migrated and purged correctly.
- Verify that files can be accessed after being migrated/purged.
- Monitor free space from the top level storage class in each hierarchy to verify that the migration and purge policy are sufficient in maintaining adequate free space.

8.9.11 Performance

Measure data transfer rates in each COS for:

- Client writes to disk
- Migration from disk to tape
- Staging from tape to disk
- Client reads from disk

Transfer rates should be as fast as the underlying hardware. The actual hardware speeds can be obtained from their specification and by testing directly from the operating system. For example, using **dd** to read and write to each device. Keep in mind that performance testing can often be limited by other factors external to HPSS. For example, a client reading a file from HPSS may be limited by the performance of the UNIX file system while writing the file rather than while reading the file from HPSS.

8.9.12 The Global Fileset File

If there are any other HPSS systems running in this DCE cell, verify that all of the Name Servers are sharing the same Global Filesets file.

Appendix A

Glossary of Terms and Acronyms

ACI	Automatic Media Library Client Interface
ACL	Access Control List
ACSLS	Automated Cartridge System Library Software (Science Technology Corporation)
ADIC	Advanced Digital Information Corporation
accounting	A log record message type used to log information to be used by the HPSS Accounting process. This message type is not currently used.
aggregate	A disk partition that has been modified to provide support for DFS filesets and access control lists.
AIX	Advanced Interactive Executive
alarm	A log record message type used to log high-level error conditions. The default logging policy is to log alarms and send alarms to the Storage System Management (SSM) to be displayed in the Alarm and Event window.
AML	Automated Media Library
AMS	Archive Management Unit
ANSI	American National Standards Institute
API	Application Program Interface
Archive	One or more interconnected storage systems of the same architecture.
archived fileset	A DFS fileset whose files are archived on HPSS but do not appear in the HPSS name space. Users can access these files from DFS but not from HPSS.

attribute	When referring to a managed object, an attribute is one discrete piece of information, or set of related information, within that object.
attribute change	When referring to a managed object, an attribute change is the modification of an object attribute. This event may result in a notification being sent to SSM, if SSM is currently registered for that attribute.
audit (security)	An operation that produces lists of HPSS log messages whose record type is SECURITY. A security audit is used to provide a trail of security-relevant activity in HPSS.
Bar code	An array of rectangular bars and spaces in a predetermined pattern (e.g., UPC symbol)
BFS	Bitfile Server
bitfile	A logical string of bits unrestricted in size or internal structure. HPSS imposes a size limitation in 8-bit bytes based upon the maximum size in bytes that can be represented by a 64-bit unsigned integer.
bitfile segment	An internal metadata structure, not normally visible, used by the Bitfile Server to map contiguous pieces of a bitfile to underlying storage provided by a Storage Server.
Bitfile Server	An HPSS server that provides a logical abstraction of bitfiles to its clients.
BMUX	Block Multiplexer Channel
bytes between tape marks	The number of data bytes that are written to a tape virtual volume before the Tape Storage Server requires a tape mark on the physical media.
CAP	Cartridge Access Port
cartridge	A physical media container, such as a tape reel or cassette, capable of being mounted on and dismounted from a drive. A fixed disk is technically considered to be a cartridge because it meets this definition and can be logically mounted and dismounted.
CDS	Cell Directory Service
central log	The main repository of logged messages from all HPSS servers enabled to send messages to the Log Daemon.
class	A type definition in Java. It defines a template on which objects with similar characteristics can be built, and includes variables and methods specific to the class.
Class of Service	A set of storage system characteristics used to group bitfiles with similar logical characteristics and performance requirements together. A Class of Service is supported by an underlying hierarchy of storage classes.
cluster	The unit of storage space allocation on HPSS disks. Each disk virtual volume is divided into a number of clusters such that the number is less than 16384. The smallest amount of disk space that can be allocated from

	a virtual volume is a cluster.
configuration	The process of initializing or modifying various parameters affecting the behavior of an HPSS server or infrastructure service.
configuration file	An Encina Structured File Server (SFS) file that stores information defining HPSS server operating parameters, storage characteristics, policies, devices and drives, and other information.
COS	Class of Service
daemon	A UNIX program that runs continuously in the background, lying dormant until some condition is met.
Data Server	A Storage System Management (SSM) component that provides the bridge between the System Manager and the Graphical User Interface (GUI).
DCE	Distributed Computing Environment
debug	A log record message type used to log lower-level error conditions. The default logging policy is to log debug messages.
DEC	Digital Equipment Corporation.
delog	The process of extraction, formatting, and outputting HPSS central log records.
deregistration	The process of disabling notification to SSM for a particular attribute change.
descriptive name	A human-readable name for an HPSS server.
device	A physical piece of hardware, usually associated with a drive, that is capable of reading or writing data.
DFS	The Distributed File Service is a system that joins local files systems of File Server machines, making the file systems available globally.
DFS/HPSS fileset	A fileset that is represented in both DFS and HPSS.
directory	An HPSS object than can contain files, symbolic links, hard links, and other directories.
dismount	An operation in which a cartridge is either physically or logically removed from a device, rendering it unreadable and unwritable. In the case of tape cartridges, a dismount operation is a physical operation. In the case of a fixed disk unit, a dismount is a logical operation.
DMAP Gateway	A server that acts as a gateway between DFS or XFS and HPSS. The server relays requests between HPSS and the HPSS/DMAP server.
DMAPI	The Data Management APIs defined by the XDSM specification. These APIs allow a Data Management Application to monitor events on files, and provide special interfaces to manage the data in the files.

DMG	Shorthand for DMAP Gateway.
DMLFS	A DCE Local File System that has been modified to support XDSM Data Management APIs.
DNS	Domain Name Service
DOE	Department of Energy
drive	A physical piece of hardware capable of reading and / or writing mounted cartridges. The terms device and drive are often used interchangeably.
DTS	Distributed Time Service
Encina	A product from Transarc Corporation that serves as the HPSS transaction manager. The Encina Structured File Server (SFS) serves as the HPSS Metadata Manager.
ERA	Extended Registry Attribute
ESCON	Enterprise System Connection
event	A log record message type used to log informational messages (e.g., subsystem starting, subsystem terminating). The default logging policy is to log events and send events to SSM to be displayed in the Alarm and Event window.
export	An operation in which a cartridge and its associated storage space are removed from the HPSS system. An export translates into a removal of a cartridge’s storage space from HPSS followed by an eject, which is the removal of the cartridge itself from its Physical Volume Repository.
FDDI	Fiber Distributed Data Interface.
file	An object than can be written to, read from, or both, with attributes including access permissions and type, as defined by POSIX (P1003.1-1990). HPSS supports only regular files.
file family	An attribute of an HPSS file that is used to group a set of files on a common set of tape virtual volumes.
file server	A machine that manages one or more DFS aggregates.
fileset	A collection of related files that are organized into a single easily managed unit. A fileset is a disjoint directory tree that can be mounted in some other directory tree to make it accessible to users.
fileset id	A 64-bit number that uniquely identifies a fileset.
fileset name	A name that uniquely identifies a fileset.
file system	Another term for a DFS aggregate.

file system id	A 32-bit number that uniquely identifies an aggregate.
FTP	File Transfer Protocol
Gatekeeper Server	An HPSS server that provides two main services: the ability to schedule the use of HPSS resources referred to as the Gatekeeping Service, and the ability to validate user accounts referred to as the Account Validation Service.
Gatekeeping Service	A registered interface in the Gatekeeper Server that provides a site the mechanism to create local policy on how to throttle or deny create, open and stage requests and which of these request types to monitor.
Gatekeeping Site Interface	The APIs of the gatekeeping site policy code.
Gatekeeping Site Policy	The gatekeeping shared library code written by the site to monitor and throttle create, open, and/or stage requests.
GB	Gigabyte (2^{30})
GDA	Global Directory Agent
GDS	Global Directory Service
GECOS	The comment field in a UNIX password entry that can contain general information about a user, such as office or phone number.
GID	Group Identifier
GK	Gatekeeper Server.
global mount point	A path in the DFS name space where a fileset has been mounted. A global mount point typically starts with the characters '/:'.
GSS	Generic Security Service
GUI	Graphical User Interface
HA	High Availability
HACMP	High Availability Clustered Multi-Processing - A software package used to implement high availability systems.
halt	A forced shutdown.
HDM	Shorthand for HPSS/DMAP.
hierarchy	See Storage Hierarchy.
HIMF	HPSS Interim Metadata Format
HiPPI	High Performance Parallel Interface
HPSS	High Performance Storage System

HPSS-only fileset	An HPSS fileset that has no counterpart in DFS.
HPSS/DMAP	A Data Management Application that monitors DFS or XFS activity in order to keep DFS or XFS synchronized with HPSS. The server relays requests between DFS or XFS and the DMAP Gateway.
IBM	International Business Machines Corporation
ID	Identifier
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
I/E	Import/Export
IETF	Internet Engineering Task Force
Imex	Import/Export
import	An operation in which a cartridge and its associated storage space are made available to the HPSS system. An import translates into an inject (the physical introduction of a cartridge to a Physical Volume Repository) followed by the addition of the cartridge's storage space to the HPSS system.
I/O	Input/Output
IOD/IOR	Input/Output Descriptor / Input/Output Reply
IP	Internet Protocol
IRIX	SGI's implementation of UNIX
JNI	Java Native Interface; a set of APIs by which Java objects and methods may be accessed from C code and C data structures and functions may be accessed from Java code within the same program.
junction	A mount point for an HPSS fileset. The fileset may be in the either a local or remote HPSS system.
KB	Kilobyte (2^{10})
LAN	Local Area Network
LANL	Los Alamos National Laboratory
LARC	Langley Research Center
latency	For tape media, the average time in seconds between the start of a read or write request, and the time when the drive actually begins reading or writing the tape.

LCU	Library Control Unit
LFS	A DCE Local File System, which is a high performance log-based file system that supports the use of access control lists and multiple filesets within a single aggregate.
LLNL	Lawrence Livermore National Laboratory
LMCP	Library Manager Control Point
LMU	Library Management Unit
local log	An optional circular log maintained by a Log Client. The central log contains formatted messages from all enabled HPSS servers residing on the same node as the Log Client.
local mount point	The name of a directory in a Unix file system where a DFS fileset has been mounted.
location server	An HPSS server that is used to help clients locate the appropriate Bitfile Server, Name Server, and/or other HPSS server to use for a particular request.
Log Client	An HPSS server executing on each HPSS node that is responsible for sending log messages to the local log, to the Log Daemon for central logging, and to SSM to display messages in the Alarm and Event window.
Log Daemon	An HPSS server responsible for writing log messages to the central log.
log record	The records received and maintained in a central log by the HPSS Log Daemon.
log record type	An indicator of whether a message to be logged is an alarm, event, status, debug, request, security, or accounting record.
logging service	An HPSS infrastructure service consisting of a central Log Daemon, one or more Log Clients, and server-specific logging policies.
LRU	Least Recently Used
LS	Location Server
LTO	Limited Tape Open
MAC	Mandatory Access Control
managed object	A programming data structure that represents an HPSS system resource. The resource can be monitored and controlled by operations on the managed object. Managed objects in HPSS are used to represent servers, drives, storage media, jobs, and other resources.
MB	Megabyte (2^{20})
metadata	Control information about the data stored under HPSS, such as location, access times, permissions, and storage policies.

Metadata Manager	The subsystem/component within HPSS responsible for the physical storage and management of HPSS metadata as well as the transactional mechanisms for manipulating HPSS meta data. The current Metadata Manager for HPSS is the Encina SFS product, together with a set of HPSS-developed application program interfaces (APIs) that provide a layer of abstraction on top of the Encina SFS data access methods.
Metadata Monitor	A type of HPSS server that is responsible for monitoring the space utilization of a single Encina SFS process. The Metadata Monitor calculates the overall amount of disk space being used by SFS and generates alarms to SSM as appropriate whenever various thresholds are exceeded.
method	A Java function or subroutine
migrate	To copy file data from a level in the file's hierarchy onto the next lower level in the hierarchy.
Migration/Purge Server	An HPSS server responsible for supervising the placement of data in the storage hierarchies based upon site-defined migration and purge policies.
mirrored fileset	A DFS fileset whose files are mirrored on HPSS. These files appear in both the DFS and HPSS name spaces, and so users can access the files from either place.
MM	Metadata Manager
MMON	Metadata Monitor
mount	An operation in which a cartridge is either physically or logically made readable and/or writable on a drive. In the case of tape cartridges, a mount operation is a physical operation. In the case of a fixed disk unit, a mount is a logical operation.
mount point	A place where a fileset is mounted in the DFS and/or HPSS name spaces.
Mount Daemon	An HPSS server object responsible for performing mount request operations for client systems accessing HPSS data through the HPSS Network File System (NFS) Daemon.
Mover	An HPSS server that provides control of storage devices and data transfers within HPSS.
MPS	Migration/Purge Server
MRA	Media Recovery Archive
MSSRM	Mass Storage System Reference Model
MVR	Mover
NASA	National Aeronautics and Space Administration
Name Server	An HPSS server that provides a mapping between names and machine-oriented identifiers. In addition, the Name Server performs access

	verification and provides the Portable Operating System Interface (POSIX).
name space	The set of name-object pairs managed by the HPSS Name Server.
NDCG	Non-DCE Client Gateway
NDAPI	Non-DCE Client Application Program Interface
NERSC	National Energy Research Supercomputer Center
Network File System	A protocol developed by Sun Microsystems that allows transparent access to files over a network.
NFS	Network File System
NFS Daemon	An HPSS server that provides access to HPSS name space objects and bitfile data for client systems through the Network File System protocol.
NLS	National Language Support
Non-DCE Client Gateway	An HPSS server that provides access to the user calls of the HPSS Client Application Program Interface for Non-DCE client applications.
Non-DCE Client Application Program Interface	An HPSS library that offers the user calls of the HPSS Client API for client applications running on platforms which do not support either DCE or Encina.
notification	A notice from one server to another about a noteworthy occurrence. HPSS notifications include notices sent from other servers to SSM of changes in managed object attributes, changes in tape mount information, and log messages that are alarm, event, and status log record message types.
NS	Name Server
NSL	National Storage Laboratory
object	See Managed Object.
ODM	Object Data Manager
OFD	Open File Descriptor
ONC	Online Network Computing
OSF	Open Software Foundation
OS/2	Operating System (multi-tasking, single user) used on the AMU controller PC
PB	Petabyte (2^{50})
PFTP	Parallel File Transfer Protocol

physical volume	An HPSS object managed jointly by the Storage Server and the Physical Volume Library that represents the portion of a cartridge that can be contiguously accessed when mounted. A single cartridge may contain multiple physical volumes.
Physical Volume Library	An HPSS server that manages mounts and dismounts of HPSS physical volumes.
Physical Volume Repository	An HPSS server that manages the robotic or human agent responsible for mounting and dismounting cartridges.
PIOFS	Parallel I/O File System
POSIX	Portable Operating System Interface (for computer environments)
purge	Deletion of file data from a level in the file's hierarchy after the data has been duplicated at lower levels in the hierarchy and is no longer needed at the deletion level.
purge lock	A lock applied to a bitfile which prohibits the bitfile from being purged.
PV	Physical Volume
PVL	Physical Volume Library
PVM	Physical Volume Manager
PVR	Physical Volume Repository
RAID	Redundant Array of Independent Disks
RAIT	Redundant Array of Independent Tapes
RAM	Random Access Memory
reclaim	The act of making virtual volumes that have no active data (i.e., empty) available for use in storing new data, so that data media can be reused.
registration	The process by which SSM requests notification of changes to specified attributes of a managed object.
reinitialization	An HPSS SSM administrative operation that directs an HPSS server to reread its latest configuration information, and to change its operating parameters to match that configuration, without going through a server shutdown and restart.
repack	The act of moving data from a virtual volume onto another virtual volume with the same characteristics (storage class) with the intention of freeing up all data references to that virtual volume. The act of emptying a virtual volume of all file data.
request	A log record message type used to log some action being performed by an HPSS server on behalf of a client. The default logging policy is to log request messages.

RISC	Reduced Instruction Set Computer/Cycles
RMI	Remote Method Invocation; the Java form of remote procedure call
RMI registry	The service with which Java programs register themselves to run remote methods and by which they find the locations of other Java programs which offer remote methods.
RMS	Removable Media Service
RPC	Remote Procedure Call
Sammi	A commercial software product group from Kinesix Corporation that manages the graphical user interface to SSM in conjunction with the SSM Data Server.
SCSI	Small Computer Systems Interface
security	A log record message type used to log security related events (e.g., authorization failures). The default logging policy is to log security messages.
SFS	Structured File Server
SGI	Silicon Graphics
shelf tape	A cartridge which has been physically removed from a tape library, but whose file metadata still resides in HPSS.
shutdown	An HPSS SSM administrative operation that causes a server to stop its execution gracefully.
sink	The set of destinations to which data is sent during a data transfer (e.g., disk devices, memory buffers, network addresses).
SMC	SCSI Medium Changer
SMIT	System Management Interface Tool
SNL	Sandia National Laboratories
SOID	Standard Object ID. An internal HPSS storage object identifier that uniquely identifies a storage resource.
source	The set of origins from which data is received during a data transfer (e.g., disk devices, memory buffers, network addresses).
SP	Scalable Processor
SS	Storage Server
SSA	Serial Storage Architecture

SSM	Storage System Management
SSM session	The environment in which an SSM user interacts with SSM to monitor and control HPSS through the SSM windows. SSM itself may be running without any sessions active. When an SSM user starts up Sammi and logs in, an SSM session begins and lasts until the user logs off. It is possible to have multiple sessions accessing the same SSM.
stage	To copy file data from a level in the file's hierarchy onto the top level in the hierarchy.
start-up	An HPSS SSM administrative operation that causes a server to begin execution.
status	A log record message type used to log processing results. This message type is being used to report status from the HPSS Accounting process. The default logging policy is to log the status messages and send them to SSM to be displayed in a pop-up window. Separate windows are displayed for each unique request. The message text within the pop-up window will update when additional messages are received for a particular request.
STK	Storage Technology Corporation
storage class	An HPSS object used to group storage media together to provide storage for HPSS data with specific characteristics. The characteristics are both physical and logical.
storage hierarchy	An ordered collection of storage classes. The hierarchy consists of a fixed number of storage levels numbered from level 1 to the number of levels in the hierarchy, with the maximum level being limited to 5 by HPSS. Each level is associated with a specific storage class. Migration and stage commands result in data being copied between different storage levels in the hierarchy. Each Class of Service has an associated hierarchy.
storage level	The relative position of a single storage class in a storage hierarchy. For example, if a storage class is at the top of a hierarchy, the storage level is 1.
storage map	An HPSS object managed by the Storage Server and used to keep track of allocated storage space.
storage segment	An HPSS object managed by the Storage Server and utilized by the Bitfile Server to provide storage for a bitfile or parts of a bitfile.
Storage Server	An HPSS object that provides control over a hierarchy of virtual and physical storage resources.
Storage Subsystem	A portion of the HPSS namespace that is managed by an independent Name Server, Bitfile Server, Migration / Purge Server, and an appropriate combination of Disk and / or Tape Storage Servers.
Storage System Management (SSM)	An HPSS component that provides monitoring and control of HPSS via a windowed operator interface. SSM has three components: (1) the System Manager, which communicates with all other HPSS components requiring monitoring or control, (2) the Data Server, which provides the bridge

	between the System Manager and the GUI, and (3) the GUI itself, which includes the Sammi Runtime Environment and the set of SSM windows.
stripe length	The number of bytes that must be written to span all the physical storage media (physical volumes) that are grouped together to form the logical storage media (virtual volume). The stripe length equals the virtual volume block size multiplied by the number of physical volumes in the stripe group (i.e., stripe width).
stripe width	The number of physical volumes grouped together to represent a virtual volume.
System Manager	A Storage System Management (SSM) server that communicates with all other HPSS components requiring monitoring or control.
TB	Terabyte (2^{40})
TCP/IP	Transmission Control Protocol/Internet Protocol
trace	A log record message type used to record entry/exit processing paths through HPSS server software. The default logging policy is to *not* log trace message types.
transaction	A programming construct that enables multiple data operations to possess the following properties: All operations commit or abort/roll-back together such that they form a single, atomic unit of work. All data modified as part of the same transaction are guaranteed to maintain a consistent state whether the transaction is aborted or committed. Data modified from one transaction are isolated from other transactions until the transaction is either committed or aborted. Once the transaction commits, all changes to data are guaranteed to be permanent.
TTY	Teletypewriter
UDP	User Datagram Protocol
UID	User Identifier
UPC	Universal Product Code
UUID	Universal Unique Identifier
virtual volume	An HPSS object managed by the Storage Server that is used to represent logical media. A virtual volume is made up of a group of physical storage media (a stripe group of physical volumes).
virtual volume block size	The size of the block of data bytes that is written to each physical volume

	of a striped virtual volume before switching to the next physical volume.
VV	Virtual Volume
XCT	Cross Cell Trust
XDSM	The Open Group's Data Storage Management standard. It defines APIs that use events to notify Data Management applications about operations on files.
XFS	A file system created by SGI available as open source for the Linux operating system.

Appendix B References

1. *3580 Ultrium Tape Drive Setup, Operator and Service Guide* GA32-0415-00
2. *3584 UltraScalable Tape Library Planning and Operator Guide* GA32-0408-01
3. *3584 UltraScalable Tape Library SCSI Reference* WB1108-00
4. *AIX Performance Tuning Guide*
5. *Data Storage Management (XDSM)* API, ISBN 1-85912-190-X
6. *DCE for AIX, Version 3.1: Quick Beginnings*
7. *Encina Administration Guide Volume 1: Introduction and Configuration*
8. *Encina Administration Guide Volume 2: Basic Administration*
9. *Encina Administration Guide Volume 3: Server Administration*
10. *Encina Administration Guide Volume 4: Advanced Administration*
11. *HACMP for AIX, Version 4.4: Concepts and Facilities*
12. *HACMP for AIX, Version 4.4: Planning Guide*
13. *HACMP for AIX, Version 4.4: Installation Guide*
14. *HACMP for AIX, Version 4.4: Administration Guide*
15. *HACMP for AIX, Version 4.4: Troubleshooting Guide*
16. *HPSS Management Guide*, September 2002, Release 4.5.
17. *HPSS Error Messages Reference Manual*, September 2002, Release 4.5.
18. *HPSS Programmer's Reference Guide*, Volume 1, September 2002, Release 4.5.
19. *HPSS Programmer's Reference Guide*, Volume 2, September 2002, Release 4.5.

20. *HPSS User’s Guide*, September 2002, Release 4.5.
21. *IBM 3494 Tape Library Dataserver Operator's Guide*, GA32-0280-02
22. *IBM 3495 Operator’s Guide*, GA32-0235-02
23. *IBM AIX Version 4.3 Installation Guide,* SC23-4112-01
24. *IBM DCE for AIX, Version 3.1: Introduction to DCE*
25. *IBM DCE Version 3.1 for Solaris: Quick Beginnings.*
26. *IBM DFS Version 3.1 for Solaris: Quick Beginnings.*
27. *IBM DCE Version 3.1 for AIX and Solaris: Administration Guide-Introduction.*
28. *IBM DCE Version 3.1 for AIX and Solaris: Administration Guide-Core Components.*
29. *IBM DCE Version 3.1 for AIX and Solaris: Administration Commands Reference.*
30. *IBM DFS Version 3.1 for AIX and Solaris: DFS Administration Guide.*
31. *IBM DFS Version 3.1 for AIX and Solaris: DFS Administration Reference.*
32. *IBM SCSI Device Drivers: Installation and User's Guide*, GC35-0154-01
33. *IBM Ultrium Device Drivers Installation and User’s Guide GA32-0430-00.1*
34. *IBM Ultrium Device Drivers Programming Reference* WB1304-01
35. *IBM WebSphere Application Server TXSeries for Solaris Version 3.0: Planning and Installation Guide (Configuring Encina).*
36. *Installing, Managing, and Using the IBM AIX Parallel I/O File System*, SH34-6065-02
37. *OSF/DCE Application Development Reference Manual*, SR28-4995-00
38. *Parallel and ESCON Channel Tape Attachment/6000 Installation and User's Guide*, GA32-0311-02
39. *Platform Notes: The hme FastEthernet Device Driver* 805-4449
40. *Sammi Runtime Reference* (for Version 4.0)
41. *Sammi System Administrator’s Guide* (for Version 3.0)
42. *Sammi User’s Guide* (for Version 3.0)
43. *Solaris 5.8 11/99 on Sun Hardware Documentation Guide*
44. *Solaris 5.8 (SPARC Platform Edition) 11/99 Release Notes*
45. *Solaris 5.8 (SPARC Platform Edition) Installation*

46. *Solaris 5.8 11/99 Sun Hardware Platform Guide*

47. *Solaris System Administration Guide, Volume I*

48. *Solaris System Administration Guide, Volume II*

49. *STK Automated Cartridge System Library Software (ACSLS) System Administrator's Guide*, PN 16716

50. *STK Automated Cartridge System Library Software Programmer's Guide*, PN 16718

51. J. Steiner, C. Neuman, and J. Schiller, *"Kerberos: An Authentication Service for Open Network Systems,"* USENIX 1988 Winter Conference Proceedings (1988).

52. R.W. Watson and R.A. Coyne, "*The Parallel I/O Architecture of the High-Performance Storage System (HPSS),*" from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.

53. T.W. Tyler and D.S. Fisher, "*Using Distributed OLTP Technology in a High-Performance Storage System,*" from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.

54. J.K. Deutsch and M.R. Gary, "*Physical Volume Library Deadlock Avoidance in a Striped Media Environment,*" from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.

55. R. Grossman, X. Qin, W. Xu, H. Hulen, and T. Tyler, "*An Architecture for a Scalable, High-Performance Digital Library,*" from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.

56. S. Louis and R.D. Burris, "*Management Issues for High-Performance Storage Systems,*" from the 1995 IEEE MSS Symposium, courtesy of the IEEE Computer Society Press.

57. D. Fisher, J. Sobolewski, and T. Tyler, "*Distributed Metadata Management in the High Performance Storage System,*" from the 1st IEEE Metadata Conference, April 16-18, 1996.

Appendix C

Developer Acknowledgments

HPSS is a product of a government-industry collaboration. The project approach is based on the premise that no single company, government laboratory, or research organization has the ability to confront all of the system-level issues that must be resolved for significant advancement in high-performance storage system technology.

HPSS development was performed jointly by IBM Worldwide Government Industry, Lawrence Berkeley National Laboratory, Lawrence Livermore National Laboratory, Los Alamos National Laboratory, NASA Langley Research Center, Oak Ridge National Laboratory, and Sandia National Laboratories.

We would like to acknowledge Argonne National Laboratory, the National Center for Atmospheric Research, and Pacific Northwest Laboratory for their help with initial requirements reviews.

We also wish to acknowledge Cornell Information Technologies of Cornell University for providing assistance with naming service and transaction management evaluations and for joint developments of the Name Server component.

We also wish to acknowledge the many discussions, design ideas, implementation and operation experiences we have shared with colleagues at the National Storage Laboratory, the IEEE Mass Storage Systems and Technology Technical Committee, the IEEE Storage System Standards Working Group, and the storage community at large.

We also wish to acknowledge the Cornell Theory Center and the Maui High Performance Computer Center for providing a test beds for the initial HPSS release.

Finally, we wish to acknowledge CEA-DAM (Commissariat à l’Énergie Atomique - Centre d’Études Bruyères-le-Châtel) for providing assistance with development of NFS V3 protocol support.

Appendix D

Accounting Examples

D.1 Introduction

This appendix describes how to set up the gathering of accounting data at a customer site. The accounting data is used by the customer to calculate charges for the use of HPSS resources. The accounting data represents a blurred snapshot of the system storage usage as it existed during the accounting run.

D.2 Site Accounting Requirements

What are the accounting requirements for your site? The accounting department at each site should be consulted to determine what kind of information it will need attached to each accounting record. Based on this, an HPSS Account Index for a user account can be set up in an Account Map to point to the required information.

Do you need information per user? Per account code? Per group? Some sites will need accounting totaled on a per user, per Account Index basis. This type of accounting is called Site-style accounting. There are many sites that will be required to implement charging in this way.

What kind of reports will be generated? Many UNIX resources report usage by user ID (UID). Some sites will need to use the UNIX-style accounting, which gives accounting totaled on a per user basis only. The accounting department will provide a program that takes accounting information on a per user (UID) basis and applies percentages to split out charges for multiple project accounts per user.

D.3 Processing of HPSS Accounting Data

How should the HPSS accounting data be processed? Should the data be written to a flat file or put into a site data base?

The default accounting output routine for HPSS generates a text file that contains two types of lines. The first type, denoted by a zero (0) in the third column, gives the following summary information about the storage used by a particular HPSS Account Index (AcctId) in a particular Class Of Service (COS):

- The total number of file accesses (#Accesses) to files owned by the Account Index in the Class Of Service. In general, file accesses are counted against the account of the user accessing the file, not the owner of the file itself.
- The total number of files (#Files) stored under the Account Index in the Class Of Service.
- The total amount of data stored (Length) under the Account Index in the Class Of Service.

The second type of line has a non-zero value in the Storage Class column (SClass). This type of line contains information about the storage used by a particular HPSS Account Index (AcctId) in a particular Storage Class (SClass) within a particular Class Of Service (COS). These line contain the following information:

- The total number of file accesses (#Accesses) in this particular Storage Class for the given Class Of Service. If a class of service is configured to stage bitfiles on open, then all file accesses will occur in the storage class at the top of the hierarchy.
- The total amount of data transferred (Transferred) into or out of this Storage Class and Class Of Service for this particular Account Index. Note that data transferred is counted against the owner of account transferring the data, not the owner of the data itself.

  Note: File transfer information is not always accounted for when it occurs through interfaces other than the HPSS Client API (i.e. HPSS NFS, HPSS DFS, etc.).

Example Accounting Report File:

```
# Comment file first line
# Comment file last line
# HPSS Accounting Snapshot completed on Wed Jul 15 12:57:00 1998
# Storage Unit Size : 1
# Total Number of Rows : 5
# Total Number of Accounts : 2
# Total Storage Units for HPSS system : 15598533
#
# Entries with ‘0’ in the SClass field are COS totals.
# Other entries apply to individual storage classes.
#
# AcctId   COS       0    #Accesses        #Files    Length  (COS)
# AcctId   COS  SClass    #Accesses   Transferred       (SClass)
# ---    -----   -----   ----------   -----------     ----------
  2033       3       0            0             5         212895
  634        1       0           89            89        4168147
  634        1       1           89            89
  634        5       0          152           152       11217491
  634        5       9          152           152
```

The HPSS accounting file will be correlated with the Account Map to determine the appropriate accounting charges. This is a customer site function.

Sites may wish to write a module that will redirect the accounting data into a local accounting data base. This module would replace the default HPSS module, **acct_WriteReport()**, which writes out the HPSS accounting data to a flat text file.

Where should the accounting data be stored? The HPSS accounting file and a copy of the current Account Map should be named with the date and time and stored for future reference. Individual sites should write scripts to copy / archive the generated accounting report file from its original location.

D.4 Site Accounting Table

What should be in the Account Map corresponding to the HPSS Account Index number? The HPSS Account Index will correspond to an Account Map entry that has the site information. The following are examples of possible Account Maps:

Site-style Account Map:

```
Acct     User  UID   Charge      Update_flag
12       dlk   3152  5A12x401    0
27       dlk   3152  5A12x501    0
341      dlk   3152  5A12x601    0
469      dpc   1478  7A14x401    0
470      dpc   1478  7A14x501    0
471      dmb   5674  5A12x401    0
7111     dmb   5674  7A14x501    0
...      ...   ...   ...         ...
```

UNIX-style Account Map

```
UID      Charge
1478     7A14x401
3152     5A12x401
5674     5A12x401
...      ...
```

If Site-style accounting is in use and Account Validation has been enabled, the user-to-account index mappings are already maintained in the Account Validation metadata file. If additional mappings are needed they will need to be kept in the Account Map file.

D.5 Account Apportionment Table

In UNIX-style accounting, the UID (as Account Index) maps only to a specific user. The mapping of a UID to various percentages of different project charge codes can be done by the site accounting department in an Account Apportionment Table as shown in the following example:

```
UID      % of (Project(s))
1478     75(DND) 25(CBC)
```

```
3152     45(DDI) 25(DND)30(CBC)
5674     100(DDI)
...      .................
```

Note: The Account Apportionment Table and Account Maps can be created by the individual sites. They are not created or maintained by HPSS. Some sites may wish to add more information, such as department and text name, or include less information, such as only the UID.

D.6 Maintaining and/or Modifying the Account Map

What are the policies for the Account Map? How will the Account Map be maintained and modified? Each site must develop tools to maintain and modify its Account Map according to local accounting policies. Tools will be necessary to change, add, or delete an HPSS Account Index and its associated information. The site must implement their own policies on what to do when an account is deleted, a user moves to another project, or a user leaves the system.

For Site-style accounting, when using Account Validation, the account validation editor may be used to maintain the user-to-account index mappings. Other needed mappings, if any, must be maintained by the site in the Account Map file. When Account Validation is disabled, the site will need to maintain the user-to-account index mappings in the Account Map file as well.

UNIX-style accounting changes of this nature are handled through the normal utilities that set up and modify users and UIDs. A basic set of site-developed utilities are described in more detail below.

- *Add a user and account*. New entries can be made and the next available HPSS Account Index number will be assigned from the free-list. The free-list will most likely consist of the last assigned number plus one, but could include reclaimed index numbers if a site chooses to re-use Account Index numbers that were previously assigned and no longer referenced. It is not likely that a site will need to reclaim Account Index numbers, but it is an option.

- *Delete a user.* When a user is deleted from the Account Map, the HPSS files must be reassigned to another HPSS Account Index. This should be done from the HPSS client interface side. The **update_flag** should be set to **true** to indicate that this account index number can be reclaimed. The reclaiming tool should check for files using the account number before reclaiming it. When the Account Index is reclaimed, it can be put on the free-list to be re-used. It is important to keep a copy of the Account Map that corresponds to the HPSS accounting snapshot of storage space in use for that time period so that the proper site information can be matched.

- *Delete account*. When an account is deleted, it is handled in the same manner as when a user is deleted.

- *Modify account*. The entries in the Account Map can be modified to correlate the Account Index to different information, but care should be taken to keep a copy of the corresponding tables for past HPSS accounting runs.

D.7 Accounting Reports

What kind of reports will be needed for your site? Learning what kind of accounting reports your site will need to generate will help you determine how detailed the collected accounting information should be. A typical Account Map will allow reports to be generated for the following:

- Total file accesses, amount of data transferred, and total space used per account, per class of service, per storage class.
- Total file accesses, amount of data transferred, and total space used per user, per class of service, per storage class.

D.8 Accounting Intervals and Charges

How often should HPSS accounting be run? How much should be charged for each unit of data transferred and each unit of space used? The time between accounting runs and the charging policy for space usage should be developed after consulting with the site accounting department. The following are some guidelines to consider:

- Accounting should be run at a regular intervals, such as once per month.
- An accounting run may take several minutes, and the storage system will probably be active during the run. The resource usage reported for each user will reflect the resources used by that user at the point when the accounting run encounters that user. This is why accounting represents a blurred snapshot instead of a snapshot at a single point in time.
- Certain accounting information is kept in cache for several minutes after it has changed. For this reason, changes to a user's accounting data may not appear in an accounting report until this several-minute period has elapsed. Those changes which are still in cache when accounting runs will not appear on the current accounting report, but will appear on the next accounting report.
- The number of file accesses and the amount of data transferred can be taken to represent the activity level of a certain user account in the HPSS system. You may wish to charge specifically for the network and server resources consumed by this activity.
- It may be useful to charge different rates for each Class of Service. For example, a Class of Service that keeps two tape copies of each file will use up more tape cartridges than a Class of Service that keeps only a single copy of each file on tape.

Appendix E

Infrastructure Configuration Example

E.1 AIX Infrastructure Configuration Example

```
% pwd
/opt/hpss/config
% mkhpss


<mkhpss>                Verify User ID
                        ==============

<mkhpss> Status ==> User root; verified; continue...


<mkhpss>                Perform HPSS Infrastructure Configuration:
                        ==========================================


<mkhpss> Status ==>   Platform: AIX
<mkhpss> Status ==>  Host Name: host.clearlake.ibm.com
<mkhpss> Status ==> Start Time: Fri Aug 10 11:36:58 CDT 2001




<mkhpss>                Set up HPSS Special Directories and Links
                        =========================================

on Fri Aug 10 11:36:58 CDT 2001

<mkhpss>                Infrastructure Configuration Menu
                        =================================

<mkhpss> Prompt ==>     Select Infrastructure Configuration Option:
<mkhpss>                [1] Configure HPSS with DCE
<mkhpss>                [2] Configure SFS Server
<mkhpss>                [3] Create and Manage SFS Files
<mkhpss>                [4] Set Up FTP Daemon
<mkhpss>                [5] Set Up Startup Daemon
<mkhpss>                [6] Add SSM Administrative User
<mkhpss>                [7] Start SSM Servers/User Session
```

```
<mkhpss>                [E] Re-run hpss_env()
<mkhpss>                [U] Un-configure HPSS
<mkhpss>                [X] Exit


<mkhpss> Reply ===> (Select Option [1-7, E, U, X]): 1

<mkhpss>                Perform DCE Set up
                        ==================





<mkhpss>                Verify DCE is Running
                        =====================

<mkhpss> Status ==> DCE is running, continue...
<mkhpss>                Perform DCE Register


<hpss_dce_register> Status => Running /opt/hpss/config/hpss_dce_register on host.clear-
lake.ibm.com by root on Fri Aug 10 11:41:51 CDT 2001
<hpss_dce_register> Prompt => Password for cell_admin:
<hpss_dce_register> Status => Create the groups, principals and accounts

<hpss_dce_register> Status => Create HPSS Server CDS directories

<hpss_dce_register> Status => Create server, client keytabs files

<hpss_dce_register> Status => Set permisiions on keytab files

<mkhpss>                Perform Set up and Check Cell
                        ============================



<hpss_setup_and_check_cell> Prompt => Password for cell_admin:
<hpss_setup_and_check_cell> Status => Found 1 Cell entries:
/.../host_cell.clearlake.ibm.com/krbtgt/host_cell.clearlake.ibm.com
<hpss_setup_and_check_cell> Status => Local Cell is /.../host_cell.clearlake.ibm.com
<hpss_setup_and_check_cell> Status => All HPSS ERAs created (or already existed).
 <hpss_setup_and_check_cell> Status => Be sure to populate the HPSS.homedir and hpss.gecos
for all customers.

This application tries to contact every cell in the
“hpss_cross_cell_members” group.

This test will indicate failures in the HPSS configuration
including:
                 A). The hpss_cross_cell_members group does NOT exist.
                 B) A krbtgt/... member in the “hpss_cross_cell_members”
                 group has no CellId ERA

Examine the output to be sure every cell that is expected to be a
Trusted Cross Cell appears in the output.

If the application stops prior to the completion message,
a cell is probably unreachable!
```

```
Check the network and DCE on both sides

Found: 1 Trusted Cell Members

TrustedCells[0].cell_id = b0e840f4-2206-11d5-9453-0004ac498ce4
TrustedCells[0].uid = 101
TrustedCells[0].cell_name = /.../host_cell.clearlake.ibm.com
TrustedCells[0].hpss_cell_id = 200090

/.../host_cell.clearlake.ibm.com successfully opened


Completed Cross Cell Trust Check

<mkhpss> Status ==> Configure HPSS with DCE completed, continue...




<mkhpss>                Infrastructure Configuration Menu
                        =================================

<mkhpss> Prompt ==>     Select Infrastructure Configuration Option:
<mkhpss>                [1] Configure HPSS with DCE
<mkhpss>                [2] Configure SFS Server
<mkhpss>                [3] Create and Manage SFS Files
<mkhpss>                [4] Set Up FTP Daemon
<mkhpss>                [5] Set Up Startup Daemon
<mkhpss>                [6] Add SSM Administrative User
<mkhpss>                [7] Start SSM Servers/User Session

<mkhpss>                [E] Re-run hpss_env()
<mkhpss>                [U] Un-configure HPSS
<mkhpss>                [X] Exit


<mkhpss> Reply ===> (Select Option [1-7, E, U, X]): 2


<mkhpss>                Perform Encina Register
                        =======================


<mkhpss> Prompt ==> HPSS Environment Table to use (/opt/hpss/config/hpss_env):
<mkhpss> Prompt ==> HPSS SFS Name to use (/.:/encina/sfs/hpss):

<mkhpss> Status ==> Creating local encina directories ...

<mkhpss> Status ==> Creating mirror encina directories ...

<mkhpss> Prompt ==> Password for cell_admin:

<mkhpss> Status ==> Creating group encina_admin_group ...

<mkhpss> Status ==> Creating group encina_servers_group ...

<mkhpss> Status ==> Creating principal encina_admin ...

<mkhpss> Status ==> Creating principal encina/sfs/hpss ...
```

```
<mkhpss> Status ==> Adding principal encina_admin to group encina_admin_group ...

<mkhpss> Status ==> Adding principal encina/sfs/hpss to group encina_admin_group ...

<mkhpss> Status ==> Adding principal encina/sfs/hpss to group encina_servers_group ...

<mkhpss> Status ==> Adding principal encina_admin to organization none ...

<mkhpss> Status ==> Adding principal encina/sfs/hpss to organization none ...

<mkhpss> Status ==> Creating account for encina_admin ...
<mkhpss> Prompt ==> Password for encina_admin:
<mkhpss> Status ==> Destroying credentials ...
<mkhpss> Prompt ==> Password for cell_admin:
<mkhpss> Prompt ==> Password for encina/sfs/hpss:
<mkhpss> Prompt ==> Reenter password for encina/sfs/hpss:
<mkhpss> Status ==> Creating account for encina/sfs/hpss ...

<mkhpss> Status ==> Destroying credentials ...
<mkhpss> Status ==> Creating keytab file /.../host_cell.clearlake.ibm.com/hosts/host/
config/keytab/hpss ...
<mkhpss> Status ==> Randomizing encina/sfs/hpss to keytab file ...

<mkhpss> Prompt ==> Password for cell_admin:
<mkhpss> Status ==> Creating cds directory /.:/encina ...

<mkhpss> Status ==> Modifying ACLs for  /.:/encina ...

<mkhpss> Status ==> Creating cds directory /.:/encina/trpc ...

<mkhpss> Status ==> Creating cds directory /.:/encina/sfs ...

<mkhpss> Status ==> Modifying ACLs for  /.:/encina/trpc ...

<mkhpss> Status ==> Modifying ACLs for  /.../host_cell.clearlake.ibm.com/host_ch ...

<mkhpss> Status ==> Destroying credentials ...



<mkhpss> Status ==> Cold starting SFS server ...
<mkhpss> Status ==> Looking for endpoint /.:/encina/sfs/hpss ...
<mkhpss> Status ==> Not found ..
<mkhpss> Status ==> Looking for endpoint /.:/encina/sfs/hpss ...
<mkhpss> Status ==> Found it ..
<mkhpss> Status ==> Pinging /.:/encina/sfs/hpss ....



 <mkhpss> Prompt ==> DCE Login (encina_admin):
 <mkhpss> Prompt ==> Password for encina_admin:

<mkhpss> Status ==> Mapping the SFS log volume ...
<mkhpss> Prompt ==> Enter AIX logical volume name to use (loglvhpss):
LOGICAL VOLUME:     loglvhpss              VOLUME GROUP:   uservg
LV IDENTIFIER:      00010816dec98a62.12    PERMISSION:     read/write
VG STATE:           active/complete        LV STATE:       closed/syncd
TYPE:               raw                    WRITE VERIFY:   off
MAX LPs:            512                    PP SIZE:        16 megabyte(s)
COPIES:             1                      SCHED POLICY:   parallel
```

```
LPs:                16                      PPs:            16
STALE PPs:          0                       BB POLICY:      relocatable
INTER-POLICY:       maximum                 RELOCATABLE:    yes
INTRA-POLICY:       middle                  UPPER BOUND:    1
MOUNT POINT:        N/A                     LABEL:          None
MIRROR WRITE CONSISTENCY: on
EACH LP COPY ON A SEPARATE PV ?: yes
<mkhpss> Prompt ==> Logical volume exists - use it, redefine it, or quit (u/d/q)(u):
<mkhpss> Prompt ==> SFS log volume name to use (logVol):
<mkhpss> Prompt ==> SFS chunk size to use (64):
<mkhpss> Prompt ==> SFS log file name to use (logFile):
<mkhpss> Status ==> Initializing SFS log volume ...
tkadmin init logvol -server /.:/encina/sfs/hpss logVol \          FILE:/opt/encinalocal/
encina/sfs/hpss/logArchive

<mkhpss> Status ==> Creating SFS log file ...
<mkhpss> Status ==> Enabling SFS log file logVol/logFile ...

<mkhpss> Status ==> Recovering SFS log volume ...

<mkhpss> Prompt ==> Do you want to enable mediaarchiving? (y/n)(n)
<mkhpss> Status ==> Enabling SFS server ...



<mkhpss> Status ==> Mapping SFS data volume ...
<mkhpss> Prompt ==> Enter AIX logical volume name to use (sfslvhpss):
LOGICAL VOLUME:     sfslvhpss               VOLUME GROUP:   uservg
LV IDENTIFIER:      00010816dec98a62.13     PERMISSION:     read/write
VG STATE:           active/complete         LV STATE:       closed/syncd
TYPE:               raw                     WRITE VERIFY:   off
MAX LPs:            512                     PP SIZE:        16 megabyte(s)
COPIES:             1                       SCHED POLICY:   parallel
LPs:                64                      PPs:            64
STALE PPs:          0                       BB POLICY:      relocatable
INTER-POLICY:       maximum                 RELOCATABLE:    yes
INTRA-POLICY:       middle                  UPPER BOUND:    1
MOUNT POINT:        N/A                     LABEL:          None
MIRROR WRITE CONSISTENCY: on
EACH LP COPY ON A SEPARATE PV ?: yes
<mkhpss> Prompt ==> Logical volume exists - use it, redefine it, or quit (u/d/q)(u):
<mkhpss> Prompt ==> SFS data volume name to use (dataVol):
<mkhpss> Prompt ==> SFS chunk size to use (64):
<mkhpss> Status ==> Enabling SFS data volume dataVol ...
tkadmin enable lvol -server /.:/encina/sfs/hpss dataVol

<mkhpss> Status ==> Adding SFS data volume dataVol ...
sfsadmin add lvol dataVol -server /.:/encina/sfs/hpss




<mkhpss> Status ==> Mapping SFS data volume ...
<mkhpss> Prompt ==> Do you want to create another SFS data volume? (y/n)(y)n
<mkhpss> Status ==> Waiting for SFS to export the ACL interface ...
<mkhpss> Status ==> Modifying ACLs for /.:/encina/sfs/hpss ...
<mkhpss> Status ==> Adding {group encina_admin_group ACQ} ...


<mkhpss> Status ==> Adding {group hpss_server ACQ} ...
```

```
<mkhpss> Status ==> Adding {user encina_admin ACQ} ...

<mkhpss> Status ==> Adding {user hosts/host/self ACQ} ...

<mkhpss> Status ==> Clearing exclusive authority ...

<mkhpss> Status ==> Stopping server ...

<mkhpss> Status ==> Destroying credentials ...



<mkhpss> Status ==> Preparing the SFS server for a warm start ...
<mkhpss> Status ==> Collating language to use (en_US):
<mkhpss> Status ==> Buffer pool size <kilobytes> to use (8192):
<mkhpss> Status ==> Idle timeout <seconds> to use (60):
<mkhpss> Status ==> Checkpoint interval <seconds> to use (60):
<mkhpss> Status ==> Checkpoint interval <min_records> to use (5000):
<mkhpss> Status ==> Checkpoint interval <max_records> to use (100000):
<mkhpss> Status ==> Thread pool <user> size to use (400):
<mkhpss> Status ==> Thread pool <emergency> size to use (20):
<mkhpss> Prompt ==> Do you want rc.encina to be invoked at system restart? (y/n)(y)
<mkhpss> Status ==> SFS configuration complete! ...

<mkhpss> Status ==> Configure HPSS with Encina completed, continue...




<mkhpss>                Infrastructure Configuration Menu
                        =================================

<mkhpss> Prompt ==>     Select Infrastructure Configuration Option:
<mkhpss>                [1] Configure HPSS with DCE
<mkhpss>                [2] Configure SFS Server
<mkhpss>                [3] Create and Manage SFS Files
<mkhpss>                [4] Set Up FTP Daemon
<mkhpss>                [5] Set Up Startup Daemon
<mkhpss>                [6] Add SSM Administrative User
<mkhpss>                [7] Start SSM Servers/User Session

<mkhpss>                [E] Re-run hpss_env()
<mkhpss>                [U] Un-configure HPSS
<mkhpss>                [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, U, X]): 3



<mkhpss>                Perform Manage SFS
                        ==================




<mkhpss>                Verify DCE is Running
```

```
                         =====================

<mkhpss> Status ==> DCE is running, continue...
<mkhpss> Prompt ==> Password for encina_admin:
Reading /opt/hpss/config/hpss_env
Enter CDS name of SFS to work with  [/.:/encina/sfs/hpss]
Querying SFS server /.:/encina/sfs/hpss

Root SFS server: /.:/encina/sfs/hpss
All SFS servers:
                 /.:/encina/sfs/hpss
Currently selected SFS server: /.:/encina/sfs/hpss
Currently selected data volume: dataVol
Filename extension: <none>

         1)  Create a file (on current SFS)
         2)  Create all global files (on root SFS)
         3)  Create all files for a storage subsystem (on current SFS)
         4)  Empty a file (on current SFS)
         5)  Destroy a file (on current SFS)
         6)  Query a file (on current SFS)
         7)  List files on current SFS
         8)  List files on all SFS servers
         9)  Select root SFS server
        10)  Select current SFS server
        11)  Add an SFS to work with
        12)  Remove an SFS from the list
        13)  Select current data volume
        14)  Change filename extension
Select operation (<RETURN> to exit): 9
1) /.:/encina/sfs/hpss
Select the root SFS server [1]:

Root SFS server: /.:/encina/sfs/hpss
All SFS servers:
                 /.:/encina/sfs/hpss
Currently selected SFS server: /.:/encina/sfs/hpss
Currently selected data volume: dataVol
Filename extension: <none>

         1)  Create a file (on current SFS)
         2)  Create all global files (on root SFS)
         3)  Create all files for a storage subsystem (on current SFS)
         4)  Empty a file (on current SFS)
         5)  Destroy a file (on current SFS)
         6)  Query a file (on current SFS)
         7)  List files on current SFS
         8)  List files on all SFS servers
         9)  Select root SFS server
        10)  Select current SFS server
        11)  Add an SFS to work with
        12)  Remove an SFS from the list
        13)  Select current data volume
        14)  Change filename extension
Select operation (<RETURN> to exit): 10
1) /.:/encina/sfs/hpss
Select the root SFS server [1]:


Root SFS server: /.:/encina/sfs/hpss
```

```
All SFS servers:
                 /.:/encina/sfs/hpss
Currently selected SFS server: /.:/encina/sfs/hpss
Currently selected data volume: dataVol
Filename extension: <none>

         1)  Create a file (on current SFS)
         2)  Create all global files (on root SFS)
         3)  Create all files for a storage subsystem (on current SFS)
         4)  Empty a file (on current SFS)
         5)  Destroy a file (on current SFS)
         6)  Query a file (on current SFS)
         7)  List files on current SFS
         8)  List files on all SFS servers
         9)  Select root SFS server
        10)  Select current SFS server
        11)  Add an SFS to work with
        12)  Remove an SFS from the list
        13)  Select current data volume
        14)  Change filename extension
Select operation (<RETURN> to exit): 13

Selecting a data volume for the current SFS server /.:/encina/sfs/hpss
Using single data volume dataVol

Root SFS server: /.:/encina/sfs/hpss
All SFS servers:
                 /.:/encina/sfs/hpss
Currently selected SFS server: /.:/encina/sfs/hpss
Currently selected data volume: dataVol
Filename extension: <none>

         1)  Create a file (on current SFS)
         2)  Create all global files (on root SFS)
         3)  Create all files for a storage subsystem (on current SFS)
         4)  Empty a file (on current SFS)
         5)  Destroy a file (on current SFS)
         6)  Query a file (on current SFS)
         7)  List files on current SFS
         8)  List files on all SFS servers
         9)  Select root SFS server
        10)  Select current SFS server
        11)  Add an SFS to work with
        12)  Remove an SFS from the list
        13)  Select current data volume
        14)  Change filename extension
Select operation (<RETURN> to exit): 2
Tentative assignment of files to data volumes on /.:/encina/sfs/hpss

dataVol:
   General Server Configurations     Migration Policies
   Accounting Policy                 Purge Policies
   Account Summary Records           Mount Daemon Configuration
   Account Snapshot Records          Mover Configurations
   Account Validation Records        Mover Devices
   BFS Configurations                NFS Daemon Configuration
   Classes of Service                PVL Configurations
   DMAP Gateway Configurations       PVL Activities
   DMAP Gateway File Sets            PVL Drives
   File Families                     PVL Jobs
   GK Configurations                  PVL Physical Volumes
   HPSS Global Configuration          PVR Configurations
   Storage Hierarchies                3494 Cartridges
   Non-DCE Gateway Configurations     3495 Cartridges
   NS Configurations                  AML Cartridges
   NS Global Filesets                 Operator Cartridges
   Log Client Configurations          STK Cartridges
   Log Daemon Configurations          STK Cartridges
   Logging Policies                   Remote Site Configurations
   Location Server Policies           Storage Server Configurations
   Metadata Monitor Configurations    Storage Classes
   Migration/Purge Configurations     Storage Subsystem Configuration
   Subsystem Storage Class Thresho


        1)  Reassign files between data volumes
        2)  Show mapping of file names to volumes
        3)  Create the global files as they are allocated
Select operation (<RETURN> to cancel): 3

...

New file nsconfig created
Creating “nsglobalfilesets”

New file nsglobalfilesets created
Creating “logclient”

New file logclient created
Creating “logdaemon”

New file logdaemon created
Creating “logpolicy”
...


Root SFS server: /.:/encina/sfs/hpss
All SFS servers:
                 /.:/encina/sfs/hpss
Currently selected SFS server: /.:/encina/sfs/hpss
Currently selected data volume: dataVol
Filename extension: <none>

         1)  Create a file (on current SFS)
         2)  Create all global files (on root SFS)
         3)  Create all files for a storage subsystem (on current SFS)
         4)  Empty a file (on current SFS)
         5)  Destroy a file (on current SFS)
         6)  Query a file (on current SFS)
         7)  List files on current SFS
         8)  List files on all SFS servers
         9)  Select root SFS server
        10)  Select current SFS server
        11)  Add an SFS to work with
        12)  Remove an SFS from the list
        13)  Select current data volume
        14)  Change filename extension
Select operation (<RETURN> to exit): 3
Type the number of the subsystem for which to create files [1]:
ssignment of files to data volumes on /.:/encina/sfs/hpss
```

```
dataVol:
   Account Log Records              NS Objects
   Migration Records                NS Text Extensions
   Purge Records                    Migration/Purge Checkpoints
   Bitfiles                         SS Disk Storage Maps
   Bitfile COS Changes              SS Tape Storage Maps
   Bitfile Tape Segments            SS Disk Storage Segments
   Bitfile Disk Segments            SS Tape Storage Segments
   Bitfile Disk Allocation Maps     SS Disk Physical Volumes
   BFS Storage Segment Checkpoints  SS Tape Physical Volumes
   BFS Storage Segment Unlinks      SS Disk Virtual Volumes
   NS ACL Extensions                SS Tape Virtual Volumes
   NS Fileset Attrs


         1)  Choose a different subsystem number
         2)  Reassign files between data volumes
         3)  Show mapping of file names to volumes
         4)  Create the subsystem files as they are allocated
Select operation (<RETURN> to cancel): 4
Creating files on volume dataVol on /.:/encina/sfs/hpss
                 Creating “acctlog.1”

New file acctlog.1 created
                 Creating “bfmigrrec.1

New file bitfile.1 created
Creating “bfcoschange.1”

...


Root SFS server: /.:/encina/sfs/hpss
All SFS servers:
                 /.:/encina/sfs/hpss
Currently selected SFS server: /.:/encina/sfs/hpss
Currently selected data volume: dataVol
Filename extension: <none>

         1)  Create a file (on current SFS)
         2)  Create all global files (on root SFS)
         3)  Create all files for a storage subsystem (on current SFS)
         4)  Empty a file (on current SFS)
         5)  Destroy a file (on current SFS)
         6)  Query a file (on current SFS)
         7)  List files on current SFS
         8)  List files on all SFS servers
         9)  Select root SFS server
        10)  Select current SFS server
        11)  Add an SFS to work with
        12)  Remove an SFS from the list
        13)  Select current data volume
        14)  Change filename extension
Select operation (<RETURN> to exit):



<mkhpss>                Infrastructure Configuration Menu
```

```
                        =================================

<mkhpss> Prompt ==>     Select Infrastructure Configuration Option:
<mkhpss>                [1] Configure HPSS with DCE
<mkhpss>                [2] Configure SFS Server
<mkhpss>                [3] Create and Manage SFS Files
<mkhpss>                [4] Set Up FTP Daemon
<mkhpss>                [5] Set Up Startup Daemon
<mkhpss>                [6] Add SSM Administrative User
<mkhpss>                [7] Start SSM Servers/User Session

<mkhpss>                [E] Re-run hpss_env()
<mkhpss>                [U] Un-configure HPSS
<mkhpss>                [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, U, X]): 4

<mkhpss>                Perform FTP Daemon Setup
                        ========================




<mkhpss>                Verify User ID
                        ==============

<mkhpss> Status ==> User root; verified; continue...
<hpss_ftpd_config> Status => Perform HPSS FTP Daemon setup

<hpss_ftpd_config> Prompt => Do you wish to use the default port IDs? (control port = 4021;
data port = 4020) (y/n)(y)


<mkhpss> Status ==> FTP Daemon setup completed, continue...


<mkhpss>                Infrastructure Configuration Menu
                        =================================

<mkhpss> Prompt ==>     Select Infrastructure Configuration Option:
<mkhpss>                [1] Configure HPSS with DCE
<mkhpss>                [2] Configure SFS Server
<mkhpss>                [3] Create and Manage SFS Files
<mkhpss>                [4] Set Up FTP Daemon
<mkhpss>                [5] Set Up Startup Daemon
<mkhpss>                [6] Add SSM Administrative User
<mkhpss>                [7] Start SSM Servers/User Session

<mkhpss>                [E] Re-run hpss_env()
<mkhpss>                [U] Un-configure HPSS
<mkhpss>                [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, U, X]): 5



<mkhpss>                Verify User ID
                        ==============

<mkhpss> Status ==> User root; verified; continue...
```

```
<mkhpss>                Perform HPSS Startup Daemon Setup
                        =================================



<mkhpss> Status ==> HPSS Startup Daemon will be invoked at system restart



<mkhpss>                Infrastructure Configuration Menu
                        =================================

<mkhpss> Prompt ==>     Select Infrastructure Configuration Option:
<mkhpss>                [1] Configure HPSS with DCE
<mkhpss>                [2] Configure SFS Server
<mkhpss>                [3] Create and Manage SFS Files
<mkhpss>                [4] Set Up FTP Daemon
<mkhpss>                [5] Set Up Startup Daemon
<mkhpss>                [6] Add SSM Administrative User
<mkhpss>                [7] Start SSM Servers/User Session

<mkhpss>                [E] Re-run hpss_env()
<mkhpss>                [U] Un-configure HPSS
<mkhpss>                [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, U, X]): 6

<mkhpss>                Add SSM User
                        ============




<mkhpss>                Verify DCE is Running
                        =====================

<mkhpss> Status ==> DCE is running, continue...
<mkhpss> Prompt ==> Password for cell_admin:

<mkhpss> Prompt ==> Enter SSM user id
<mkhpss> Reply ===> (user id:) hpss


DCE: Adding DCE User ‘hpss’ ...

Enter cell_admin password :


AIX: Adding UNIX User ‘hpss’ ...

Enter User’s Full Name [John Smith]: hpss
Enter User’s Password :
Retype User’s Password :
Enter UID [999]: 798
Enter Group [hpss]:
AIX: User ‘hpss’ already exists in the passwd file (not added)

SSM: Adding SSM User ‘hpss’ ...
```

```
<mkhpss>                Infrastructure Configuration Menu
                        =================================

<mkhpss> Prompt ==>     Select Infrastructure Configuration Option:
<mkhpss>                [1] Configure HPSS with DCE
<mkhpss>                [2] Configure SFS Server
<mkhpss>                [3] Create and Manage SFS Files
<mkhpss>                [4] Set Up FTP Daemon
<mkhpss>                [5] Set Up Startup Daemon
<mkhpss>                [6] Add SSM Administrative User
<mkhpss>                [7] Start SSM Servers/User Session

<mkhpss>                [E] Re-run hpss_env()
<mkhpss>                [U] Un-configure HPSS
<mkhpss>                [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, U, X]): 7

<mkhpss>                Start SSM Session
                        =================




<mkhpss>                Verify User ID
                        ==============

<mkhpss> Status ==> User root; verified; continue...
<mkhpss> Prompt ==> Do you wish to start SSM servers under user id root?
<mkhpss> Reply ===> (Y)

<mkhpss> Prompt ==> Do you wish to start SSM session under default user id? (hpss)
<mkhpss> Reply ===> (Y)

<mkhpss> Prompt ==> Enter SAM2_DISPLAY for displaying SSM session: hpss:0.0

...
```

E.2 AIX Infrastructure Un-Configuration Example

```
<mkhpss>                Infrastructure Configuration Menu
                        =================================

<mkhpss> Prompt ==>     Select Infrastructure Configuration Option:
<mkhpss>                [1] Configure HPSS with DCE
<mkhpss>                [2] Configure SFS Server
<mkhpss>                [3] Create and Manage SFS Files
<mkhpss>                [4] Set Up FTP Daemon
```

```
<mkhpss>                [5] Set Up Startup Daemon
<mkhpss>                [6] Add SSM Administrative User
<mkhpss>                [7] Start SSM Servers/User Session

<mkhpss>                [E] Re-run hpss_env()
<mkhpss>                [U] Un-configure HPSS
<mkhpss>                [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, U, X]):U

<mkhpss> Prompt ==>     Select Unconfiguration Option:

<mkhpss>                [1] Unconfigure encina
<mkhpss>                [2] Unconfigure an installation node

<mkhpss>                [M] Return to the Main Menu

<mkhpss> Reply ===> (Select Option [1-2, M]):2

<mkhpss> WARNING => ABOUT TO UN-CONFIGURE AN INSTALLATION NODE! ! ! ! !
<mkhpss> WARNING => ABOUT TO UN-CONFIGURE AN INSTALLATION NODE! ! ! ! !
<mkhpss> WARNING => ABOUT TO UN-CONFIGURE AN INSTALLATION NODE! ! ! ! !
<mkhpss> WARNING => -ALL- HPSS CONFIGURATION and METADATA WILL BE DESTROYED! ! ! ! !
<mkhpss> WARNING => -ALL- HPSS CONFIGURATION and METADATA WILL BE DESTROYED! ! ! ! !
<mkhpss> WARNING => -ALL- HPSS CONFIGURATION and METADATA WILL BE DESTROYED! ! ! ! !
<mkhpss> WARNING => (i.e., HPSS Server Principals, Keytabs, CDS Entries, SFS Files, ...
etc.,)
<mkhpss> Prompt ==> Do you wish to continue? (y/n) (n) y


<mkhpss>                Perform HPSS Un-configuration
                        =============================


<mkhpss> Prompt ==> Password for cell_admin:

<hpss_ftpd_unconfig>            Perform /opt/hpss/config/hpss_ftpd_unconfig

<hpss_ftpd_unconfig> Status => ftpd Unconfigured processing completed!

<mkhpss>                Perform Undo Startup Daemon Setup
                        =================================



<mkhpss> Status ==> Undo Startup Daemon Setup processing completed!

<mkhpss> Status ==> Remove encina/sfs/hpss from encina_admin_group

<mkhpss> Status ==> Remove encina/sfs/hpss from encina_servers_group

<mkhpss> Status ==> Remove encina/sfs/hpss from organzation none

<mkhpss> Status ==> Delete keytab file
/.../host_cell.clearlake.ibm.com/hosts/host.clearlake.ibm.com/config/keytab/hpss

<mkhpss> Status ==> Delete principal encina/sfs/hpss


<mkhpss> Status ==> Remove /opt/encinalocal/encina/sfs/hpss? (y/n)(y)
```

```
<mkhpss> Status ==> Remove /opt/encinamirror/encina/sfs/hpss? (y/n)(y)

<mkhpss> Prompt ==> Remove /opt/encinalocal? (y/n)(y)
<mkhpss> Prompt ==> Remove /opt/encinamirror? (y/n)(y)
<mkhpss>                Perform DCE Deregister

<hpss_dce_deregister> Status => Remove /.:/hpss? (y/n)(y)
<hpss_dce_deregister> Status => Removing keytab file, /krb5/hpss.keytabs.

<hpss_dce_deregister> Status => Removing keytab file, /krb5/hpssclient.keytab.

<hpss_dce_deregister> Status => Principals for hpss_server deleted
<hpss_dce_deregister> Status => Group, hpss_server and hpss_client, deleted
<hpss_dce_deregister> Status => DCE Deregister processing completed!

<mkhpss>                Perform Undo Init Setup

<mkhpss> Status ==> undo_init_setup processing completed!

[root@host:/opt/hpss/config]
```

E.3 Solaris Infrastructure Configuration Example

```
<mkhpss>                Infrastructure Configuration Menu
                        =================================

<mkhpss> Prompt ==>     Select Infrastructure Configuration Option:
<mkhpss>                [1] Configure HPSS with DCE
<mkhpss>                [2] Configure SFS Server
<mkhpss>                [3] Create and Manage SFS Files
<mkhpss>                [4] Set Up FTP Daemon
<mkhpss>                [5] Set Up Startup Daemon
<mkhpss>                [6] Add SSM Administrative User
<mkhpss>                [7] Start SSM Servers/User Session

<mkhpss>                [E] Re-run hpss_env()
<mkhpss>                [U] Un-configure HPSS
<mkhpss>                [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, U, X]):1


<mkhpss>                Perform DCE Set up
                        ==================




<mkhpss>                Verify DCE is Running
                        =====================
<mkhpss> Status ==> DCE is running, continue...
<mkhpss>                Perform DCE Register



<hpss_dce_register> Status => Running /opt/hpss/config/hpss_dce_register on host by root
on Tue Aug 28 11:36:22 CDT 2001
<hpss_dce_register> Prompt => Password for cell_admin:
<hpss_dce_register> Prompt => Remove /krb5/hpss.keytabs? (y/n)(y)
<hpss_dce_register> Prompt => Remove /krb5/hpssclient.keytab? (y/n)(y)
<hpss_dce_register> Status => Create the groups, principals and accounts


<hpss_dce_register> Status => Create HPSS Server CDS directories

<hpss_dce_register> Status => Create server, client keytabs files

<hpss_dce_register> Status => Set permisiions on keytab files

<mkhpss>                Perform Set up and Check Cell
                        ============================
<hpss_setup_and_check_cell> Status => Found 1 Cell entries:
/.../host_cell.clearlake.ibm.com/krbtgt/host_cell.clearlake.ibm.com
<hpss_setup_and_check_cell> Status => Local Cell is /.../host_cell.clearlake.ibm.com
<hpss_setup_and_check_cell> Status => Extended Registry Attribute, HPSS.CellId , NOT
Defined”
<hpss_setup_and_check_cell> Status => Creating HPSS.CellId ERA

<hpss_setup_and_check_cell> Status => Creating hpss_cross_cell_members ERA

<hpss_setup_and_check_cell> Status => DCE group, hpss_cross_cell_members, Empty
<hpss_setup_and_check_cell> Status => The Local Cell has NO HPSS.CellId Extended Registry
Attribute
<hpss_setup_and_check_cell> Status => Please obtain your HPSS.CellId from IBM
<hpss_setup_and_check_cell> Prompt => Enter Assigned HPSS.CellId:200040

<hpss_setup_and_check_cell> Status => Extended Registry Attribute, HPSS.homedir , NOT
Defined”
<hpss_setup_and_check_cell> Status => Creating HPSS.homedir ERA

<hpss_setup_and_check_cell> Status => Extended Registry Attribute, HPSS.gecos , NOT
Defined”
<hpss_setup_and_check_cell> Status => Creating HPSS.gecos ERA

<hpss_setup_and_check_cell> Status => All HPSS ERAs created (or already existed).
 <hpss_setup_and_check_cell> Status => Be sure to populate the HPSS.homedir and hpss.gecos
for all customers.

This application tries to contact every cell in the
“hpss_cross_cell_members” group.

This test will indicate failures in the HPSS configuration
including:
                 A). The hpss_cross_cell_members group does NOT exist.
                 B) A krbtgt/... member in the “hpss_cross_cell_members”
           group has no CellId ERA

Examine the output to be sure every cell that is expected to be a
Trusted Cross Cell appears in the output.
```

```
If the application stops prior to the completion message,
a cell is probably unreachable!
Check the network and DCE on both sides

Found: 1 Trusted Cell Members

TrustedCells[0].cell_id = b499b8ba-98d3-11d5-8d0d-c05e2fc1aa77
TrustedCells[0].uid = 101
TrustedCells[0].cell_name = /.../host_cell.clearlake.ibm.com
TrustedCells[0].hpss_cell_id = 200040


/.../host_cell.clearlake.ibm.com successfully opened


Completed Cross Cell Trust Check

<mkhpss> Status ==> Configure HPSS with DCE completed, continue...

<mkhpss>                Infrastructure Configuration Menu
                        =================================

<mkhpss> Prompt ==>     Select Infrastructure Configuration Option:
<mkhpss>                [1] Configure HPSS with DCE
<mkhpss>                [2] Configure SFS Server
<mkhpss>                [3] Create and Manage SFS Files
<mkhpss>                [4] Set Up FTP Daemon
<mkhpss>                [5] Set Up Startup Daemon
<mkhpss>                [6] Add SSM Administrative User
<mkhpss>                [7] Start SSM Servers/User Session

<mkhpss>                [E] Re-run hpss_env()
<mkhpss>                [U] Un-configure HPSS
<mkhpss>                [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, U, X]):2


<mkhpss>                Perform Encina Register
                        =======================


<mkhpss> Prompt ==> HPSS Environment Table to use (/opt/hpss/config/hpss_env):
<mkhpss> Prompt ==> HPSS SFS Name to use (/.:/encina/sfs/hpss):

<mkhpss> Status ==> Creating local encina directories ...

<mkhpss> Status ==> Creating mirror encina directories ...



<mkhpss> Prompt ==> Password for cell_admin:

<mkhpss> Status ==> Creating group encina_admin_group ...

<mkhpss> Status ==> Creating group encina_servers_group ...

<mkhpss> Status ==> Creating principal encina_admin ...

<mkhpss> Status ==> Creating principal encina/sfs/hpss ...
```

```
<mkhpss> Status ==> Adding principal encina_admin to group encina_admin_group ...

<mkhpss> Status ==> Adding principal encina/sfs/hpss to group encina_admin_group ...

<mkhpss> Status ==> Adding principal encina/sfs/hpss to group encina_servers_group ...

<mkhpss> Status ==> Adding principal encina_admin to organization none ...

<mkhpss> Status ==> Adding principal encina/sfs/hpss to organization none ...

<mkhpss> Status ==> Creating account for encina_admin ...
<mkhpss> Prompt ==> Password for encina_admin:
<mkhpss> Prompt ==> Reenter password for encina_admin:

<mkhpss> Prompt ==> Password for encina/sfs/hpss:

<mkhpss> Status ==> Creating account for encina/sfs/hpss ...

<mkhpss> Status ==> Destroying credentials ...
<mkhpss> Status ==> Creating keytab file
/.../host_cell.clearlake.ibm.com/hosts/host.clearlake.ibm.com/config/keytab/hpss ...
<mkhpss> Status ==> Randomizing encina/sfs/hpss to keytab file ...

<mkhpss> Prompt ==> Password for cell_admin:

<mkhpss> Status ==> Creating cds directory /.:/encina ...

<mkhpss> Status ==> Modifying ACLs for  /.:/encina ...

<mkhpss> Status ==> Creating cds directory /.:/encina/trpc ...

<mkhpss> Status ==> Creating cds directory /.:/encina/sfs ...

<mkhpss> Status ==> Modifying ACLs for  /.:/encina/trpc ...

<mkhpss> Status ==> Modifying ACLs for
/.../host_cell.clearlake.ibm.com/host.clearlake.ibm.com_ch ...

<mkhpss> Status ==> Destroying credentials ...



<mkhpss> Status ==> Cold starting SFS server ...
<mkhpss> Status ==> Looking for endpoint /.:/encina/sfs/hpss ...
<mkhpss> Status ==> Not found ..
<mkhpss> Status ==> Looking for endpoint /.:/encina/sfs/hpss ...
<mkhpss> Status ==> Found it ..
<mkhpss> Status ==> Pinging /.:/encina/sfs/hpss ....


<mkhpss> Prompt ==> DCE Login (encina_admin):
<mkhpss> Prompt ==> Password for encina_admin:

<mkhpss> Status ==> Mapping the SFS log volume ...
<mkhpss> Prompt ==> Enter disk name to use for log volume: /dev/rdsk/c0t8d0s1
<mkhpss> Prompt ==> Enter physical volume name (logpvhpss):
<mkhpss> Prompt ==> SFS log volume name to use (logVol):
<mkhpss> Prompt ==> SFS chunk size to use (64):
<mkhpss> Prompt ==> SFS log file name to use (logFile):
```

```
<mkhpss> Status ==> Init disk ...
tkadmin init disk -server /.:/encina/sfs/hpss /dev/rdsk/c0t8d0s1
Initialized disk partition /dev/rdsk/c0t8d0s1
disk size (in pages): 128401
<mkhpss> Status ==> Create a physical volume of /dev/rdsk/c0t8d0s1 ...
tkadmin create pvol logpvhpss 64 1            /dev/rdsk/c0t8d0s1 0
<mkhpss> Status ==> Create a logical volume of logpvhpss ...
tkadmin create lvol logVol logpvhpss

<mkhpss> Prompt ==> Do you want to mirror the logical volume logVol? (y/n)(y)n
<mkhpss> Status ==> Initializing SFS log volume ...
tkadmin init logvol -server /.:/encina/sfs/hpss logVol \
FILE:/opt/encinalocal/encina/sfs/hpss/logArchive

<mkhpss> Status ==> Creating SFS log file ...
<mkhpss> Status ==> Enabling SFS log file logVol/logFile ...

<mkhpss> Status ==> Recovering SFS log volume ...

<mkhpss> Prompt ==> Do you want to enable mediaarchiving? (y/n)(n)

<mkhpss> Status ==> Enabling SFS server ...



<mkhpss> Status ==> Mapping SFS data volume ...
<mkhpss> Prompt ==> Enter disk name to use for data volume: /dev/rdsk/c0t8d0s3
<mkhpss> Prompt ==> Enter physical volume name (sfspvhpss):
<mkhpss> Prompt ==> SFS data volume name to use (dataVol):
<mkhpss> Prompt ==> SFS chunk size to use (64):
<mkhpss> Status ==> Init disk ...
tkadmin init disk  -server /.:/encina/sfs/hpss /dev/rdsk/c0t8d0s3
Initialized disk partition /dev/rdsk/c0t8d0s3
disk size (in pages): 65967
<mkhpss> Status ==> Create a physical volume of /dev/rdsk/c0t8d0s3 ...
tkadmin create pvol -server /.:/encina/sfs/hpss sfspvhpss 64 1 /dev/rdsk/c0t8d0s3 0
<mkhpss> Status ==> Create a logical volume of sfspvhpss ...
tkadmin create lvol dataVol sfspvhpss


<mkhpss> Prompt ==> Do you want to mirror the logical volume dataVol? (y/n)(y)n

<mkhpss> Status ==> Enabling SFS data volume dataVol ...
tkadmin enable lvol -server /.:/encina/sfs/hpss dataVol

<mkhpss> Status ==> Adding SFS data volume dataVol ...
sfsadmin add lvol dataVol -server /.:/encina/sfs/hpss




<mkhpss> Status ==> Mapping SFS data volume ...
<mkhpss> Prompt ==> Do you want to create another SFS data volume? (y/n)(y)n
<mkhpss> Status ==> Waiting for SFS to export the ACL interface ...
<mkhpss> Status ==> Modifying ACLs for /.:/encina/sfs/hpss ...
<mkhpss> Status ==> Adding {group encina_admin_group ACQ} ...


<mkhpss> Status ==> Adding {group hpss_server ACQ} ...
```

```
<mkhpss> Status ==> Adding {user encina_admin ACQ} ...

<mkhpss> Status ==> Adding {user hosts/host.clearlake.ibm.com/self ACQ} ...

<mkhpss> Status ==> Clearing exclusive authority ...

<mkhpss> Status ==> Stopping server ...

<mkhpss> Status ==> Destroying credentials ...

<mkhpss> Status ==> Preparing the SFS server for a warm start ...
<mkhpss> Status ==> Collating language to use (en_US):
<mkhpss> Status ==> Buffer pool size <kilobytes> to use (8192):
<mkhpss> Status ==> Idle timeout <seconds> to use (60):
<mkhpss> Status ==> Checkpoint interval <seconds> to use (60):
<mkhpss> Status ==> Checkpoint interval <min_records> to use (5000):
<mkhpss> Status ==> Checkpoint interval <max_records> to use (100000):
<mkhpss> Status ==> Thread pool <user> size to use (400):
<mkhpss> Status ==> Thread pool <emergency> size to use (20):
<mkhpss> Prompt ==> Do you want rc.encina to be invoked at system restart? (y/n)(y)
<mkhpss> Status ==> SFS configuration complete! ...

<mkhpss> Status ==> Configure HPSS with Encina completed, continue...



<mkhpss>                Infrastructure Configuration Menu
                        =================================

<mkhpss> Prompt ==>     Select Infrastructure Configuration Option:
<mkhpss>                [1] Configure HPSS with DCE
<mkhpss>                [2] Configure SFS Server
<mkhpss>                [3] Create and Manage SFS Files
<mkhpss>                [4] Set Up FTP Daemon
<mkhpss>                [5] Set Up Startup Daemon
<mkhpss>                [6] Add SSM Administrative User
<mkhpss>                [7] Start SSM Servers/User Session

<mkhpss>                [E] Re-run hpss_env()
<mkhpss>                [U] Un-configure HPSS
<mkhpss>                [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, U, X]):3

<mkhpss>                Perform Manage SFS
                        ==================






<mkhpss>                Verify DCE is Running
                        =====================

<mkhpss> Status ==> DCE is running, continue...
<mkhpss> Prompt ==> Password for encina_admin:

Reading /opt/hpss/config/hpss_env
```

```
Enter CDS name of SFS to work with  [/.:/encina/sfs/hpss]
Querying SFS server /.:/encina/sfs/hpss

Root SFS server: /.:/encina/sfs/hpss
All SFS servers:
                 /.:/encina/sfs/hpss
Currently selected SFS server: /.:/encina/sfs/hpss
Currently selected data volume: dataVol
Filename extension: <none>

                  1)  Create a file (on current SFS)
                  2)  Create all global files (on root SFS)
                  3)  Create all files for a storage subsystem (on current SFS)
                  4)  Empty a file (on current SFS)
                  5)  Destroy a file (on current SFS)
                  6)  Query a file (on current SFS)
                  7)  List files on current SFS
                  8)  List files on all SFS servers
                  9)  Select root SFS server
                 10)  Select current SFS server
                 11)  Add an SFS to work with
                 12)  Remove an SFS from the list
                 13)  Select current data volume
                 14)  Change filename extension
Select operation (<RETURN> to exit): 2

Tentative assignment of files to data volumes on /.:/encina/sfs/hpss

dataVol:
   General Server Configurations     Migration Policies
   Accounting Policy                 Purge Policies
   Account Summary Records           Mount Daemon Configuration
   Account Snapshot Records          Mover Configurations
   Account Validation Records        Mover Devices
   BFS Configurations                NFS Daemon Configuration
   Classes of Service                PVL Configurations
   DMAP Gateway Configurations       PVL Activities
   DMAP Gateway File Sets            PVL Drives
   File Families                     PVL Jobs
   GK Configurations                 PVL Physical Volumes
   HPSS Global Configuration         PVR Configurations
   Storage Hierarchies               3494 Cartridges
   Non-DCE Gateway Configurations    3495 Cartridges
   NS Configurations                 AML Cartridges
   NS Global Filesets                Operator Cartridges
   Log Client Configurations         STK Cartridges
   Log Daemon Configurations         STK RAIT Cartridges
   Logging Policies                  LTO Cartridges
   Location Server Policies          Remote Site Configurations
   Metadata Monitor Configurations   Storage Server Configurations
   Migration/Purge Configurations    Storage Classes
   Subsystem Storage Class Thresho   Storage Subsystem Configuration


         1)  Reassign files between data volumes
         2)  Show mapping of file names to volumes
         3)  Create the global files as they are allocated
Select operation (<RETURN> to cancel):

Creating files on volume dataVol on /.:/encina/sfs/hpss
```

```
                 Creating “serverconfig”

New file serverconfig created

                 Creating “accounting”

New file accounting created
        ...

Root SFS server: /.:/encina/sfs/hpss
All SFS servers:
                 /.:/encina/sfs/hpss
Currently selected SFS server: /.:/encina/sfs/hpss
Currently selected data volume: dataVol
Filename extension: <none>

         1)  Create a file (on current SFS)
         2)  Create all global files (on root SFS)
         3)  Create all files for a storage subsystem (on current SFS)
         4)  Empty a file (on current SFS)
         5)  Destroy a file (on current SFS)
         6)  Query a file (on current SFS)
         7)  List files on current SFS
         8)  List files on all SFS servers
         9)  Select root SFS server
        10)  Select current SFS server
        11)  Add an SFS to work with
        12)  Remove an SFS from the list
        13)  Select current data volume
        14)  Change filename extension
Select operation (<RETURN> to exit): 3
Type the number of the subsystem for which to create files [1]:
Tentative assignment of files to data volumes on /.:/encina/sfs/hpss

dataVol:
   Account Log Records              NS Objects
   Migration Records                NS Text Extensions
   Purge Records                    Migration/Purge Checkpoints
   Bitfiles                         SS Disk Storage Maps
   Bitfile COS Changes              SS Tape Storage Maps
   Bitfile Tape Segments            SS Disk Storage Segments
   Bitfile Disk Segments            SS Tape Storage Segments
   Bitfile Disk Allocation Maps     SS Disk Physical Volumes
   BFS Storage Segment Checkpoints  SS Tape Physical Volumes
   BFS Storage Segment Unlinks      SS Disk Virtual Volumes
   NS ACL Extensions                SS Tape Virtual Volumes
   NS Fileset Attrs


         1)  Choose a different subsystem number
         2)  Reassign files between data volumes
         3)  Show mapping of file names to volumes
         4)  Create the subsystem files as they are allocated
Select operation (<RETURN> to cancel): 4
Creating files on volume dataVol on /.:/encina/sfs/hpss
                 Creating “acctlog.1”

New file acctlog.1 created
                 Creating “bfmigrrec.1”
```

```
New file bfmigrrec.1 created

        ...

oot SFS server: /.:/encina/sfs/hpss
All SFS servers:
                 /.:/encina/sfs/hpss
Currently selected SFS server: /.:/encina/sfs/hpss
Currently selected data volume: dataVol
Filename extension: <none>

         1)  Create a file (on current SFS)
         2)  Create all global files (on root SFS)
         3)  Create all files for a storage subsystem (on current SFS)
         4)  Empty a file (on current SFS)
         5)  Destroy a file (on current SFS)
         6)  Query a file (on current SFS)
         7)  List files on current SFS
         8)  List files on all SFS servers
         9)  Select root SFS server
        10)  Select current SFS server
        11)  Add an SFS to work with
        12)  Remove an SFS from the list
        13)  Select current data volume
        14)  Change filename extension
Select operation (<RETURN> to exit):



<mkhpss>                Infrastructure Configuration Menu
                        =================================

<mkhpss> Prompt ==>     Select Infrastructure Configuration Option:
<mkhpss>                [1] Configure HPSS with DCE
<mkhpss>                [2] Configure SFS Server
<mkhpss>                [3] Create and Manage SFS Files
<mkhpss>                [4] Set Up FTP Daemon
<mkhpss>                [5] Set Up Startup Daemon
<mkhpss>                [6] Add SSM Administrative User
<mkhpss>                [7] Start SSM Servers/User Session

<mkhpss>                [E] Re-run hpss_env()
<mkhpss>                [U] Un-configure HPSS
<mkhpss>                [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, U, X]):4

<mkhpss>                Perform FTP Daemon Setup
                        ========================




<mkhpss>                Verify User ID
                        ==============

<mkhpss> Status ==> User root; verified; continue...
<hpss_ftpd_config> Status => Perform HPSS FTP Daemon setup

<hpss_ftpd_config> Prompt => Do you wish to use the default port IDs? (control port = 4021;
data port = 4020) (y/n)(y)


<mkhpss> Status ==> FTP Daemon setup completed, continue...


<mkhpss>                Infrastructure Configuration Menu
                        =================================

<mkhpss> Prompt ==>     Select Infrastructure Configuration Option:
<mkhpss>                [1] Configure HPSS with DCE
<mkhpss>                [2] Configure SFS Server
<mkhpss>                [3] Create and Manage SFS Files
<mkhpss>                [4] Set Up FTP Daemon
<mkhpss>                [5] Set Up Startup Daemon
<mkhpss>                [6] Add SSM Administrative User
<mkhpss>                [7] Start SSM Servers/User Session

<mkhpss>                [E] Re-run hpss_env()
<mkhpss>                [U] Un-configure HPSS
<mkhpss>                [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, U, X]):5

<mkhpss>                Verify User ID
                        ==============

<mkhpss> Status ==> User root; verified; continue...


<mkhpss>                Perform HPSS Startup Daemon Setup
                        =================================



<mkhpss> Status ==> HPSS Startup Daemon will be invoked at system restart


<mkhpss>                Infrastructure Configuration Menu
                        =================================

<mkhpss> Prompt ==>     Select Infrastructure Configuration Option:
<mkhpss>                [1] Configure HPSS with DCE
<mkhpss>                [2] Configure SFS Server
<mkhpss>                [3] Create and Manage SFS Files
<mkhpss>                [4] Set Up FTP Daemon
<mkhpss>                [5] Set Up Startup Daemon
<mkhpss>                [6] Add SSM Administrative User
<mkhpss>                [7] Start SSM Servers/User Session

<mkhpss>                [E] Re-run hpss_env()
<mkhpss>                [U] Un-configure HPSS
<mkhpss>                [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, U, X]):6

<mkhpss>                Add SSM User
                        ============
<mkhpss>                Verify DCE is Running
                        =====================

<mkhpss> Status ==> DCE is running, continue...
<mkhpss> Prompt ==> Password for cell_admin:

<mkhpss> Prompt ==> Enter SSM user id
<mkhpss> Reply ===> (user id:)hpss

DCE: Adding DCE User ‘hpss’ ...

Enter cell_admin password :

Enter User’s Full Name [John Smith]: hpss

Enter User’s Password :
Retype User’s Password :
Enter UID [999]: 798
Enter Group [hpss]:
Enter Organization [hpss]: none

DCE: User ‘hpss’ created

SunOS: Adding UNIX User ‘hpss’ ...
Enter Group [hpss]:

SSM: Adding SSM User ‘hpss’ ...
SSM: [ Hostname = host : Full hostname = host.clearlake.ibm.com ]
Select SAMMI Security Level :
   1. User
   2. Privileged User
   3. Operator
   4. Admin
Enter Security Level [4]:

SSM: User ‘hpss’ added.
SSM: The Data Server needs to be recycled for the changes to take effect.



<mkhpss>                Infrastructure Configuration Menu
                        =================================

<mkhpss> Prompt ==>     Select Infrastructure Configuration Option:
<mkhpss>                [1] Configure HPSS with DCE
<mkhpss>                [2] Configure SFS Server
<mkhpss>                [3] Create and Manage SFS Files
<mkhpss>                [4] Set Up FTP Daemon
<mkhpss>                [5] Set Up Startup Daemon
<mkhpss>                [6] Add SSM Administrative User
<mkhpss>                [7] Start SSM Servers/User Session

<mkhpss>                [E] Re-run hpss_env()
<mkhpss>                [U] Un-configure HPSS
<mkhpss>                [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, U, X]):7
```

```
<mkhpss>                Start SSM Session
                        =================




<mkhpss>                Verify User ID
                        ==============

<mkhpss> Status ==> User root; verified; continue...
<mkhpss> Prompt ==> Do you wish to start SSM servers under user id root?
<mkhpss> Reply ===> (Y)

<mkhpss> Prompt ==> Do you wish to start SSM session under default user id? (hpss)
                                                        s)
<mkhpss> Reply ===> (Y)

<mkhpss> Prompt ==> Enter SAM2_DISPLAY for displaying SSM session: hpss:0.0
```

E.4 Solaris Infrastructure Un-Configuration Example

```
<mkhpss>                Infrastructure Configuration Menu
                        =================================

<mkhpss> Prompt ==>     Select Infrastructure Configuration Option:
<mkhpss>                [1] Configure HPSS with DCE
<mkhpss>                [2] Configure SFS Server
<mkhpss>                [3] Create and Manage SFS Files
<mkhpss>                [4] Set Up FTP Daemon
<mkhpss>                [5] Set Up Startup Daemon
<mkhpss>                [6] Add SSM Administrative User
<mkhpss>                [7] Start SSM Servers/User Session

<mkhpss>                [E] Re-run hpss_env()
<mkhpss>                [U] Un-configure HPSS
<mkhpss>                [X] Exit

<mkhpss> Reply ===> (Select Option [1-7, E, U, X]):U

<mkhpss> Prompt ==>     Select Unconfiguration Option:

<mkhpss>                [1] Unconfigure encina
<mkhpss>                [2] Unconfigure an installation node

<mkhpss>                [M] Return to the Main Menu

<mkhpss> Reply ===> (Select Option [1-2, M]):2

<mkhpss> WARNING => ABOUT TO UN-CONFIGURE AN INSTALLATION NODE! ! ! ! !
<mkhpss> WARNING => ABOUT TO UN-CONFIGURE AN INSTALLATION NODE! ! ! ! !
```

```
<mkhpss> WARNING => ABOUT TO UN-CONFIGURE AN INSTALLATION NODE! ! ! ! !
<mkhpss> WARNING => -ALL- HPSS CONFIGURATION and METADATA WILL BE DESTROYED! ! ! ! !
<mkhpss> WARNING => -ALL- HPSS CONFIGURATION and METADATA WILL BE DESTROYED! ! ! ! !
<mkhpss> WARNING => -ALL- HPSS CONFIGURATION and METADATA WILL BE DESTROYED! ! ! ! !
<mkhpss> WARNING => (i.e., HPSS Server Principals, Keytabs, CDS Entries, SFS Files, ...
etc.,)
<mkhpss> Prompt ==> Do you wish to continue? (y/n) (n) y


<mkhpss>                Perform HPSS Un-configuration
                        =============================



<mkhpss> Prompt ==> Password for cell_admin:

<hpss_ftpd_unconfig>            Perform /opt/hpss/config/hpss_ftpd_unconfig

<hpss_ftpd_unconfig> Status => ftpd Unconfigured processing completed!

<mkhpss>                Perform Undo Startup Daemon Setup
                        =================================




<mkhpss> Status ==> Undo Startup Daemon Setup processing completed!

<mkhpss> Status ==> Remove encina/sfs/hpss from encina_admin_group

<mkhpss> Status ==> Remove encina/sfs/hpss from encina_servers_group

<mkhpss> Status ==> Remove encina/sfs/hpss from organzation none

<mkhpss> Status ==> Delete keytab file
/.../host_cell.clearlake.ibm.com/hosts/host.clearlake.ibm.com/config/keytab/hpss

<mkhpss> Status ==> Delete principal encina/sfs/hpss


<mkhpss> Status ==> Remove /opt/encinalocal/encina/sfs/hpss? (y/n)(y)

<mkhpss> Status ==> Remove /opt/encinamirror/encina/sfs/hpss? (y/n)(y)

<mkhpss> Prompt ==> Remove /opt/encinalocal? (y/n)(y)
<mkhpss> Prompt ==> Remove /opt/encinamirror? (y/n)(y)
<mkhpss>                Perform DCE Deregister

<hpss_dce_deregister> Status => Remove /.:/hpss? (y/n)(y)
<hpss_dce_deregister> Status => Removing keytab file, /krb5/hpss.keytabs.

<hpss_dce_deregister> Status => Removing keytab file, /krb5/hpssclient.keytab.

<hpss_dce_deregister> Status => Principals for hpss_server deleted
<hpss_dce_deregister> Status => Group, hpss_server and hpss_client, deleted
<hpss_dce_deregister> Status => DCE Deregister processing completed!

<mkhpss>                Perform Undo Init Setup

<mkhpss> Status ==> undo_init_setup processing completed!

[root@host:/opt/hpss/config]
```

Appendix F

Additional SSM Information

F.1 Using the SSM Windows

Before using the SSM windows, it is helpful to be aware of some of the conventions used by SSM and by Sammi (on which SSM is based). While the following list does not cover all features of all windows, it does describe the most important points.

- Almost all mouse actions should be performed with the left button. One exception is opening help windows (see Section F.2). Some windows (very few) may use other buttons for special purposes. For example, the HPSS Alarms and Events window uses the left mouse button to select a message for detailed viewing, while the right mouse button is for acknowledging alarms. Such exceptions are documented in the online help.

- Colors are used consistently from window to window. While it is possible for these colors to be customized, the default colors will be used in descriptions below.

- All buttons are goldenrod (a pale orange) in color. Plain buttons perform a single action when clicked with a mouse. Toggle buttons have a small square indicator on the left, and are used to select one of two possible states. When the indicator is red, the button is **ON**; otherwise the button is **OFF**.

Buttons may be "desensitized". In this state, a button is visible, but its label text is grayed out, and clicking it has no effect. This happens when the current state of the window does not allow the button to be clicked. It also happens when your SSM security level is such that you are not permitted to do any more than view the button (see Section 11.1: *SSM Security* on page 275 of the *HPSS Management Guide*).

- Some buttons are labelled only with three periods, "**...**". Clicking such a button opens a window giving more information on the item next to it. For example, if a "**...**" button is to the right of a field displaying a storage class name, clicking it will open the Storage Class Configuration window giving complete details on that storage class.

- A "text" field is any field which displays alphanumeric text or number data. (This does not include "static" text painted on the window background, or labels on things like buttons.) Text fields may be single or multiple line, and they may be "enterable" (you can alter the displayed data) or "non-enterable" (you cannot change the displayed data directly).

- Most non-enterable text fields have gray backgrounds slightly lighter than the window background, and no borders. Some multi-line fields have the same background color, but use borders to help set them off from their surroundings. Some special fields display a fixed set of text strings, and use different background colors for different strings. These are mostly used to mark status conditions with different colors. For example, the status fields on the Health and Status window display text strings like "**Normal**", "**Warning**", "**Critical**", and so on, with green, yellow, and red background colors to match the severity of the condition.

- Enterable text fields have lighter backgrounds than non-enterable fields, and are framed within borders. To change the contents of a field, move the cursor into the field (with the mouse or the **Tab** key) and type in the desired value. As soon as you type the first character, the field colors change to white text on a blue background. This indicates that the field is being modified. When you are finished typing, press Tab to move to the next field, or Enter. This changes the field back to its normal colors.

SSM does not record what you have entered until you press the Tab or Enter key. If you fill in a configuration window, for example, and click the "Add" button while some fields are still displaying the white-on-blue colors, those fields will not be set correctly in the new configuration record.

Some enterable text fields are wider than they appear. If you type up to the right edge of such a field, the text automatically scrolls to the left and allows you to keep typing up to the actual size of the field. You can scroll back and forth within the field using the left and right cursor keys. "Scroll indicators" (graphics which look like something between parentheses and square brackets) appear on either end of the field if there is any data beyond the visible edge of the field.

Enterable fields may be "desensitized". On configuration windows, for example, some fields are enterable when creating a new record, but may not be changed when updating an existing record. A desensitized field has a grayed-out look, and you cannot type anything in it.

Your SSM security level (see Section 11.1: *SSM Security* on page 275 of the *HPSS Management Guide*) may not be high enough to allow you to change what looks like an enterable field. In this case, the field will not be grayed out, but you will be unable to type anything in it.

- Special pairs of text fields are used for entering "byte count" data. These consist of an enterable field on the left, a non-enterable field on the right, and a ">" character between them to mark the relation between the two. In the enterable field you can enter a number of bytes, or you can enter a number followed by "KB", "MB", "GB", or "TB" to specify kilobytes, megabytes, gigabytes, or terabytes, respectively. (Any of these may be abbreviated to the first letter, and case is ignored.) No matter how the value is entered, it is translated to bytes for display in the non-enterable part of the field.

- An "option list" field is a non-enterable text field surrounded by a goldenrod box. It looks like a button, but it has a small dash-shaped graphic (similar to a Motif option menu widget) on the right end. Clicking one of these fields pops up a list of options, also in goldenrod. You may select one of the displayed options (in which case it will replace the displayed contents of the field), or you can dismiss the list to cancel the change. Note that if there is only one possible option, the popup list will not appear; instead, the only choice will be automatically entered into the field.

The popup lists are of two types. Where practical, a popup "menu" is used, with all possible options displayed at once. This type can be dismissed by clicking the mouse anywhere outside of the popup. For potentially longer option lists, a “selection list” is used. This type uses scrollbars, if necessary, to display all the option data, and it has a “**Cancel**” button at the bottom of the list. You must click the “**Cancel**” button to dismiss this type of popup.

- Popup selection lists are used in other places besides being part of option lists. This type of popup is gray in color, but they work the same way as the option list variety.
- A “status line” field is a long non-enterable text field along the bottom of some windows. Status lines display messages which show important information about the state of the window, the progress of some operation started from the window, etc. In most cases, a status line message will be erased as soon as you do something to affect the window (click a button, type something into a field, etc.).
- Each window always has one item which has “input focus”. This item is surrounded by a red highlight border. If an enterable text field has input focus, typing anything on your keyboard will enter characters into the field. If a button has input focus, pressing the space bar will have the same effect as clicking the button with the mouse.
- You can do Motif select/cut/paste operations on enterable text fields, but not on non-enterable fields.

F.2 SSM On-line Help

Each SSM window has a help file which describes the features and fields on that window. To access the help for a window, position the mouse cursor to any blank spot on the window, hold down the **Shift** key, and click the right mouse button. A help browser window will open. When finished, click the “**DONE**” button at the bottom of the help browser to dismiss it.

Note that the mouse cursor must be in a blank spot when the button is clicked. If the cursor is over a button, field, or some other dynamic feature, an empty help browser will appear. This is because Sammi thinks you are requesting help on the field, not on the window, and SSM does not include any field-level help. The one exception is the main menu on the Health and Status window. Shift-clicking the right mouse button within this menu bar will display help for the main menu.

The on-line help provides fairly detailed information on all fields, buttons, and other features of each window. It does not, however, provide the detailed planning information provided in this Administration Guide. For example, the on-line help for the Storage Class Configuration window has a description of each field on the window, but it does not include the overall planning advice on how to design storage classes for the most efficient use of resources at your site.

F.3 Viewing SSM Session Information

The **Show Sammi Environment** menu option, under the **Session** heading on the SSM main menu, opens a window which shows the settings of all Sammi environment variables for the current SSM session. This information can be useful in diagnosing Sammi problems.

The **View Sammi Errorlog** menu option opens a window which displays the Sammi error log file for the current SSM session. The error log display is difficult to interpret and usually not very informative, but it sometimes can be useful in diagnosing SSM or Sammi problems.

The **About Sammi** menu option opens a window which displays information about Sammi and about Kinesix, the Sammi developer. Among other items, it shows the current version of the Sammi runtime, the host operating system and operating system version, and the hostname where Sammi is running. Again, this information can sometimes be useful in diagnosing problems.

The **SSM Consoles** menu option, under the **Monitor** heading on the SSM main menu, opens the SSM Console Information window. A "console" refers to one Sammi session which has connected to SSM. When first opened, the window shows information on the local (i.e., your) console. If other consoles are connected to the same SSM Data Server, you can use the "<<" and ">>" buttons to view information on other consoles.

Once a console has connected to an SSM Data Server, the Data Server continues to maintain information on that console in its internal console list, even after its SSM session has completely shut itself down. Seeing an inactive console in the list which is marked "Down" and "Logged Off" is perfectly normal. Console information is re-initialized only when the Data Server is shut down and restarted.

The window shows information such as console ID, connection state (Up or Down), last uptime and downtime, logon state, last logged-on user, and so forth (see the window help file for details). Sometimes, a Sammi session will crash, which can generate a lot of SSM error messages and cause problems for the remaining consoles. In such a situation, this window can be useful in tracking down which console may be causing the problem. Many times such console problems can be cleared up by restarting the crashed SSM session.

F.4 Customizing SSM and Sammi

This section lists a mixed bag of activities which can be performed to provide some site customization of SSM and Sammi. Some of these activities may be of interest to individual users; others only to the HPSS administrator.

- The file **<Sammi installation directory>/data/timzones.dat** defines time zones and the start/end dates for daylight savings time in a particular year. It should be edited by the HPSS administrator at the beginning of each calendar year. It should also be edited immediately after HPSS is installed, to make sure that the installed file is up to date, and that it specifies the correct local time zone. The format of the file is explained by comments within the file.

- The **Print SSM Window** menu option, under the **Session** heading on the SSM main menu, allows you to print a hard copy of an SSM window on a PostScript printer. Sammi uses the environment variable **SAM2_SPOOL** as the command which sends the hard copy to the printer. The **/opt/hpss/bin/start_ssm_session** script defines **SAM2_SPOOL** as "**lpr**".

  With the "**lpr**" command, output will ordinarily go to the default printer on the host where Sammi is executing. Users can direct output to a different printer by defining the **PRINTER** environment variable before running **start_ssm_session**.

  The HPSS administrator may want to edit **start_ssm_session** to change the print command defined by **SAM2_SPOOL**, or to add options to the "**lpr**" command.

- The file /opt/hpss**/sammi/hpss_ssm/hpss.def** contains Sammi "defaults" settings for SSM. Sammi automatically loads **<Sammi installation directory>/data/s2_defaults.def** when it starts, and **hpss.def** is used to override the settings in **s2_defaults.def**. Some of the user-preference features which can be set in a defaults file are:

1. Whether or not popup items on windows cause a beep tone.
2. The volume of the beep which is issued when the user tries to type beyond the end of a field.
3. The text entry foreground and background colors.
4. The blink rate for blinking fields.

Again, consult the comments for more information. For a complete list of default options, consult s2_defaults.def, or see the documentation on the set-default command in Chapter 4 of the Sammi Version 4.0 Runtime Reference.

The HPSS administrator may, if desired, modify **hpss.def** to change the site defaults. Alternatively, new defaults files may be created for individual users or groups of users. This would also involve the user authorization file and login command files (see below). The names of new defaults files must end in "**.def**", and must reside in **/opt/hpss/sammi/hpss_ssm** directory.

Any changes to defaults beyond the most innocuous user-preference items should be done with great care.

- Each user that logs into SSM has a Sammi command file executed automatically. The command file for each user is specified in **/opt/hpss/sammi/hpss_ssm/user_authorization.dat**, as the fifth field in each user entry. The default is **hpss.cmds**, which is in the **<Sammi installation directory>/bin** directory. All command files are expected to be in that directory, but they can be placed elsewhere by specifying the full path name of the file in **user_authorization.dat**.

The default command file loads the SSM "defaults" file (described above). If desired, the HPSS administrator can modify **hpss.cmds**, or create new command files for individual users or groups of users. A new command file might be created, for example, to load an alternate defaults file for a special group of users. Or a "**redefine-keys**" command might be added to load an alternate keyboard definition file for a particular user (see below).

If a new command file is created, **user_authorization.dat** will need to be edited to assign the file to selected SSM users.

Sammi command files must contain nothing but valid Sammi commands, as documented in the Sammi Version 4.0 Runtime Reference.

- The file **SAMMI** contains X defaults data important to the correct execution of Sammi. SSM users need a copy of this file in their home directories on all hosts where they will run Sammi (which may not be the same as the hosts where they display the windows). Normally, the **hpssuser** utility copies a **SAMMI** file to the home directory of each new SSM user which it creates, but that applies only to the host where **hpssuser** is executed. The template copy of **SAMMI** is in **/opt/hpss/config/templates/SAMMI.template.**

The **SAMMI** file is customizable to some extent, and each SSM user is free to modify his own copy. These modifications should be kept to a minimum, however, or there may be severely adverse effects on the appearance of the SSM windows.

- The file **.motifbind** is read by the Motif window manager when it starts up, and is used to set key bindings to make Sammi work correctly. There are six versions of this file in **/usr/ lpp//sammi/data**:

  .motifbind.dec

  .motifbind.hp

  .motifbind.ibm

  .motifbind.sgi

  .motifbind.sun_mit

  .motifbind.sun_news

In theory, all SSM users need **.motifbind** files in their home directories on the hosts where they work with SSM and view the windows (which may not be the same as the hosts where they are running Sammi). The Sammi documentation states that the proper file for the platform should be copied to each home directory and renamed to "**.motifbind**".

In practice, on AIX systems, the file is not required because the settings in **.motifbind.ibm** are the same as the defaults built into Motif. On other platforms, you may want to experiment to see if using one of these files solves any keyboard problems you may be having.

- The file **<Sammi installation directory>/data/keydefs.dat** also affects Sammi keyboard behavior. This file is loaded into Sammi whenever it starts. By default, it contains keyboard data appropriate to the platform where Sammi is installed.

A user who views SSM windows on a different platform may need to load a different **keydefs.dat** file. The alternatives in **<Sammi installation directory>/data** are:

  keydefs.dat.ALPHA

  keydefs.dat.HP

  keydefs.dat.NIGHT

  keydefs.dat.RS6000

  keydefs.dat.SCO

  keydefs.dat.SGI

  keydefs.dat.SPARC_SOLARIS

  keydefs.dat.SUN

Each user can load a **keydefs.dat** file by using the **Set Keyboard** option under the **Session** heading on the SSM main menu. This change lasts as long as the current SSM session. Alternatively, the HPSS administrator can permanently change the **keydefs.dat** file for a user by creating a Sammi login command file which includes a redefine-keys command specifying the appropriate keyboard file. Once the login command file is created, the administrator will have to edit the Sammi user authorization file to assign the login command file to the users who need it (see above).

- Each user is allowed to set "preferences" to customize selected windows in SSM (see Chapter 1: *Managing HPSS Servers* (page 19) in the *HPSS Management Guide*). These preferences may be saved in disk files which SSM can automatically load each time the user logs into an SSM session. Preferences are saved in disk files in each user's SSM work area, **/opt/hpss/sammi/ssm_user/<user>**, where **<user>** is an actual SSM username.

While each SSM work area is owned by an SSM user, the SSM Data Server process is the entity which actually writes and reads the preferences files. This means that in most cases, the Data Server will be performing file operations in a directory which does not belong to it. This introduces permissions problems which can interfere with the user's ability to save and reload SSM preferences. The best way to avoid this problem is to have all SSM users in the same Unix group as the SSM processes, and to set group "rwx" permissions on the SSM work area directories.

F.5 Detailed Information on Setting Up an SSM User

The HPSS User Management Utility, **hpssuser**, must be run as **root** to set up each SSM user. Use the **-aix**, **-dce**, and **-ssm** flags to add a new user as an AIX user, a DCE principal, and an SSM user, respectively. Use whichever of these flags are appropriate for each new SSM user, but do not use a flag that is not needed. Do not, for example, use **-aix** if the user already has an AIX account on the SSM host. Use the **-h** flag to view a short summary of how to use **hpssuser**. Refer to the manual page for the **hpssuser** utility in Section 12.2: *HPSS Utility Manual Pages* on page 293 of the *HPSS Management Guide* for more usage information.

Operations performed by **hpssuser** are summarized below:

1. 1.If the **-dce** flag is specified, **hpssuser** creates a DCE principal entry and a DCE account entry for the new user in the DCE security registry. You will need to know the **cell_admin** password for your DCE cell in order to do this.

2. If the **-aix** flag is specified, **hpssuser** adds the new user as an AIX user.

3. If the **-ssm** flag is specified, you will be prompted for the hostname on which SSM will run and the user security level. The **hpssuser** utility will setup the SSM user as follows:

   - ◆ The Sammi API configuration file (**/opt/hpss/bin/api_config.dat**) is searched for existing "console ID" numbers. Each Sammi user requires a unique console ID number. After searching through the API configuration file, **hpssuser** creates a new console ID that is not yet in use.

   - ◆ Sun ONC RPC addresses are created for the two vital Sammi processes (**s2_evtsvr** and **s2_stream**) by using the new console ID as part of each address. This results in a unique set of RPC addresses for each console ID.

   - ◆ Three new lines are added to the API configuration file. The first line is a comment line that includes the user's name. The last two lines begin with **logical_server**, and define the RPC address, RPC version number (equal to the console ID), and hostname (the one you entered at the prompt earlier) for each of the two vital Sammi processes. This information is read by the SSM Data Server process at start-up, and is used by the Data Server to contact the Sammi processes being run by a specific SSM user.

   - ◆ A new entry for the user is added to the Sammi User Authorization file **/opt/hpss/sammi/hpss_ssm/user_authorization.dat**.

- SSM work areas for the new user are created. These work areas are **/opt/hpss/sammi/ssmuser/<user>** (where **<user>** is the actual ID of the new user), and **/opt/hpss/sammi/ssmuser/<user>/mapfiles**.
- A template Sammi configuration file (**ssm_console.dat**) is copied to the new user's work area and modified to be user-specific. This involves editing the console ID number, process hostnames, and the RPC addresses and version numbers for the two vital Sammi processes.
- An X resources file named **SAMMI** is copied to the user's home directory.

F.6 Non-standard SSM Configurations

A number of SSM configurations are possible that do not get set up by the default HPSS installation or SSM user configuration. Using these configurations involves performing some procedures manually, instead of accepting the automated defaults. The information below can help in setting up these non-standard configurations.

You should not attempt to set up non-standard configurations unless you are familiar with how SSM works, and only when there is a real need to do so. These configurations are more difficult to set up and maintain, and most of them will have negative impacts on SSM performance.

Multiple Data Servers. The defaults assume only one SSM Data Server, running on the same host as the other SSM components. You may want multiple Data Servers, with some or all of them running on different hosts. For each SSM user with a non-default Data Server, you must edit the user's Sammi configuration file (**/opt/hpss/sammi/ssmuser/<user>/ssm_console.dat**). Near the bottom of this file is a **logical_server** line for **ssm_ds**, which is the Data Server. Edit the RPC address and the hostname on this line as appropriate for the non-default Data Server. Note that all concurrently running Data Servers must have unique RPC addresses (although it is not mandatory that they run on different hosts). Make sure that any RPC addresses you select are not being used by any other processes on that host. Also note that multiple Data Servers must have unique CDS entries, as is true for any HPSS server.

If any of the above changes are made, the user must create a customized version of the **start_ssm** script which reflects the non-default RPC address for the Data Server and the non-default CDS names. If the Data Server and System Manager are to be executed on different hosts, invoke the **start_ssm** script with the **-s** option (to start the System Manager only) or **-d** option (to start up the Data Server only) on the appropriate host. If more than one Data Server is to be run on the same host, the logic in the script that disallows duplicate data servers must be removed.

Non-standard Sammi Runtime. The defaults assume that all SSM users are running Sammi on the same host where you executed the **hpssuser** script. If your site has multiple Sammi licenses, an SSM user may be running Sammi on a workstation remote from the main SSM processes. For any Sammi console ID not on the default host, you must edit the API configuration file (for that console) to correct the hostname for the two Sammi processes. You must also edit the user's Sammi configuration file (**ssm_console.dat**) to change the same two hostnames. You must then transfer the user's Sammi work area (see above) from the default host to the actual host. Finally, you will likely have to provide the user a customized copy of the **start_ssm_session** script to deal with the non-standard Sammi runtime.

Multiple SSM Sessions. The defaults assume that each user will run only one SSM session at a time. If one user must run multiple SSM sessions, the easiest way to configure this is to create multiple user names for that user with **hpssuser**. If you choose not to do this, you must create completely separate Sammi execution environments for each concurrent session that a user may want to run. This involves separate Sammi work areas, hand-customized Sammi configuration files and API configuration file, and customized **start_ssm_session** scripts. Each concurrent session will have to use a different Sammi console ID and different sets of Sammi process RPC addresses.

Appendix G

High Availability

G.1 Overview

The High Availability (HA) feature of HPSS allows a properly configured HPSS system to automatically recover from a number of possible failures, with the goal of eliminating all single points of failure in the system. The same functionality can be used to minimize the impact of regularly scheduled maintenance and / or software upgrades.

The High Availability feature is only available on AIX platforms.

Failures of the following components will be protected against when using a properly configured HA HPSS system:

- Core server
- Network-related
  - adapters
  - cables
- Disk-related
  - adapters
  - cables
  - disks
- Power-related
  - node power supply
  - disk power supply
  - power distribution strip

High availability is not the same as fault tolerance. The failures above are “protected against” from the standpoint that the HA HPSS system will be able to return to an operational state without intervention when any **one** of the above failures occur. There certainly may be some down-time, especially when the core server fails (crashes).

After a recovery, HPSS will function properly, but it will no longer be in a Highly Available state. A subsequent failure may not be recoverable. For instance, if the core server crashes and the backup takes over, there is no longer a backup node. It will be necessary to correct the original failure in order to return the system to a Highly Available state.

G.1.1 Architecture

The following diagram shows the necessary components for an HA HPSS configuration:

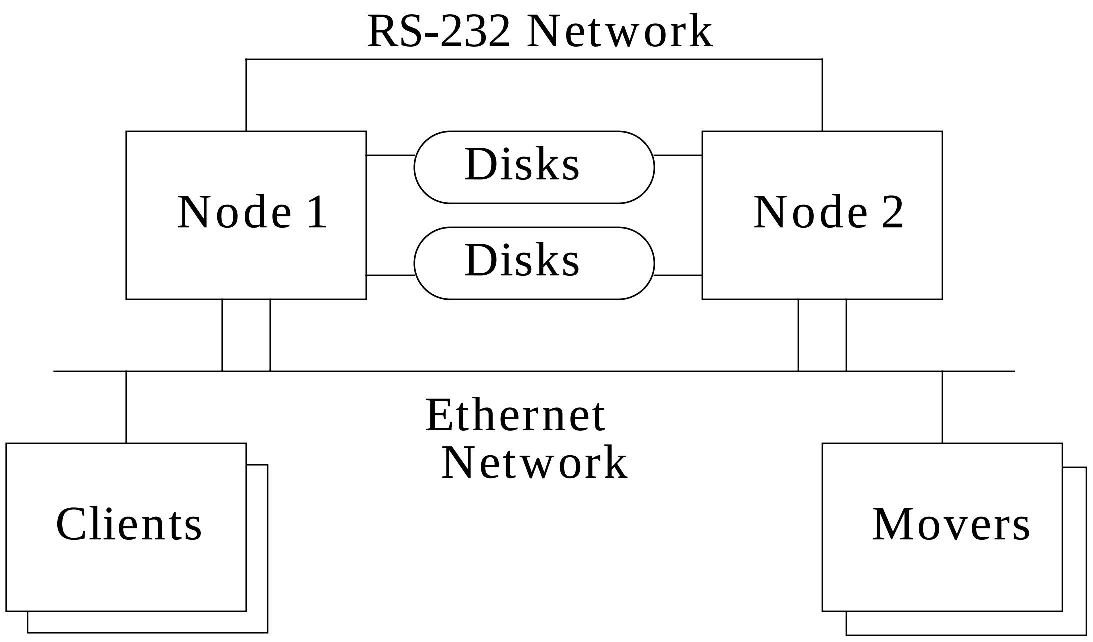


Figure H-1 HA HPSS Architecture

This diagram does not include the power system, but it does have several features that are very important:

- At any point in time, either Node 1 or Node 2 can act as the core HPSS server.
- The two shared disk busses are mirrored to one another and accessed by each node using separate adapter cards so that any single failure (disk, adapter, or bus) will result in accessibility of at least one good copy of the data.

- Each node has two connections to the ethernet network. One is a “standby” that can take over the IP and hardware addresses of the primary adapter in case of failure.
- There is an RS-232 serial cable connecting Node 1 and Node 2 to enable communication even in the event that the main network fails.

G.2 Planning

G.2.1 Before You Begin

Ensure that appropriate prerequisite software is installed. See section Section 2.3.2.1: *HPSS Server/ Mover Machine - AIX* on page 50 for specific prerequisite information.

HACMP for AIX is complex software and is the core component of a Highly Available HPSS system. As such, the planning, installation, and configuration of an HA HPSS system should be done by personnel with the appropriate HACMP training and should not be attempted by HPSS administrators or operators with little HACMP background.

It is important that administrators understand the concepts presented in the *HACMP for AIX Concepts and Facilities* guide. After reviewing that guide, administrators should have enough background to begin the planning process.

G.2.2 HACMP Planning Guide

Appendix A of the *HACMP for AIX Planning Guide* has all the information necessary to walk an administrator through configuration of a basic system, including a number of planning worksheets. For an HA HPSS configuration the following worksheets would normally be used:

- TCP/IP Networks Worksheet
- TCP/IP Network Adapter Worksheet
- Serial Networks Worksheet
- Serial Network Adapter Worksheet
- Shared SCSI-2 Differential or Differential Fast/Wide Disks Worksheet (or one of the other shared disk worksheets that matches your shared disk hardware)
- Non-Shared Volume Group Worksheet (Non-Concurrent Access)
- Shared Volume Group/Filesystem Worksheet (Non-Concurrent Access)
- Application Worksheet
- Application Server Worksheet
- Resource Group Worksheet

- Cluster Event Worksheet

However, this is a large list of worksheets to go through, so they have been condensed down to cover only what is needed for an HA HPSS system. The following pages contain the condensed HA HPSS Planning Worksheet with suggested values. Please refer to the *HACMP for AIX Planning Guide, Appendix A* for help in filling in the blanks below. A sample completed worksheet is given in Section G.2.2.2: *HA HPSS Planning Worksheet (example)* on page 541.

The name chosen for the service adapter must be the same as the host name given to each node. For instance, the example worksheet lists **hasvc** as the service adapter's name. In that case, the host name (as given by the **hostname** command) for each node also needs to be **hasvc**. This is necessary for DCE and HPSS to properly function during a failover.

G.2.2.1 HA HPSS Planning Worksheet

Cluster ID	**1**
Cluster Name	**HAHPSS**

Network:

Name	**ether1**	
Type	**Ethernet**	
Attr	**public**	
Netmask	__________	
Node names	**hanode1**,	**hanode2**

Boot and Standby Adapters (hanode1):

IP Label	__________	__________
Function	**boot**	**standby**
IP Address	__________	__________
Network Name	**ether1**	**ether1**
Network Attr	__________	__________

Boot and Standby Adapters (hanode2):

IP Label	__________	__________
Function	**boot**	**standby**
IP Address	__________	__________

Network Name **_ether1___** **_ether1___**

Network Attr __________ __________

Service Adapters:

IP Label __________

Function **_service___**

IP Address __________

Network Name **_ether1___**

Network Attr __________

HW Address __________

Serial Networks:

Network Name **_serial1___**

Network Type **_RS232____**

Node Names **_hanode1__**, **_hanode2__**

Serial Network Adapter Worksheet (node A):

Slot Number __________

Interface Name __________

Adapter Label __________

Network Name **_serial1___**

Serial Network Adapter Worksheet (node B):

Slot Number __________

Interface Name __________

Adapter Label __________

Network Name **_serial1___**

Shared SCSI-2 Differential or Differential Fast / Wide Disks Worksheet (bus1):

Type of Bus __________

Node Name **_hanode1_**, **_hanode2_**

Slot Number __________, __________

Adapter Logical Name	__________, __________
Adapter Bus ID	**_6________, _5________**
Shared Disk Bus IDs	__________

Shared SCSI-2 Differential or Differential Fast/Wide Disks Worksheet (bus2):

Type of Bus	__________
Node Name	**_hanode1_, _hanode2_**
Slot Number	__________, __________
Adapter Logical Name	__________, __________
Adapter Bus ID	**_6________, _5________**
Shared Disk Bus IDs	__________

Non-Shared Volume Group Worksheet (Non-Concurrent Access) (hanode1):

Volume Group Name	**_rootvg___**
Physical Volumes	**_hdisk0___, _hdisk1___**
Logical Volumes	__________, __________, __________, __________
Mirrored?	__________, __________, __________, __________

Non-Shared Volume Group Worksheet (Non-Concurrent Access) (hanode2):

Volume Group Name	**_rootvg___**
Physical Volumes	**_hdisk0___, _hdisk1___**
Logical Volumes	__________, __________, __________, __________
Mirrored?	__________, __________, __________, __________

Shared Volume Groups/Filesystems:

Volume Group Name	__________
Major Number	__________
Log LV Name (if any)	__________
Physical Volumes	__________, __________, __________, __________, __________,
	__________, __________, __________, __________, __________,
	__________, __________, __________, __________, __________

Volume Group Name	________
Major Number	________
Log LV Name (if any)	________
Physical Volumes	________, ________, ________, ________, ________,
	________, ________, ________, ________, ________,
	________, ________, ________, ________, ________
Volume Group Name	________
Major Number	________
Log LV Name (if any)	________
Physical Volumes	________, ________, ________, ________, ________,
	________, ________, ________, ________, ________,
	________, ________, ________, ________, ________
Volume Group Name	________
Major Number	________
Log LV Name (if any)	________
Physical Volumes	________, ________, ________, ________, ________,
	________, ________, ________, ________, ________,
	________, ________, ________, ________, ________

HPSS Resource Group:

Resource Group Name	**HPSSResourceGroup**
Resource Group Type	**Rotating**
Application Server Name	**HPSSAppServer**
Start Command	**/usr/sbin/cluster/events/hpss/hpss_start.ksh**
Stop Command	**/usr/sbin/cluster/events/hpss/hpss_stop.ksh**

G.2.2.2 HA HPSS Planning Worksheet (example)

Cluster ID **1**

Cluster Name **HAHPSS**

Network:

Name	**ether1**
Type	**Ethernet**
Attr	**public**
Netmask	**255.255.255.0**
Node names	**hanode1** , **hanode2**

Boot and Standby Adapters (hanode1):

IP Label	**ha1boot**	**ha1stby**
Function	**boot**	**standby**
IP Address	**192.94.47.244**	**192.94.48.145**
Network Name	**ether1**	**ether1**
Network Attr	**public**	**public**

Boot and Standby Adapters (hanode2):

IP Label	**ha2boot**	**ha2stby**
Function	**boot**	**standby**
IP Address	**192.94.47.69**	**192.94.48.146**
Network Name	**ether1**	**ether1**
Network Attr	**public**	**public**

Service Adapters:

IP Label	**hasvc**
Function	**service**
IP Address	**192.94.47.156**
Network Name	**ether1**
Network Attr	**public**
HW Address	**0x10deadbeef01**

Serial Networks:

Network Name	**serial1**
Network Type	**RS232**
Node Names	**hanode1**, **hanode2**

Serial Network Adapter Worksheet (node A):

Slot Number	**sa2**
Interface Name	**/dev/tty1**
Adapter Label	**ha1tty**
Network Name	**serial1**

Serial Network Adapter Worksheet (node B):

Slot Number	**sa2**
Interface Name	**/dev/tty1**
Adapter Label	**ha2tty**
Network Name	**serial1**

Shared SCSI-2 Differential or Differential Fast / Wide Disks Worksheet (bus1):

Type of Bus	**Ultra SCSI**
Node Name	**hanode1**, **hanode2**
Slot Number	**5**, **5**
Adapter Logical Name	**scsi3**, **scsi3**
Adapter Bus ID	**6**, **5**
Shared Disk Bus IDs	**0-4,8-11**, **0-4,8-11**

Shared SCSI-2 Differential or Differential Fast / Wide Disks Worksheet (bus2):

Type of Bus	**Ultra SCSI**
Node Name	**hanode1**, **hanode2**
Slot Number	**6**, **6**
Adapter Logical Name	**scsi5**, **scsi5**
Adapter Bus ID	**6**, **5**
Shared Disk Bus IDs	**0-4,8-11**, **0-4,8-11**

Non-Shared Volume Group Worksheet (Non-Concurrent Access) (hanode1):

Volume Group Name	**rootvg**
Physical Volumes	**hdisk0**, **hdisk1**
Logical Volumes	**hd1-6, hd8, hd9var**
Mirrored?	**all are mirrored**

Non-Shared Volume Group Worksheet (Non-Concurrent Access) (hanode2):

Volume Group Name	**rootvg**
Physical Volumes	**hdisk0**, **hdisk1**
Logical Volumes	**hd1-6, hd8, hd9var**
Mirrored?	**all are mirrored**

Shared Volume Groups/Filesystems:

Volume Group Name	**dcevg**
Major Number	**51**
Log LV Name (if any)	**jfslogdce**
Physical Volumes	**hdisk2**, **hdisk11**

Volume Group Name	**sfsvg**
Major Number	**52**
Log LV Name (if any)	**jfslogsfs**
Physical Volumes	**hdisk3**, **hdisk4**, **hdisk5**, **hdisk6**, **hdisk8**, **hdisk12**, **hdisk13**, **hdisk14**, **hdisk15**, **hdisk17**

Volume Group Name	**hpssvg**
Major Number	**53**
Log LV Name (if any)	**jfsloghpss**
Physical Volumes	**hdisk10**, **hdisk19**

HPSS Resource Group:

Resource Group Name	**HPSSResourceGroup**
Resource Group Type	**Rotating**
Application Server Name	**HPSSAppServer**
Start Command	**/var/hahpss/hpss_start.ksh**
Stop Command	**/var/hahpss/hpss_stop.ksh**

G.3 System Preparation

G.3.1 Physically set up your system and install AIX

The first step to prepare a system for HA HPSS is to perform the physical setup. This includes all the physical cabling for power, networks, disk devices, etc. We will not go into detail on how to accomplish this as most of this procedure is routine. However, be sure to follow the guidelines below to ensure that the resulting system is capable of serving as a highly available system.

- Keep the shared disks on separate busses
- Cable each disk bus on each node to a separate SCSI adapter, and make sure that the disk drawer that attaches to the SCSI adapter in slot X of node 1 also attaches to the same port on the SCSI adapter in slot X of node 2. This will ensure that hdiskYY on node 1 is the same physical disk as hdiskYY on node 2.
- Route power cables such that the dual power supplies for each node and disk drawer are supplied from a different power distribution strip.
- Use only the ethernet adapters in the PCI slots, do not use the ethernet cards that are integrated into the motherboards. (This is because any ethernet adapter may fail, and replacing the integrated adapter would involve replacing the motherboard).
- Don't forget to connect your serial (null-modem) cable. Be sure to use it to connect like-named serial ports on both machines. For example, **S1-S1** or **S2-S2**.
- Be as symmetric as possible (make connections exactly the same way on both nodes).

G.3.2 Mirror rootvg

Mirror rootvg on each machine. Before this step, your rootvg should contain only one disk (we'll assume hdisk0), and you should have another internal disk available (we'll assume hdisk1). It is preferable to have each disk on a different internal bus:

```
% extendvg rootvg hdisk1
% mirrorvg -m rootvg hdisk1
% bosboot -a
% bootlist -m normal hdisk0 hdisk1
% shutdown -Fr
```

Note that the last step reboots the machine. This is necessary to disable the normal quorum checking for rootvg.

G.3.3 Diagram the Disk Layout

One key aspect to setting up your disks, volume groups, logical volumes, and file systems properly is knowing which disks to use. This is easiest if you draw a diagram to plan out your volume groups. To make sure your drawing is correct, it is important to know what bus the disks are on. Use the **lsdev -Cc disk** command, and you will get output similar to the following (for SCSI):

```
% lsdev -Cc disk
...
hdisk2   Available   20-58-00-0,0SCSI Disk Drive
hdisk3   Available   20-58-00-1,0SCSI Disk Drive
...
hdisk11  Available   20-60-00-0,0SCSI Disk Drive
hdisk12  Available   20-60-00-1,0SCSI Disk Drive
...
```

Use the **lsdev -Cc adapter** command, and you will get something like:

```
% lsdev -Cc adapter
...
scsi3    Available   20-58       SCSI Controller
scsi4    Available   20-59       SCSI Controller
scsi5    Available   20-60       SCSI Controller
scsi6    Available   20-61       SCSI Controller
...
```

By comparing the third column in both outputs, it can be determined that hdisk2 and hdisk3 have SCSI IDs 0 and 1 respectively on the bus attached to scsi3 and that hdisk11 and hdisk12 have SCSI IDs 0 and 1 respectively on the bus attached to scsi5. Therefore, you would most likely want to have hdisk2 mirror hdisk11 and hdisk3 mirror hdisk12.

G.3.4 Plan Shared File Systems

Before creating your diagram, take note that file systems supporting DCE, Encina SFS, HPSS, Sammi, and Java must reside on shared disks. Other site-specific file systems may also need to reside on shared disks in order to ensure that they are available to HPSS regardless of which node is active. The default list of shared file systems to support the above mentioned components are:

- **/krb5**
- **/var/dce**
- **/etc/dce**
- **/opt/encinalocal**

- **/opt/encinamirror**
- **/usr/lpp/encina**
- **/opt/hpss**
- **/var/hpss**
- **/usr/java130**
- **/usr/local/sammi**

Note that this lists default file system mount points only. Determine the appropriate file systems and mount points for your site before continuing.

The sizing of these file systems is very important. To determine sizing for HPSS-related file systems, see Section 2.10.3: *System Memory and Disk Space* on page 132. To determine sizing for DCE, Java, and Sammi file systems, refer to the appropriate documentation for those applications.

It is imperative that these filesystems are mirrored in such a way that each mirrored copy is stored on a different bus.

G.3.5 Mirror Shared JFS logs

AIX's JFS (Journaled File System) also requires a logical volume on which to store its log. Each volume group with at least one JFS will have its own JFS log logical volume. It is important that these be mirrored across busses as well since JFSs can't operate without their associated JFS logs.

To create your own JFS logs, instead of using the LVs that AIX creates automatically, do the following after creating the volume group but before doing anything else:

1. Create a Logical Volume of type **jfslog** with a size of one logical partition (1 LP), sequential update policy, and 2 copies of each LP.

2. Format the Logical Volume as a JFS log using the **logform** command:

```
% logform /dev/<LVname>
```

Now any JFS that is added to this volume group will use this JFS log.

G.3.6 Connect Robotic Tape Libraries

Since an HA HPSS system has two nodes which need to be capable of performing all functions of an HPSS core server, it is important that any connections to robotic libraries be identical on both nodes. For example, if **/dev/lmcp0** is a control point for a library on one node, then **/dev/lmcp0** also needs to be a control point to the same library on the other node. If your library performs security checking by IP address or hostname, use the IP address and interface name of the service adapter when adding your HA system to the library's security list.

G.4 Initial Install and Configuration for HACMP

G.4.1 Install HACMP

Install the following file sets from the HACMP 4.4 installation media on **each** node:

```
cluster.base
cluster.cspoc
cluster.doc.en_US
cluster.man.en_US
cluster.taskguides
cluster.vsm
```

G.4.2 Setup the AIX Environment for HACMP

1. Create the `/dev/tty<#>` devices on each node

2. Setup your SCSI adapters on each node

3. Create the `/.rhosts` file on each node

   This file must contain entries for the service and boot adapters for each node in the cluster plus one entry for the loopback interface (**127.0.0.1**). If you have concerns about using the **.rhosts** file, refer to page 1-8 of the *HACMP for AIX Administration Guide Version 4.4.*

   An example **/.rhosts** for our example configuration would be:

   ```
   hasvc
   ha1boot
   ha2boot
   ha1stby
   ha2stby
   127.0.0.1
   ```

4. Create the `/etc/hosts` file on each node

   This file must contain all node IP interfaces. Since HACMP cannot rely on DNS or NIS to be available, HACMP operations rely on properly configured **/etc/hosts** files. For our example configuration, the file would contain:

   ```
   127.0.0.1      loopback    localhost
   192.94.47.156  hasvc
   192.94.47.244  ha1boot
   192.94.47.69   ha2boot
   192.94.48.145  ha1stby
   192.94.48.146  ha2stby
   ```

5. Configure your boot adapters, standby adapters, and serial adapters

Note that you do not configure your service adapter. This is because HACMP will change your boot adapter into your service adapter when it brings up HPSS. Therefore, without HACMP running, you configure the adapter with its boot name and address.

6. Add Physical Volume IDs (PV IDs) for all shared hdisks (if necessary)

   If this is the first time that your shared disks have ever been used, you will need to manually assign PV IDs to them. To check for this, run **lspv**. The second column of the output should have a big 16-digit hexadecimal number, the PV ID, for each physical volume. If no PV ID is shown for certain volumes, you will have to create them yourself using the following command for each physical volume (the example assigned a PV ID to **hdisk3**):

   ```
   % chdev -l ‘hdisk3’ -a pv=’yes’
   ```

7. Increase the maximum number of allowed logins

8. Turn off NetWare if it is installed

   One potential problem is an incompatibility between NetWare and HACMP. NetWare will often keep a network interface busy which will cause attempted adapter swaps to fail. An easy test for this problem is to run the following command:

   ```
   % grep rcnetw /etc/inittab
   ```

   If the command came back with no output, you are fine. Otherwise, you will need to comment out the offending line in the **/etc/inittab**.

9. Create **/etc/environment** entry

   Since DCE’s RPC mechanism will attempt to use any and all configured interfaces, it is important to put a **RPC_UNSUPPORTED_NETADDRS** entry in the **/etc/environment** file on both nodes with the IP address of every non-service interface listed. This will force the RPCs to utilize the service interface. For example, our sample configuration from the example worksheet would require the following entry:

   ```
   RPC_UNSUPPORTED_NETADDRS=192.94.47.244:192.94.47.145:192.94.47
   .146:192.94.47.147
   ```

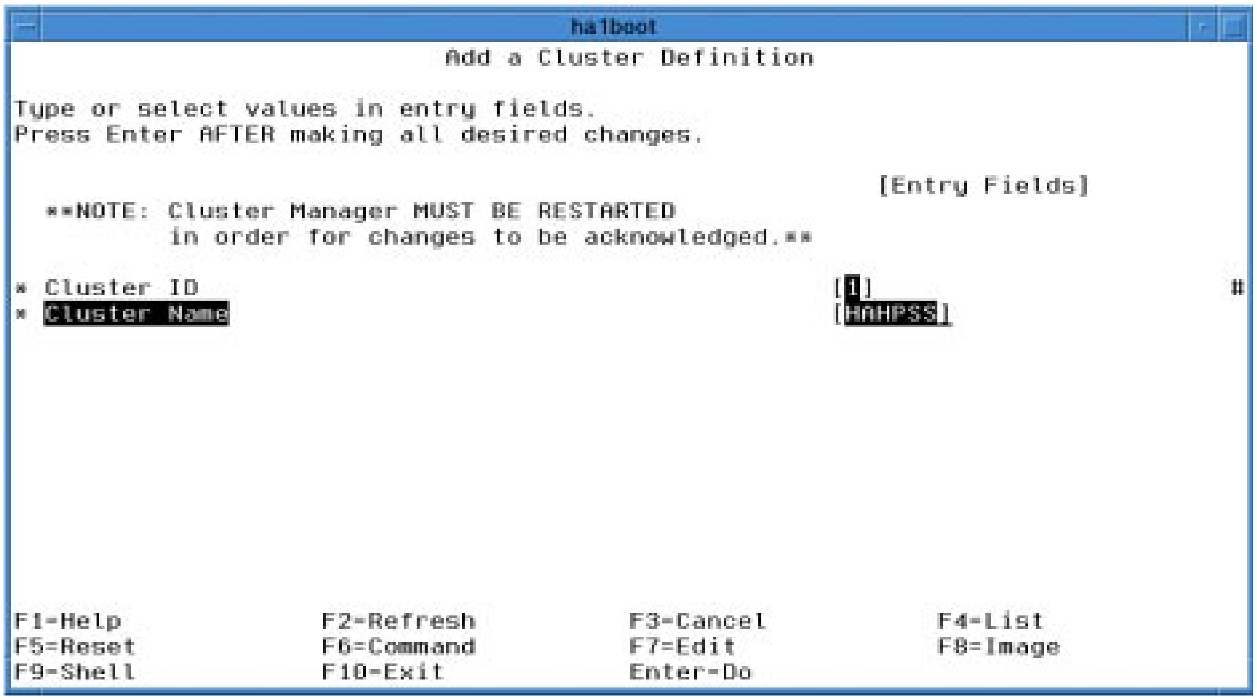
G.4.3 Initial HACMP Configuration

Using the information from your worksheets, perform the following steps from either nodes:

G.4.3.1 Define a new cluster

```
% smitty cluster

Cluster Configuration
   -> Cluster Topology
         -> Configure Cluster
```

```
                    -> Add a Cluster Definition
```



```
                          ha1boot
                   Add a Cluster Definition

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                      [Entry Fields]
  **NOTE: Cluster Manager MUST BE RESTARTED
          in order for changes to be acknowledged.**

* Cluster ID                                       [1]                  #
* Cluster Name                                     [HAHPSS]

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure H-2 Adding a Cluster Definition

G.4.3.2 *Define the two nodes in the cluster*

Now tell the HACMP the names of the nodes in the cluster. These names do not have to be the same as the hostnames or adapters of the nodes since the names are used internally to HACMP.

```
% smitty hacmp

Cluster Configuration
   -> Cluster Topology
         -> Configure Nodes
               -> Add Cluster Nodes
```

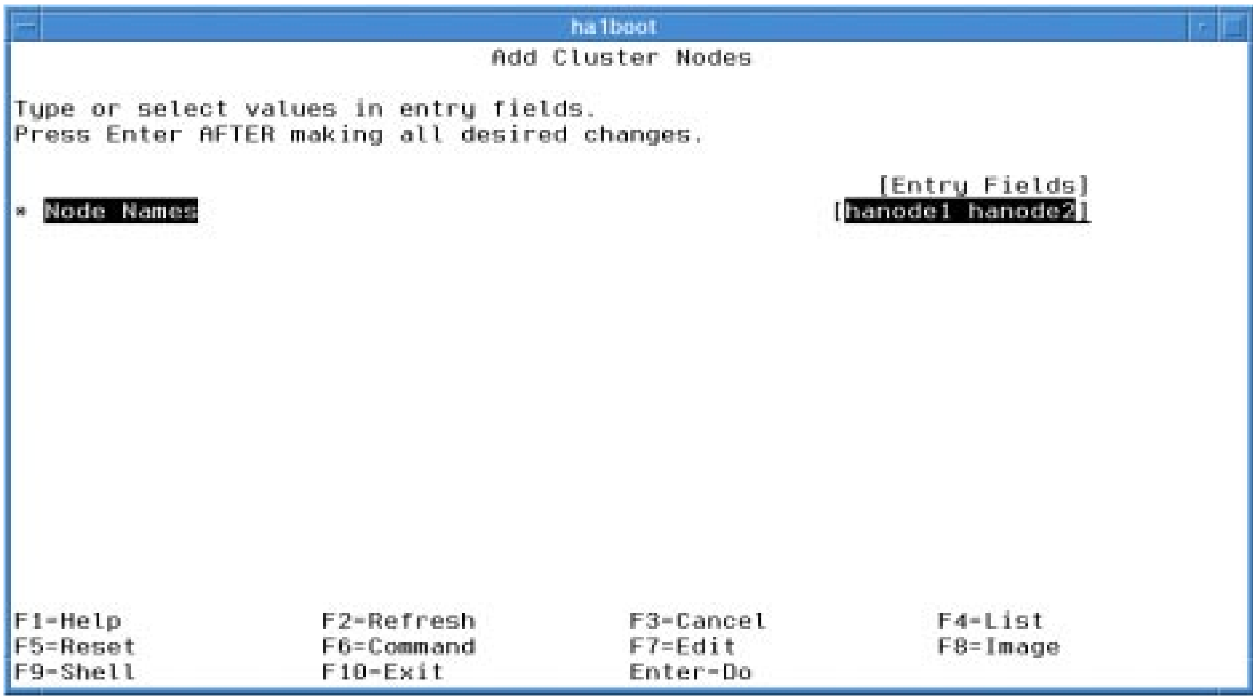

```
halboot
                              Add Cluster Nodes

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                         [Entry Fields]
* Node Names                                            [hanode1 hanode2]






F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure H-3 Adding Cluster Nodes

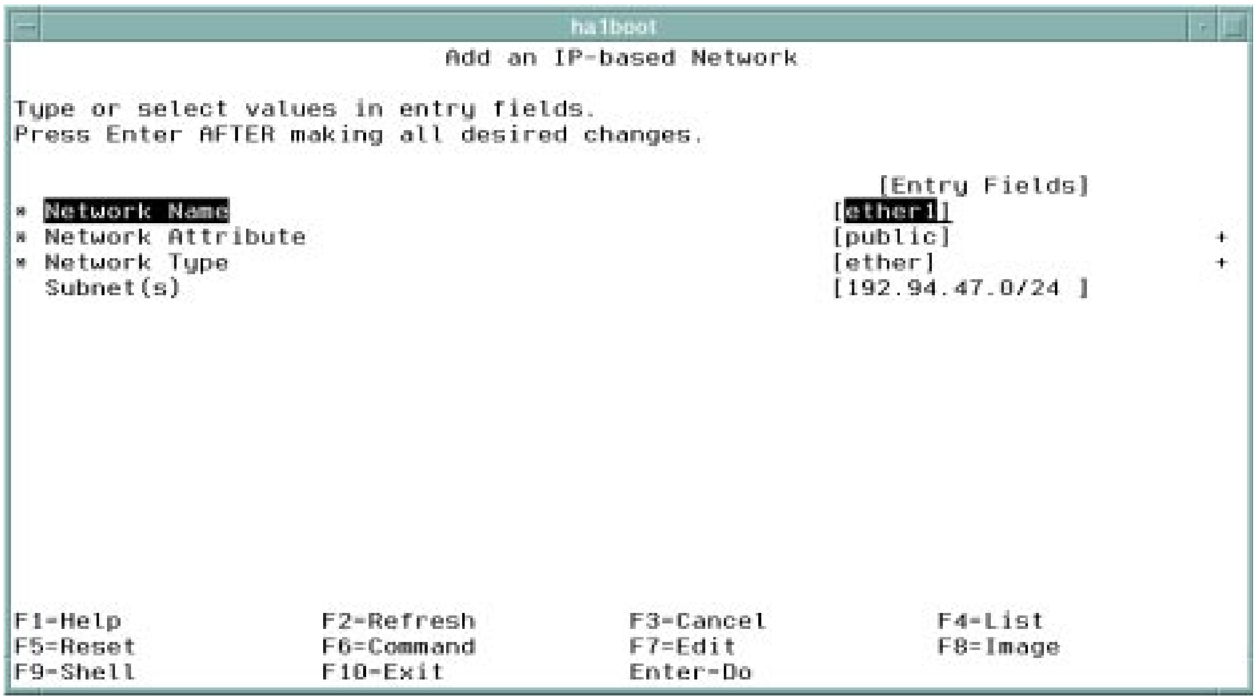
G.4.3.3 *Define the networks*

The SMIT path for configuring each network (ethernet and RS232) is the same

```
% smitty hacmp

Cluster Configuration
   -> Cluster Topology
         -> Configure Networds
               -> Add an Network
```

At this point, you will be offered two options, **`IP-based Network`** and **`Non IP-based Network`**.

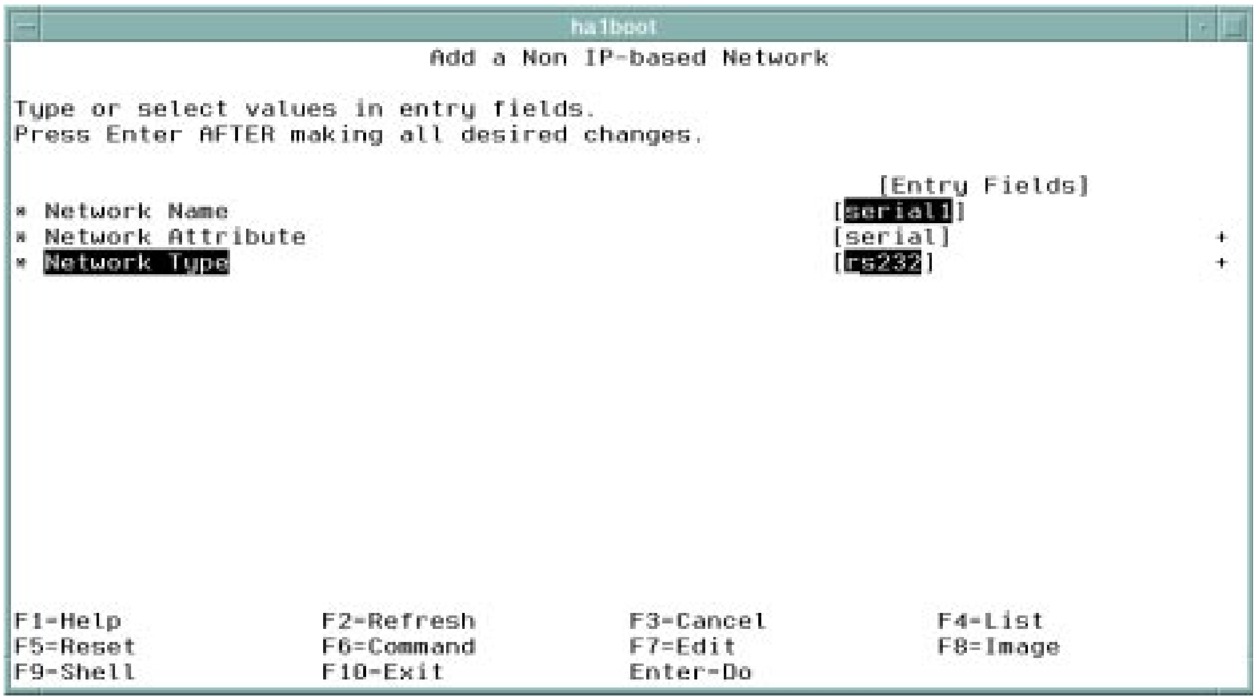
Choose **`IP-based Network`** to configure your ethernet network. When this choice is entered, SMITTY will query your machine for IP-based networks and provide you with a list of networks. There will likely be only one choice, but make sure that the network you choose corresponds to the IP addresses for your service, boot, and standby adapters.



```
                                  ha1boot
                          Add an IP-based Network

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                          [Entry Fields]
* Network Name                                         [ether1]
* Network Attribute                                    [public]              +
* Network Type                                         [ether]               +
  Subnet(s)                                            [192.94.47.0/24 ]

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure H-4 Adding an IP-based Network

Choose **`Non IP-based Network`** to configure your RS232 network:



```
                                  ha1boot
                        Add a Non IP-based Network

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                          [Entry Fields]
* Network Name                                         [serial1]
* Network Attribute                                    [serial]              +
* Network Type                                         [rs232]               +

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure H-5 Adding a Non IP-based Network

G.4.3.4 *Define the network adapters*

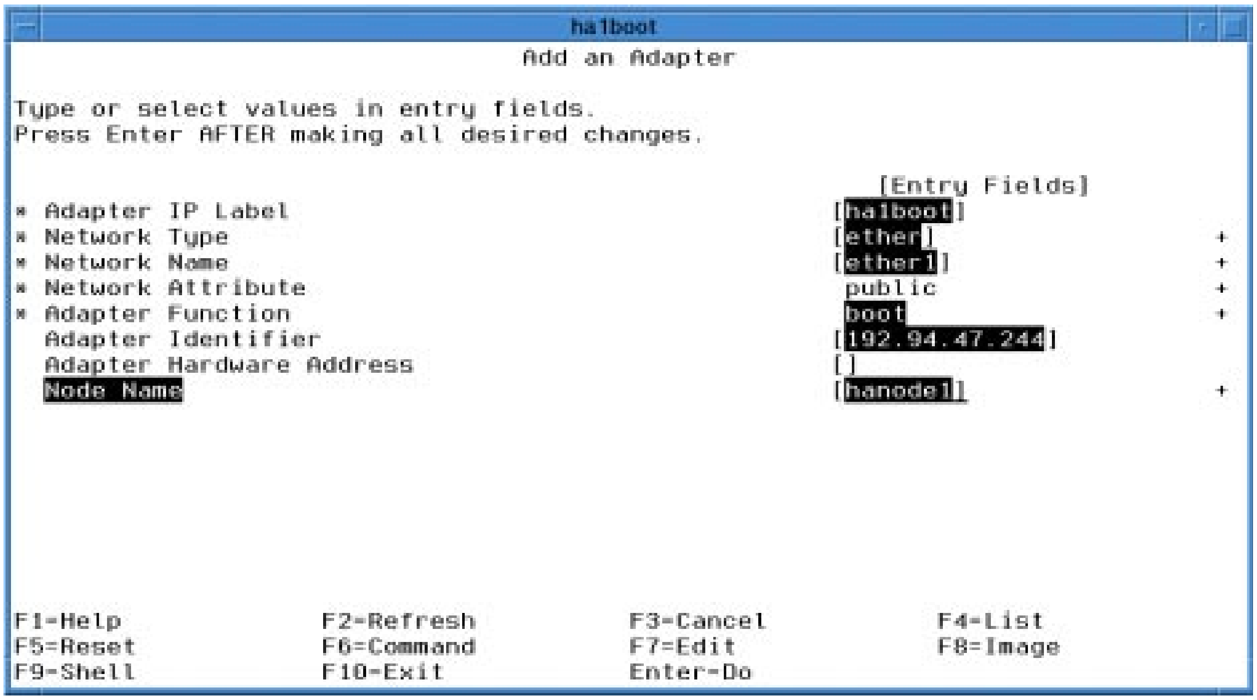
When defining network adapters, there are some slight differences between the values supplied for service, boot, standby, and serial adapters:

- Boot, standby, and serial adapters are tied to particular nodes, while service adapters are not.
- Service adapters have associated hardware (MAC) addresses so that clients don't have to flush their ARP caches when it moves from one physical adapter to another.
- Serial adapters have a field that says "service", but we don't call them service adapters.

The SMIT path for configuring each adapter is the same

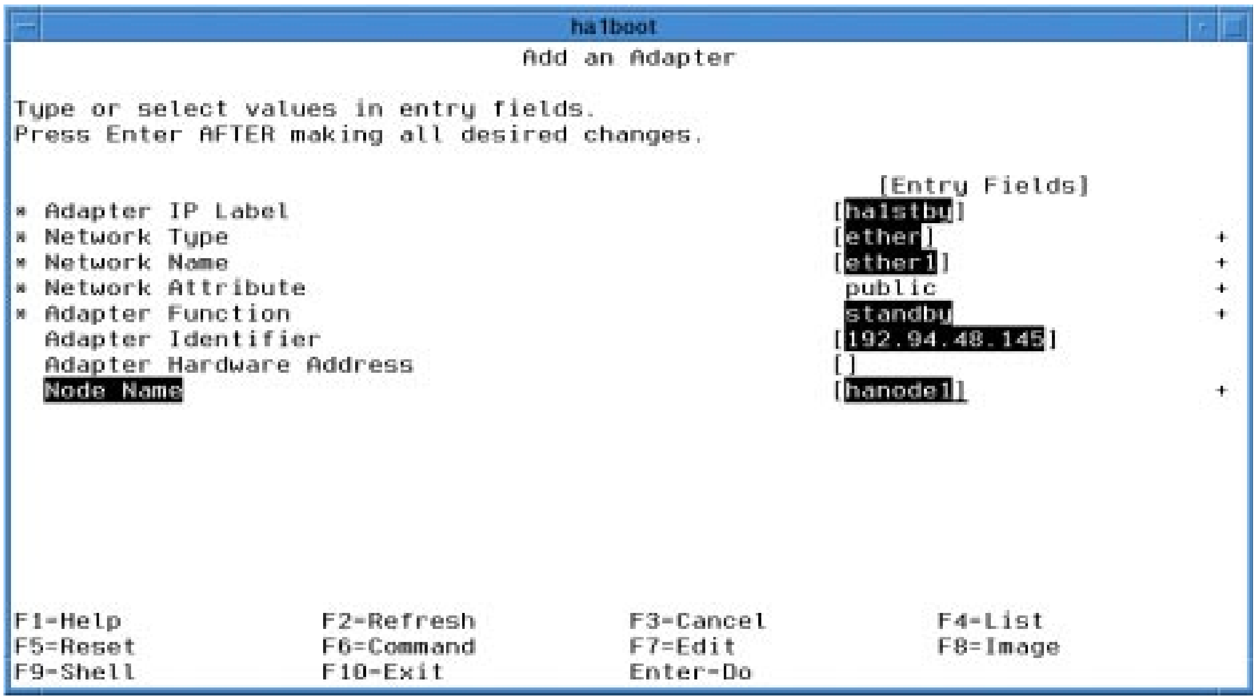
```
% smitty hacmp

Cluster Configuration
   -> Cluster Topology
         -> Configure Adapters
               -> Add an Adapter
```

HACMP will complain if a boot or standby adapter is configured before their corresponding service adapter. However, HACMP will not allow a service adapter to be configured before its boot adapter is configured. So configure the boot adapters first and ignore the warnings.



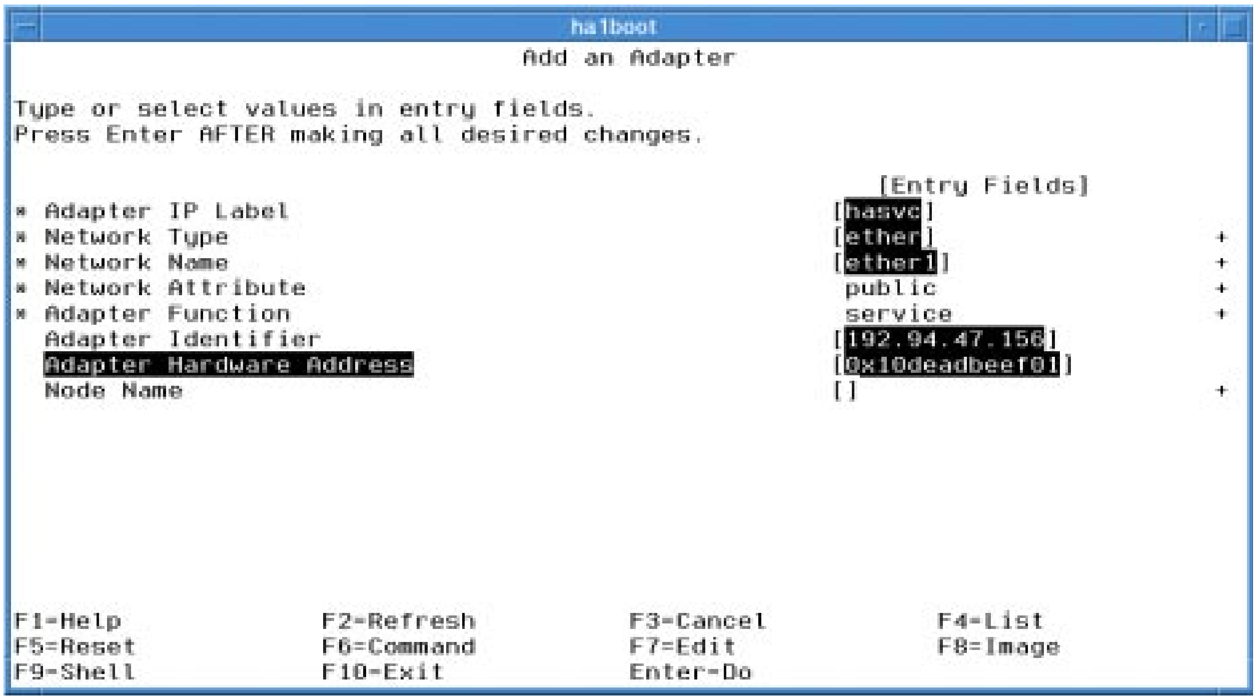
```
                                ha1boot
                            Add an Adapter

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                          [Entry Fields]
* Adapter IP Label                                     [ha1boot]
* Network Type                                         [ether]             +
* Network Name                                         [ether1]            +
* Network Attribute                                     public             +
* Adapter Function                                      boot               +
  Adapter Identifier                                   [192.94.47.244]
  Adapter Hardware Address                             []
  Node Name                                            [hanode1]           +

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure H-6 Adding an Ethernet Boot Adapter



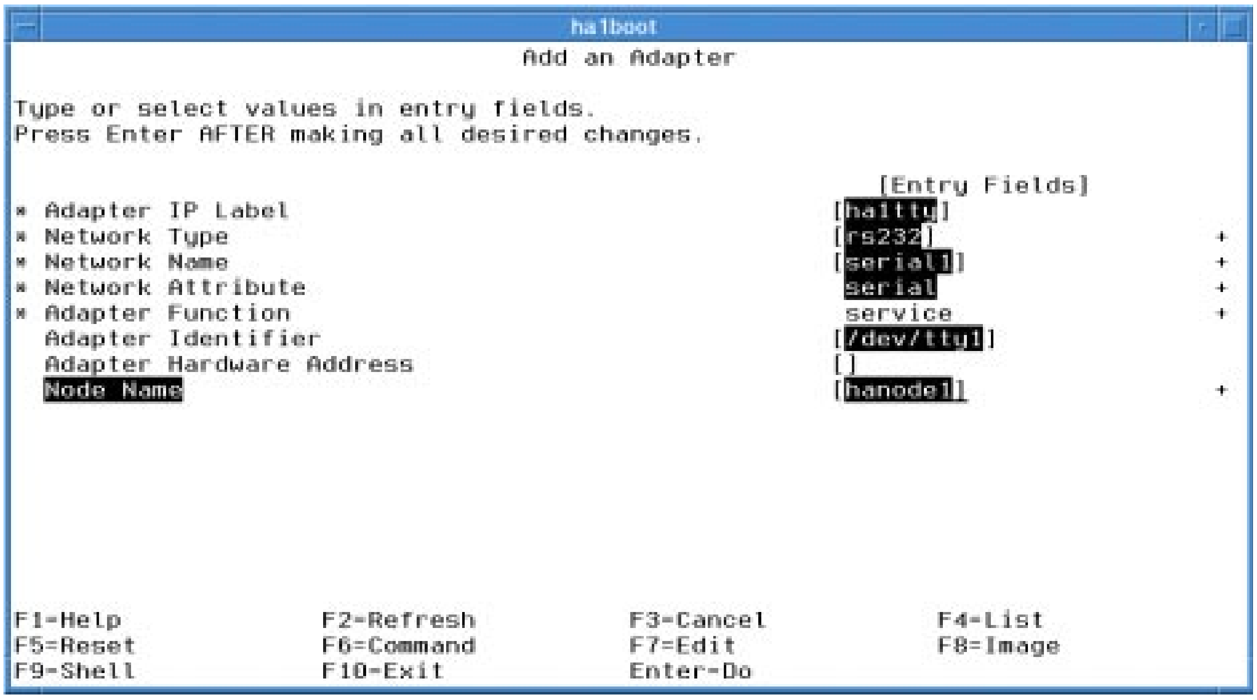
```
                                ha1boot
                            Add an Adapter

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                          [Entry Fields]
* Adapter IP Label                                     [ha1stby]
* Network Type                                         [ether]             +
* Network Name                                         [ether1]            +
* Network Attribute                                     public             +
* Adapter Function                                      standby            +
  Adapter Identifier                                   [192.94.48.145]
  Adapter Hardware Address                             []
  Node Name                                            [hanode1]           +

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure H-7 Adding an Ethernet Standby Adapter



```
                                    ha1boot
                                 Add an Adapter

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                          [Entry Fields]
* Adapter IP Label                                     [hasvc]
* Network Type                                         [ether]                  +
* Network Name                                         [ether1]                 +
* Network Attribute                                     public                  +
* Adapter Function                                      service                 +
  Adapter Identifier                                   [192.94.47.156]
  Adapter Hardware Address                             [0x10deadbeef01]
  Node Name                                            []                       +

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure H-8 Adding an Ethernet Service Adapter



```
                                    ha1boot
                                 Add an Adapter

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                          [Entry Fields]
* Adapter IP Label                                     [ha1tty]
* Network Type                                         [rs232]                  +
* Network Name                                         [serial1]                +
* Network Attribute                                     serial                  +
* Adapter Function                                      service                 +
  Adapter Identifier                                   [/dev/tty1]
  Adapter Hardware Address                             []
  Node Name                                            [hanode1]                +

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure H-9 Adding a Serial Adapter

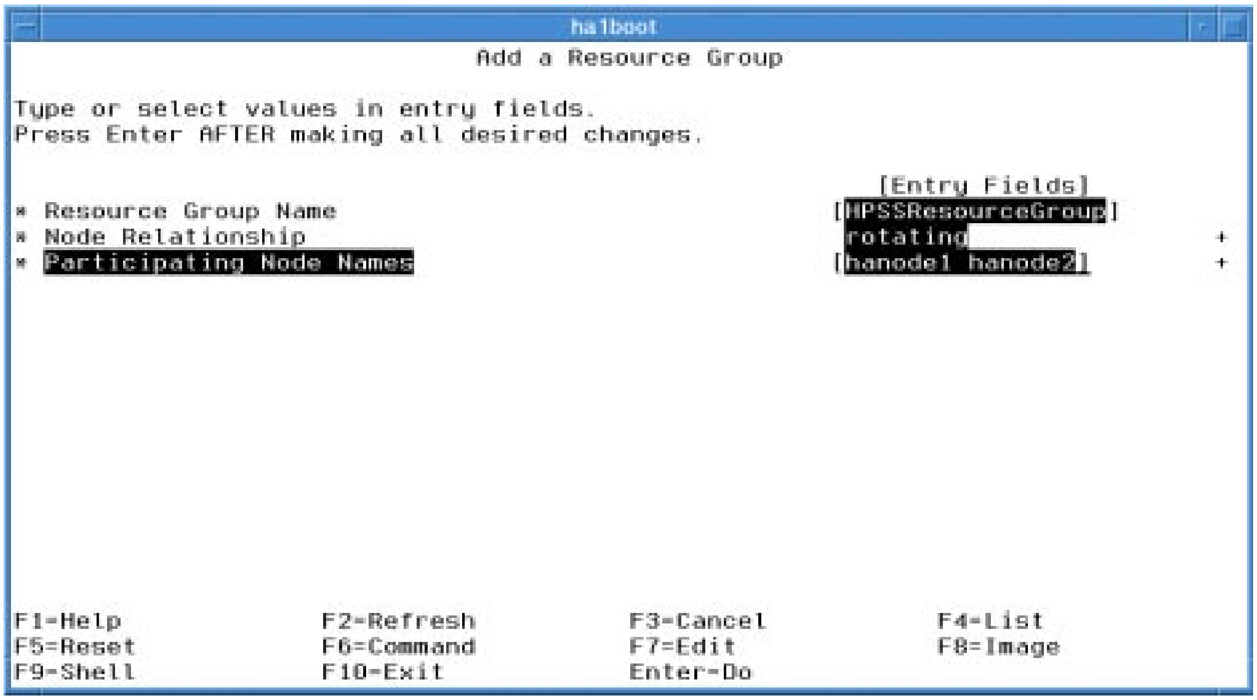
G.4.3.5 Synchronize the Cluster Topology

By this point in the configuration, you should have given HACMP all the information it needs about the topology of your networks. However, only one of the nodes has this configuration information, and it needs to be on both nodes. So, it's time to synchronize the cluster topology. See Section G.10.4.1: *Synchronize Topology* on page 573 for instructions.:

G.4.3.6 Define the resource groups

To define a resource group, associate the name of the resource group with its type and the nodes involved.

```
% smitty hacmp

Cluster Configuration
   -> Cluster Resources
         -> Define Resource Groups
               -> Add a Resource Group
```



```
ha1boot
                         Add a Resource Group

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                    [Entry Fields]
* Resource Group Name                             [HPSSResourceGroup]
* Node Relationship                                rotating                 +
* Participating Node Names                        [hanode1 hanode2]         +

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

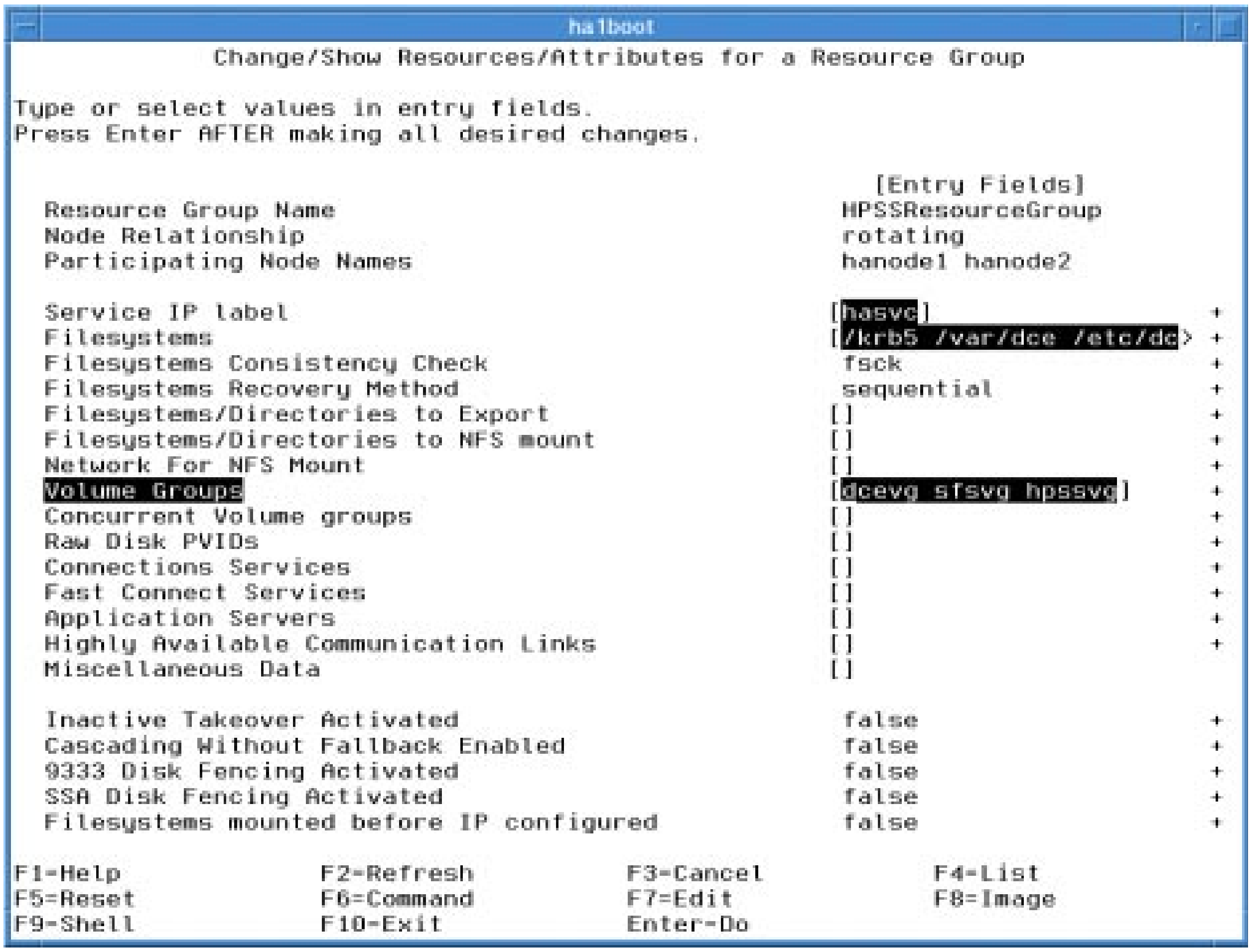
Figure H-10 Adding a Resource Group

G.4.3.7 Assign Resources to the Resource Group

```
% smitty hacmp

Cluster Configuration
```

```
-> Cluster Resources
     -> Change/Show Resources/Attributes for a Resource Group
          [HPSSResourceGroup]
```

There are only three fields that need to be filled in on the SMIT screen at this time: **Service IP Label**, **Filesystems**, and **Volume Groups**. Special care should be taken to ensure that all shared file systems and volume groups are listed (it's usually easy if you use the F4 pick lists):



```
                                 ha1boot
          Change/Show Resources/Attributes for a Resource Group

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                    [Entry Fields]
  Resource Group Name                                HPSSResourceGroup
  Node Relationship                                  rotating
  Participating Node Names                           hanode1 hanode2

  Service IP label                                  [hasvc]                 +
  Filesystems                                       [/krb5 /var/dce /etc/dc> +
  Filesystems Consistency Check                      fsck                   +
  Filesystems Recovery Method                        sequential             +
  Filesystems/Directories to Export                 []                      +
  Filesystems/Directories to NFS mount              []                      +
  Network For NFS Mount                             []                      +
  Volume Groups                                     [dcevg sfsvg hpssvg]    +
  Concurrent Volume groups                          []                      +
  Raw Disk PVIDs                                    []                      +
  Connections Services                              []                      +
  Fast Connect Services                             []                      +
  Application Servers                               []                      +
  Highly Available Communication Links              []                      +
  Miscellaneous Data                                []

  Inactive Takeover Activated                        false                  +
  Cascading Without Fallback Enabled                 false                  +
  9333 Disk Fencing Activated                        false                  +
  SSA Disk Fencing Activated                         false                  +
  Filesystems mounted before IP configured           false                  +

F1=Help              F2=Refresh           F3=Cancel            F4=List
F5=Reset             F6=Command           F7=Edit              F8=Image
F9=Shell             F10=Exit             Enter=Do
```

Figure H-11 Configuring a Resource Group

G.4.3.8 *Synchronize Cluster Resources*

By this point in the configuration, you should have given HACMP all the information it needs about your resource group. However, only one of the nodes has this configuration information, and it needs to be on both nodes. So, it's time to synchronize the cluster resources. See Section G.10.4.2: *Synchronize Resources* on page 573 for instructions.

G.4.4 *Configure DCE, SFS, and HPSS*

G.4.4.1 *Start the Cluster*

To continue configuring DCE, Encina, and HPSS, it will be necessary to start the HACMP cluster. This will cause one of the nodes to acquire the service address, vary on the shared volume groups, and mount the shared file systems.

For information on how to start cluster, see Section G.10.1: *Startup the Cluster* on page 570.

G.4.4.2 *Install Shared Software*

Now that the cluster is up and running, it is possible to install DCE, SFS, Sammi, HPSS, and any other software that must reside on the shared disks (i.e. Java). To determine versions, refer to the HPSS Compatibility Chart at `http://www4.clearlake.ibm.com/hpss/about/ServerTable.jsp`. However, there are a couple of extra steps that will have to be taken to ensure that both nodes are installed correctly. Here is the procedure:

1. Install *all* the software on the active node (the node with the service address, shared volume groups, and shared file systems).

2. Move the HPSS resource group to the second node (Refer to Section G.10.5: *Move a Resource Group* on page 574).

3. Install *all* the software on the newly active node. This is necessary because not all installed files reside on the shared disks. In order to get these locally stored files properly installed on both nodes, a second complete install of all software must be performed on both nodes

4. Move the HPSS resource group back to the original node.

G.4.4.3 *Configure HPSS As Usual*

Now that all the software is properly installed, configuration may proceed as usual starting with Chapter 5: *HPSS Infrastructure Configuration* (page 215). There is no need to worry about moving resource groups or doing configurations multiple times. The configuration steps only affect files on the shared file systems, so proceed as usual.

One thing that will need to be done is to copy the /etc/rc.hpss file to both nodes. By default, it will only be copied to the /etc directory of the node that is active when HPSS is being configured.

G.5 *Configure HA HPSS*

G.5.1 *HA HPSS Scripts*

HACMP is able to control HPSS by using a set of scripts that are included in the HPSS installation under **$HPSS_ROOT/tools/ha** (by default, **/opt/hpss/tools/ha**):

```
hpss_environment
hpss_start.ksh
hpss_stop.ksh
hpss_sync.ksh
hpss_verify.ksh
hpss_snapshot.ksh
hpss_notify.ksh
hpss_aix_error.ksh
hpss_cluster_notify.ksh
```

These scripts need to be stored locally on each node's internal disks, not on shared storage. They will also have to be customized to operate within any particular HA HPSS environment. To ready the scripts, take the following steps:

1. Copy the scripts to a directory on the **rootvg** of one of the HA HPSS nodes (we'll call this node 1 for the remainder of these steps). We'll assume that the scripts are copied to **/var/hahpss**.

2. Customize the **hpss_environment** file. Pay special attention to the following fields:

   HAHPSS_HPSSADM_SUPPORT - Must be set to **off**. This option is not yet supported.

   HAHPSS_PATH - Must be set to the absolute path to the directory containing these scripts. This would be **/var/hahpss** in this example.

3. Customize the script paths to the **hpss_environment** file. Each script file must be able to find the hpss_environment file to pick up configuration settings. However, there is no guarantee that the scripts will be called from any particular directory, therefore they have to know the absolute pathname of the hpss_environment file in order to find it reliably. Edit all the **.ksh** files, and update the path in the following line:

   ```
   . /usr/sbin/cluster/events/hpss/hpss_environment
   ```

   For example, the updated line will most likely be:

   ```
   . /var/hahpss/hpss_environment
   ```

4. Customize **hpss_notify.ksh** if desired. This file is called when an event occurs that may be interesting to a system administrator. The default script outputs the information to a file (default: **hpss_notify_hist**). However, the intent is that sites may modify this file to send email or page administrators when messages come through.

   The script is called with two arguments. The first is a text string relaying the basic message, and the second contains particular details. For example, the first string may say, "**A hard disk has failed.**" While the second might say, "**hdisk2**".

5. Change the owner and mode bits.

   ```
   % cd /var/hpss
   % chown root:hpss *
   % chmod u+x *.ksh
   ```

6. Create **/var/hahpss** (or corresponding directory) on node 2.

7. Synchronize the scripts using **hpss_sync.ksh**. If you are setup properly with your **/.rhost** files, run the following from the script directory:

   % ./hpss_sync.ksh <other node's standby>

   Where **<other node's standby>** is the standby address of node 2. This will copy the HA HPSS scripts from the current node (node 1) to the specified node (node 2). Any time changes are made to these scripts, it will be necessary to resynchronize using this command.

G.5.1.1 Build hpss_start_list

Now that configuration is done, the next step is to create a file that will enable HACMP to startup all of the HPSS servers, **hpss_start_list**. When finished, the contents of **hpss_start_list** should be something like this (lines are wrapped):

```
./hpss_logd "Log Daemon" /.:/encina/sfs/hpss/globalconfig hpss_log /
krb5/hpss.keytabs 2 &
./hpss_bfs "BFS" /.:/encina/sfs/hpss/globalconfig hpss_bfs /krb5/
hpss.keytabs 2 &
./hpss_mmon "Metadata Monitor" /.:/encina/sfs/hpss/globalconfig
hpss_mmon /krb5/hpss.keytabs 2 &
./hpss_logc "Log Client (hanode1)" /.:/encina/sfs/hpss/globalconfig
hpss_log /krb5/hpss.keytabs 2 &
./hpss_mvr "Mover (hanode1)" /.:/encina/sfs/hpss/globalconfig hpss_mvr
/krb5/hpss.keytabs 2 &
./hpss_cns "Name Server" /.:/encina/sfs/hpss/globalconfig hpss_cns /
krb5/hpss.keytabs 2 &
./hpss_ls "Location Server" /.:/encina/sfs/hpss/globalconfig hpss_ls /
krb5/hpss.keytabs 2 &
./hpss_ss_disk "Disk Storage Server" /.:/encina/sfs/hpss/globalconfig
hpss_ss /krb5/hpss.keytabs 2 &
./hpss_mvr "Mover (mover3)" /.:/encina/sfs/hpss/globalconfig hpss_mvr
/krb5/hpss.keytabs 2 &
./hpss_ndcg "Non-DCE Gateway" /.:/encina/sfs/hpss/globalconfig
hpss_ndcg /krb5/hpss.keytabs 2 &
./hpss_pvl "PVL" /.:/encina/sfs/hpss/globalconfig hpss_pvl /krb5/
hpss.keytabs 2 &
./hpss_mps "Migration/Purge Server" /.:/encina/sfs/hpss/globalconfig
hpss_mps /krb5/hpss.keytabs 2 &
```

Follow these steps to create **hpss_start_list** for your system:

1. Ensure all HPSS servers are up and running

2. Go to the directory on the active node that contains your scripts (**/var/hahpss**).

3. Use the a process table dump as a starting point:

   ```
   % ps -ef | grep hpss > hpss_start_list
   ```

4. Remove lines that meet any of the following criteria:

   ➢ Executable name doesn't begin with **hpss**

   ➢ Executable name is **hpssd**, **hpss_ssmds**, or **hpss_ssmsm**

   ➢ The process is a Non-DCE Client Gateway subprocess, **hpss_ndcg_***

   ➢ The process is a Mover subprocess, **hpss_mvr_***

5. From the beginning of each line, replace all the text preceeding the executable name with "`./`".

6. On each line, place double quotes around the server names.

7. At the end of each line, put a " **&**".

8. Compare what you have now to the example above, and it should be very similar.

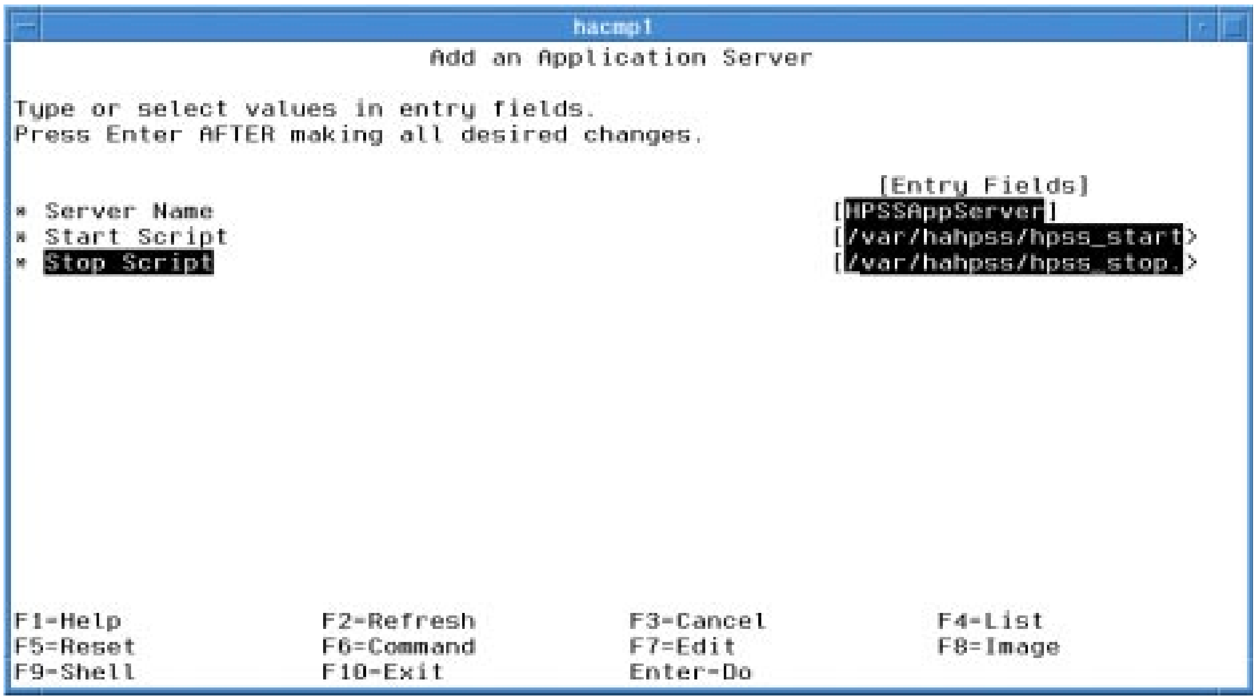
9. Now make sure that it's readable by root, and synchronize the HA HPSS files between nodes (this will pick up the new **hpss_start_list** as well as the scripts). For information on how to synchronize, see Section G.10.4.3: *Synchronize HA HPSS Files* on page 573.

G.5.2 Finish the HACMP Configuration

G.5.2.1 Define the application server

All that makes up the definition of an application server are a name and start/stop scripts.

```
% smitty hacmp

Cluster Configuration
   -> Cluster Resources
         -> Define Application Servers
               -> Add an Application Server
```



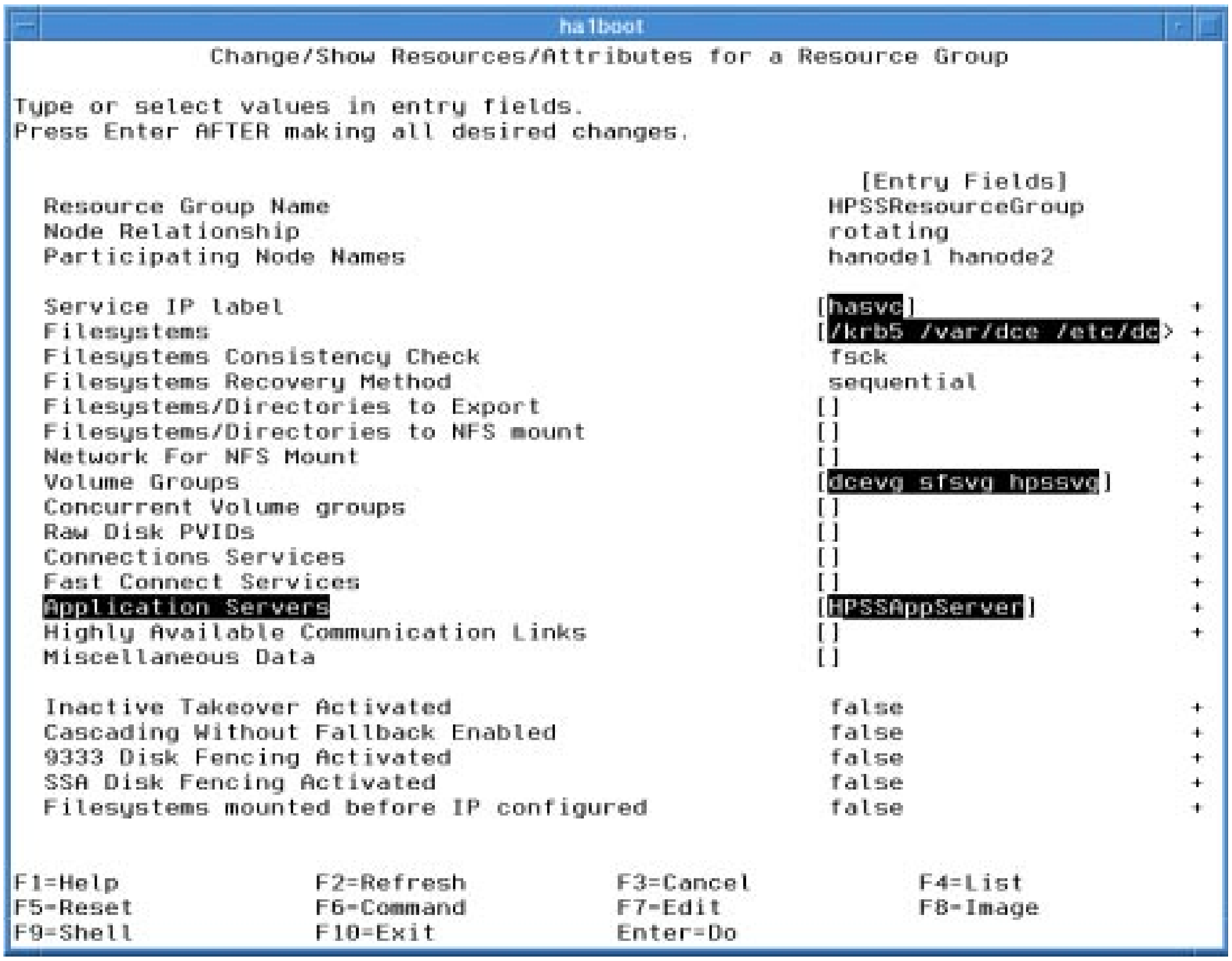
```
hacmp1
                        Add an Application Server

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                          [Entry Fields]
* Server Name                                          [HPSSAppServer]
* Start Script                                         [/var/hahpss/hpss_start>
* Stop Script                                          [/var/hahpss/hpss_stop.>

F1=Help          F2=Refresh         F3=Cancel          F4=List
F5=Reset         F6=Command         F7=Edit            F8=Image
F9=Shell         F10=Exit           Enter=Do
```

Figure H-12 Adding an Application Server

G.5.2.2 Attach the Application Server to the Resource Group

```
% smitty hacmp

Cluster Configuration
   -> Cluster Resources
         -> Change/Show Resources/Attributes for a Resource Group
               [HPSSResourceGroup]
```

There are only four fields that need to be filled in on this SMIT screen: “Service IP Label”, “Filesystems”, “Volume Groups”, and ”Application Servers”. Special care should be taken to ensure that all shared file systems and volume groups are listed (it’s usually easy if you use the F4 pick lists).



```
                                    ha1boot
             Change/Show Resources/Attributes for a Resource Group

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                          [Entry Fields]
   Resource Group Name                                   HPSSResourceGroup
   Node Relationship                                     rotating
   Participating Node Names                              hanode1 hanode2

   Service IP label                                     [hasvc]                 +
   Filesystems                                          [/krb5 /var/dce /etc/dc> +
   Filesystems Consistency Check                         fsck                   +
   Filesystems Recovery Method                           sequential             +
   Filesystems/Directories to Export                    []                      +
   Filesystems/Directories to NFS mount                 []                      +
   Network For NFS Mount                                []                      +
   Volume Groups                                        [dcevg sfsvg hpssvg]    +
   Concurrent Volume groups                             []                      +
   Raw Disk PVIDs                                       []                      +
   Connections Services                                 []                      +
   Fast Connect Services                                []                      +
   Application Servers                                  [HPSSAppServer]         +
   Highly Available Communication Links                 []                      +
   Miscellaneous Data                                   []

   Inactive Takeover Activated                           false                  +
   Cascading Without Fallback Enabled                    false                  +
   9333 Disk Fencing Activated                           false                  +
   SSA Disk Fencing Activated                            false                  +
   Filesystems mounted before IP configured              false                  +


F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure H-13 Adding an Application Server to a Resource Group

G.5.2.3 *Bring Down the Cluster*

Now that the cluster is back in sync and that it knows how to run the **hpss_stop.ksh** script, it's time to bring down the cluster. However, HACMP doesn't yet know how to stop HPSS, SFS, and DCE (it won't know that until the cluster is synchronized in the next step), so you'll need to shut HPSS, SFS, and DCE down manually.

After HPSS, SFS, and DCE are down, bring down the cluster. Refer to Section G.10.2: *Shutdown the Cluster* on page 571 for details.

G.5.2.4 *Re-Synchronize the Cluster Resources*

This is just like synchronizing resources like you did in Section G.4.3.8: *Synchronize Cluster Resources* on page 557.

G.5.2.5 Bring Up the Cluster (just to test that it works)

Now bring the cluster back up using the instructions in Section G.10.1: *Startup the Cluster* on page 570.

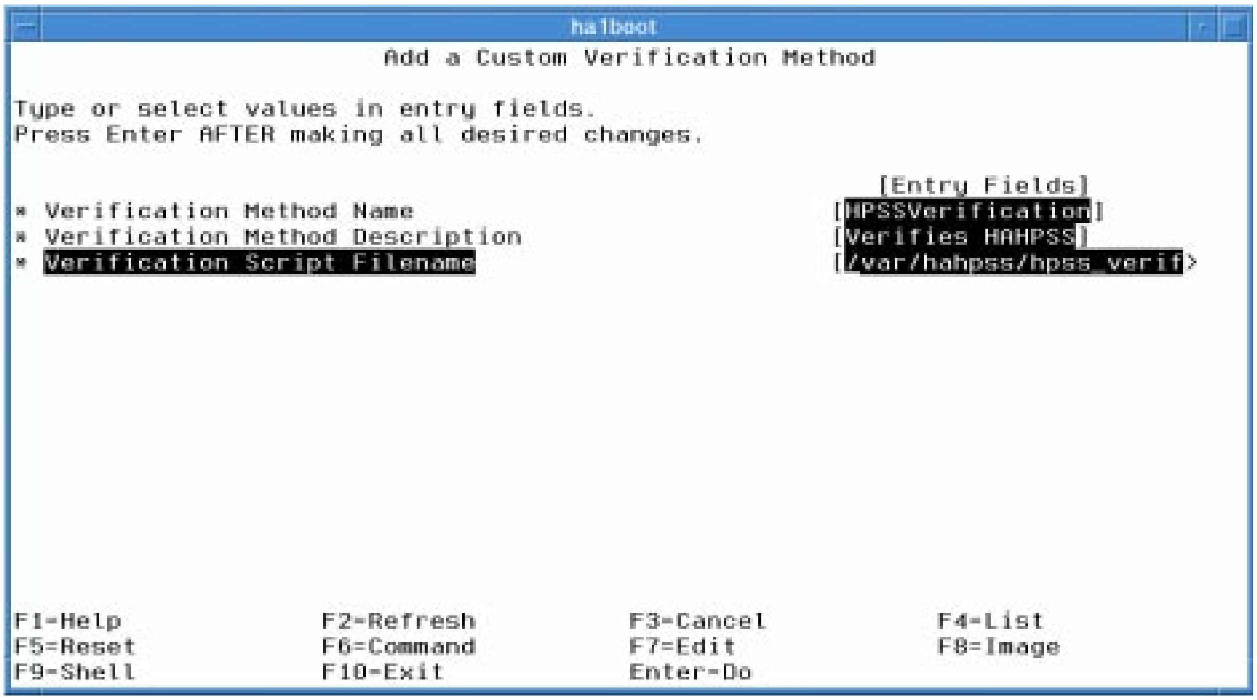
When the cluster is active again, HPSS will be fully operational. It will be able to service requests immediately, and all an administrator needs to do is start an SSM session to begin administering the system.

G.5.3 Define HA HPSS Verification Method

HACMP has the capability to verify its own configuration, to a certain extent. As an extension to this capability, the **hpss_verify.ksh** will aid in finding potential problems with an HA HPSS configuration.

Setup custom HA HPSS verification using:

```
% smitty hacmp

Cluster Configuration
   -> Cluster Verification
         -> Define Custom Verification Method
               -> Add a Custom Verification Method
```



```
ha1boot
                        Add a Custom Verification Method

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                         [Entry Fields]
* Verification Method Name                            [HPSSVerification]
* Verification Method Description                     [Verifies HAHPSS]
* Verification Script Filename                        [/var/hahpss/hpss_verif>

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure H-14 Adding a Custom Verification Method

G.5.4 Setup Error Notification

Even though an HA HPSS system is designed to recover from failures, the recovered system is often unable to handle subsequent failures. For this reason, it is important that administrators know immediately when a component fails so that it can be replaced or fixed quickly in order to get the HA HPSS system back to a highly available state.

This is why the **hpss_notify.ksh** script is supplied. If you have customized your **hpss_notify.ksh** to email your system administrator, all that remains is to direct error messages to the **hpss_notify.ksh**.

This can be done in two ways, through AIX Error Notification and through HACMP Notify Events.

G.5.4.1 AIX Error Notification

When configuring AIX Error Notification, use the following SMIT path:

```
% smitty hacmp

RAS Support
   -> Error Notification
         -> Add a Notify Method
```

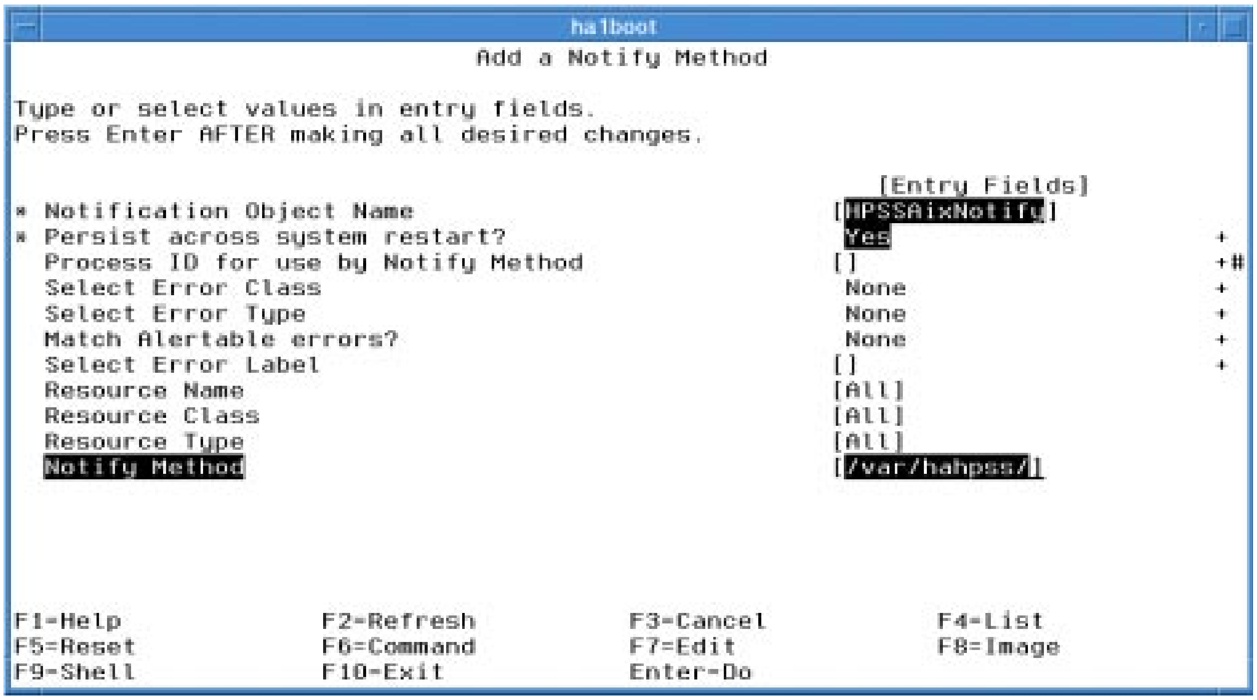
The notify method should call the **hpss_aix_error.ksh** script for **all** AIX errors. The script filters out particular errors that are interesting. These include power supply, disk, and SCSI bus failures.

Unfortunately, there is no easy way to synchronize both nodes' error notification settings. Therefore, these steps will have to be performed on each node independently.

The notify method should call the **hpss_aix_error.ksh** script with the following syntax:

```
hpss_aix_error.ksh $1 $2 $3 $4 $5 $6 $7 $8
```

It is also important that the **Persist across system restart** field be set to **Yes**.



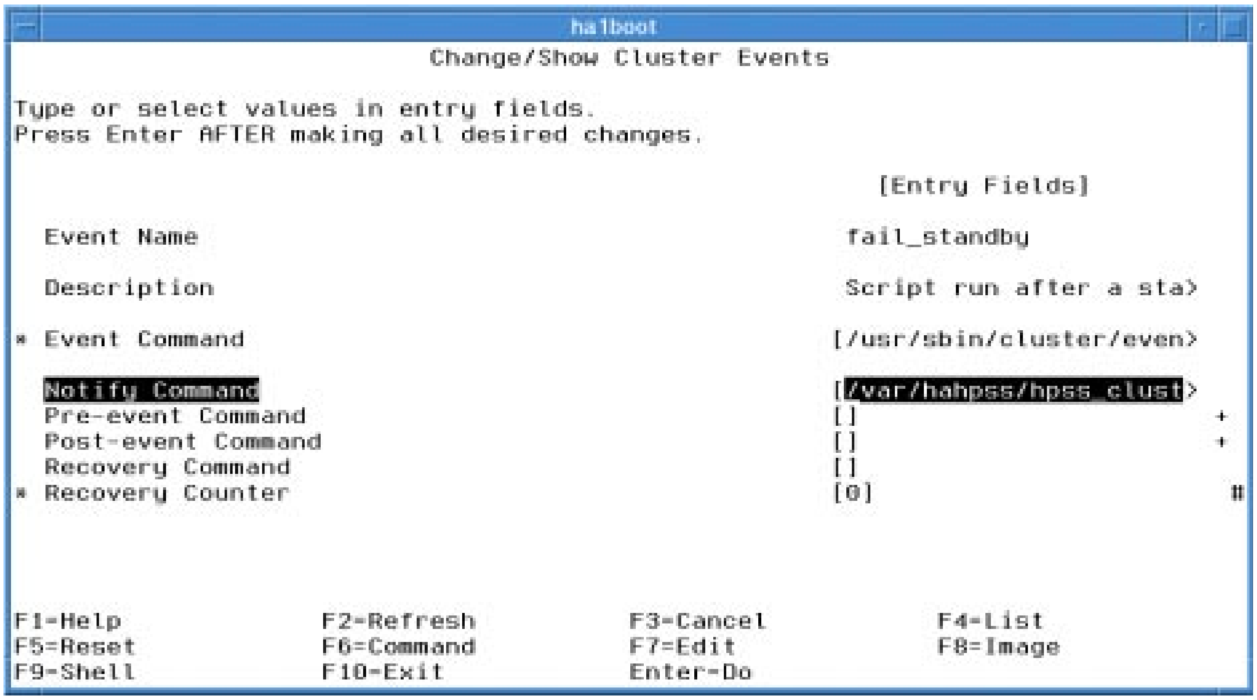
```
                                  ha1boot
                            Add a Notify Method

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                            [Entry Fields]
* Notification Object Name                              [HPSSAixNotify]
* Persist across system restart?                         Yes              +
  Process ID for use by Notify Method                   []                +#
  Select Error Class                                     None             +
  Select Error Type                                      None             +
  Match Alertable errors?                                None             +
  Select Error Label                                    []                +
  Resource Name                                         [All]
  Resource Class                                        [All]
  Resource Type                                         [All]
  Notify Method                                         [/var/hahpss/]

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure H-15 Configuring AIX Error Notification

G.5.4.2 HACMP Notify Events

When some failures occur, they generate events in HACMP. These events can be configured to cause a notification to be sent before and after the event occurs using the **hpss_cluster_notify.ksh** script. To do this:

```
% smitty hacmp

Cluster Configuration
   -> Cluster Resources
         -> Cluster Events
               -> Change/Show Cluster Events
                     [Choose the event]
```



```
                                    ha1boot
                          Change/Show Cluster Events

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                        [Entry Fields]

  Event Name                                           fail_standby

  Description                                          Script run after a sta>

* Event Command                                       [/usr/sbin/cluster/even>

  Notify Command                                      [/var/hahpss/hpss_clust>
  Pre-event Command                                   []                      +
  Post-event Command                                  []                      +
  Recovery Command                                    []
* Recovery Counter                                    [0]                     #



F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure H-16 Configuring Cluster Event Notification

Fill in the **Notify Command** field using the following syntax:

```
hpss_cluster_notify.ksh
```

There are no arguments to pass.

The events that you should consider setting up this way include:

```
fail_standby
join_standby
network_down
network_up
node_down
node_up
node_up_complete
swap_adapter
```

It may be necessary to setup these event notification on each node independently.

G.5.5 crontab Considerations

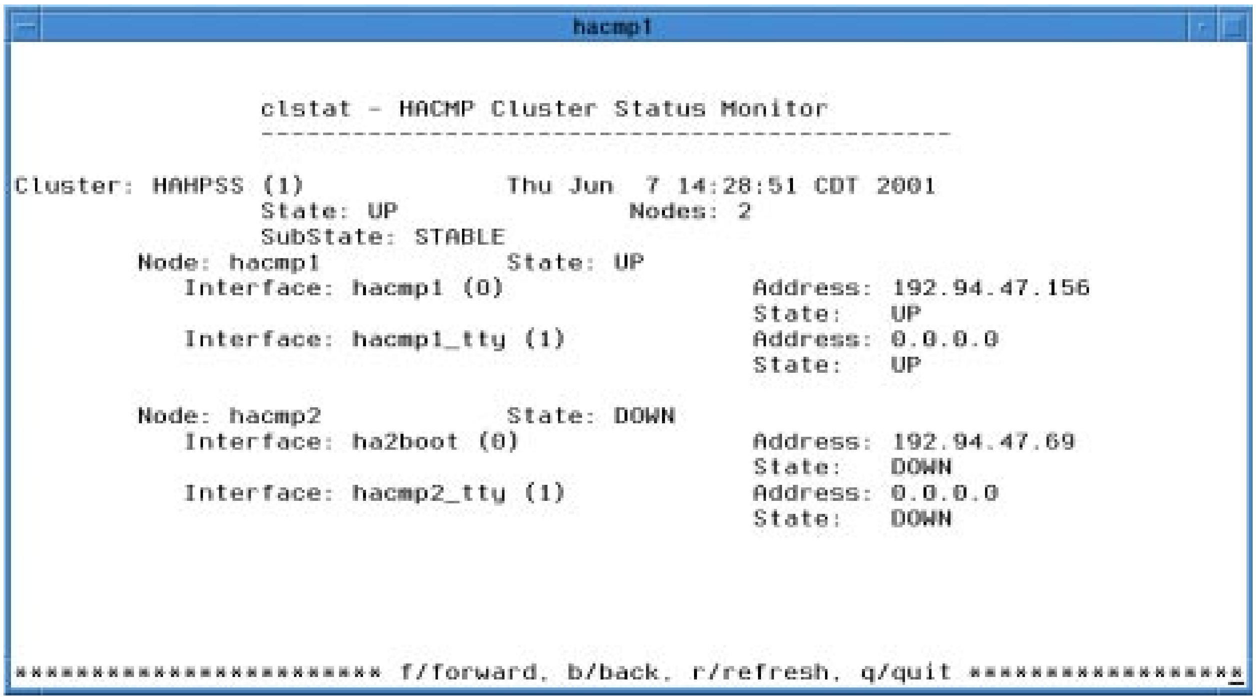
Since crontab files are stored on the rootvg of each node, it is important that they be kept in sync between the two nodes of an HA HPSS configuration. Even more important, perhaps, is modifying your crontabs to work correctly in the event that all the HPSS, DCE, Encina, Java, and Sammi file systems are currently unavailable (which will always be true for at least one of the nodes).

The answer to this problem will depend from site to site, but one good way to make this work is to have a set of intermediate scripts between the crontab file and the commands it executes. These scripts could test for the existence of any prerequisite files and / or file systems and only execute the associated command if all the prerequisites are met. This is exactly the strategy we recommend for the crontab entries for the **sfs_backup_util** in Section G.7: *Metadata Backup Considerations* on page 569.

G.6 Monitoring and Maintenance

G.6.1 clstat

One quick way to get an overview of a cluster's current status is using **/usr/sbin/cluster/clstat**. This utility will report on which nodes are currently a part of the cluster and what interfaces are currently available on them (boot, service, standby). It can run in either ASCII mode or X mode depending on what services are available and how it is invoked. Below is a sample ASCII output:



```
                clstat - HACMP Cluster Status Monitor
                ---------------------------------------------

Cluster: HAHPSS (1)           Thu Jun  7 14:28:51 CDT 2001
                State: UP               Nodes: 2
                SubState: STABLE
        Node: hacmp1            State: UP
           Interface: hacmp1 (0)                Address: 192.94.47.156
                                                State:   UP
           Interface: hacmp1_tty (1)            Address: 0.0.0.0
                                                State:   UP

        Node: hacmp2            State: DOWN
           Interface: ha2boot (0)               Address: 192.94.47.69
                                                State:   DOWN
           Interface: hacmp2_tty (1)            Address: 0.0.0.0
                                                State:   DOWN




************************ f/forward, b/back, r/refresh, q/quit *****************
```

Figure H-17 sample output from 'clstat -a'

In order for **clstat** to work, the **clinfo** daemon must be running. When cluster services are started, there is an option to start **clinfo** or not. The default is not to start it, so it will have to be started purposely in order for **clstat** to be utilized.

See the **clstat** man page for more information.

G.7 Metadata Backup Considerations

Special care should be taken when configuring your HA HPSS system to work with the **sfs_backup_util** to keep the source, configuration, and backup (LA and TRB) files stored on the shared disks. Otherwise a failover could easily make backup files or the **sfs_backup_util** program itself unreachable.

Also, crontab entries related to SFS Backup will require special attention. First, they must be kept up-to-date and synchronized between both nodes of the cluster. Second, some provision must be made for the fact that only one of the nodes will be able to run the SFS Backup utility at any given time (since it resides on a shared file system).

Keeping the crontabs for both nodes in sync is an AIX administration problem that is beyond the scope of this manual; however, there is a good method to use in order to address the problem of certain commands being unavailable on the backup node. Put your normal command in a script which will check for the existence of the executable before running the command, and call the script from your crontab.

For example, if your crontab entry normally looked like this:

```
0 * * * * /opt/hpss/tools/sfs/sfs_backup_util/run/run.ksh
```

Then create the a script stored on the local node (`/var/run.ksh`) like this:

```
#!/bin/ksh
# Check for existence of the run.ksh before execution
if [ -a /opt/hpss/tools/sfs/sfs_backup_util/run/run.ksh ]; then
   /opt/hpss/tools/sfs/sfs_backup_util/run/run.ksh
fi
```

And replace your normal crontab entry with a call to the script:

```
0 * * * * /var/run.ksh
```

This will ensure that your command gets executed on schedule when it is available, and it will keep cron from sending you error messages when the command is not available.

Just make sure that you make ALL these changes on BOTH nodes in your cluster.

G.8 Using Secure Shell (ssh)

Normally, HA HPSS requires that **rsh** and **rcp** commands be available in order to function properly. However, it is possible to configure it to use **ssh** and **scp** instead for environments where **/.rhost** files pose too much of a security risk.

HACMP itself depends heavily on the use of **rsh**, so refer to HACMP documentation for information on what features might not work as expected.

Most dependencies are for configuration and synchronization, which hopefully will be mostly finished at install time. It should not affect the ability of the HA HPSS system to fail over properly.

After setting up ssh and scp for your cluster, follow these steps to configure HA HPSS to use them:

1. Edit the **/var/hahpss/hpss_environment** file on either of the cluster nodes, and set the following environment variables:

   ```
   HPSS_REMOTE_SHELL=ssh
   HPSS_REMOTE_COPY=scp
   ```

2. Now synchronize this change to the other node:

   ```
   % ./hpss_sync.ksh <other_node>
   ```

If **ssh** was installed and configured properly (and if the **sshd** was started on both nodes) the above steps will complete your conversion to an **ssh**-enabled HA HPSS system.

G.9 Important Information

When using HACMP to manage shared resources, there are many differences from a normal AIX system that must be taken into account. Addressing all these differences is beyond the scope of this document but it nonetheless very important. It is an absolute necessity that administrators familiarize themselves with the HACMP for AIX Administration Guide in order to understand these issues.

For example, updating volume groups and file systems will have to be done using HACMP's SMIT menus, not the normal LVM commands. Otherwise, the ODMs on the two cluster nodes will get out of sync.

For the most helpful troubleshooting advice, see the *HACMP for AIX Troubleshooting Guide*.

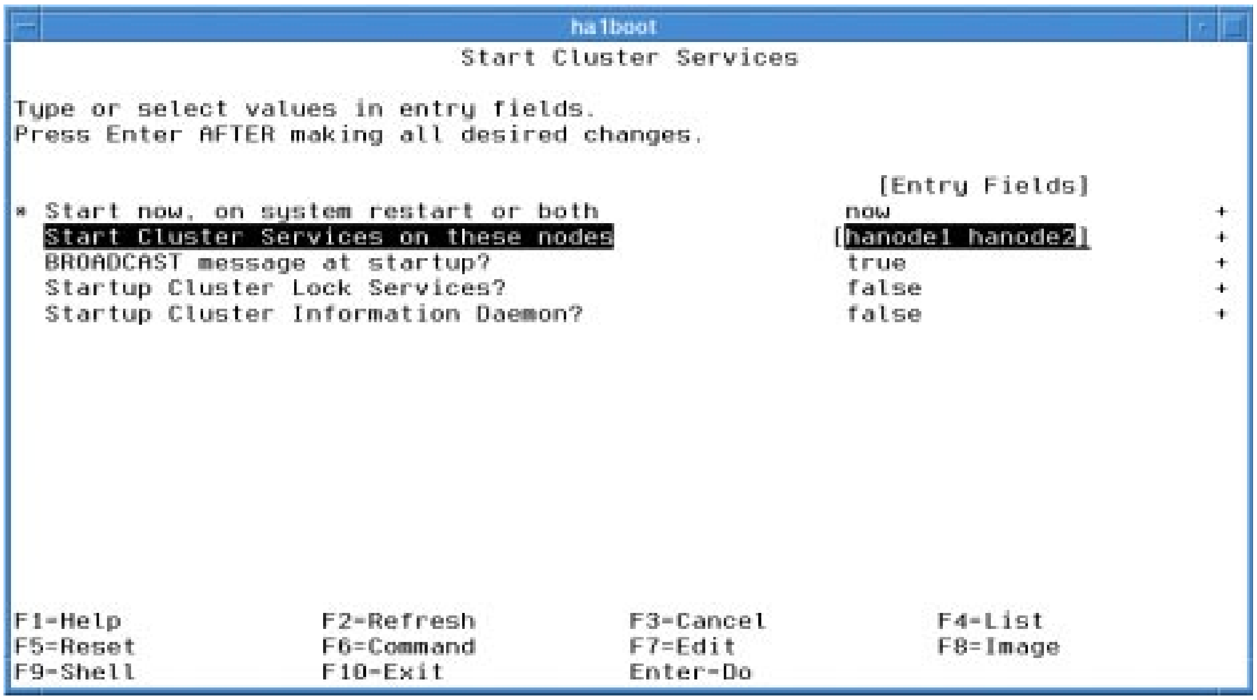
G.10Routine HA Operations

G.10.1 Startup the Cluster

To start the HACMP cluster (the HACMP Cluster Manager) on the cluster nodes, there are two methods.

1. The first method is the most convenient; however, it can only be used if **rsh** is enabled. It allows the Cluster Manager to be started on both nodes with a single command:

   ```
   % smitty hacmp

   Cluster System Management
         -> HACMP Cluster Services
               -> Start Cluster Services
   ```



```
ha1boot
                         Start Cluster Services

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                        [Entry Fields]
* Start now, on system restart or both                  now                    +
  Start Cluster Services on these nodes                [hanode1 hanode2]       +
  BROADCAST message at startup?                         true                   +
  Startup Cluster Lock Services?                        false                  +
  Startup Cluster Information Daemon?                   false                  +

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure H-18 Starting Cluster Services

2. Alternatively, it is possible to use a slightly different SMIT path to start the Cluster Manager on the local node. Of course, this requires logging into each node independently to activate both Cluster Managers.

```
% smitty hacmp

Cluster Services
      -> Start Cluster Services
```

Take the defaults and press <Enter>.

G.10.2 Shutdown the Cluster

The procedure for shutting the cluster down is almost exactly the same as starting the cluster.

The SMIT path that will let you shut down both nodes at once is:

```
% smitty hacmp

Cluster System Management
   -> HACMP Cluster Services
         -> Stop Cluster Services
```

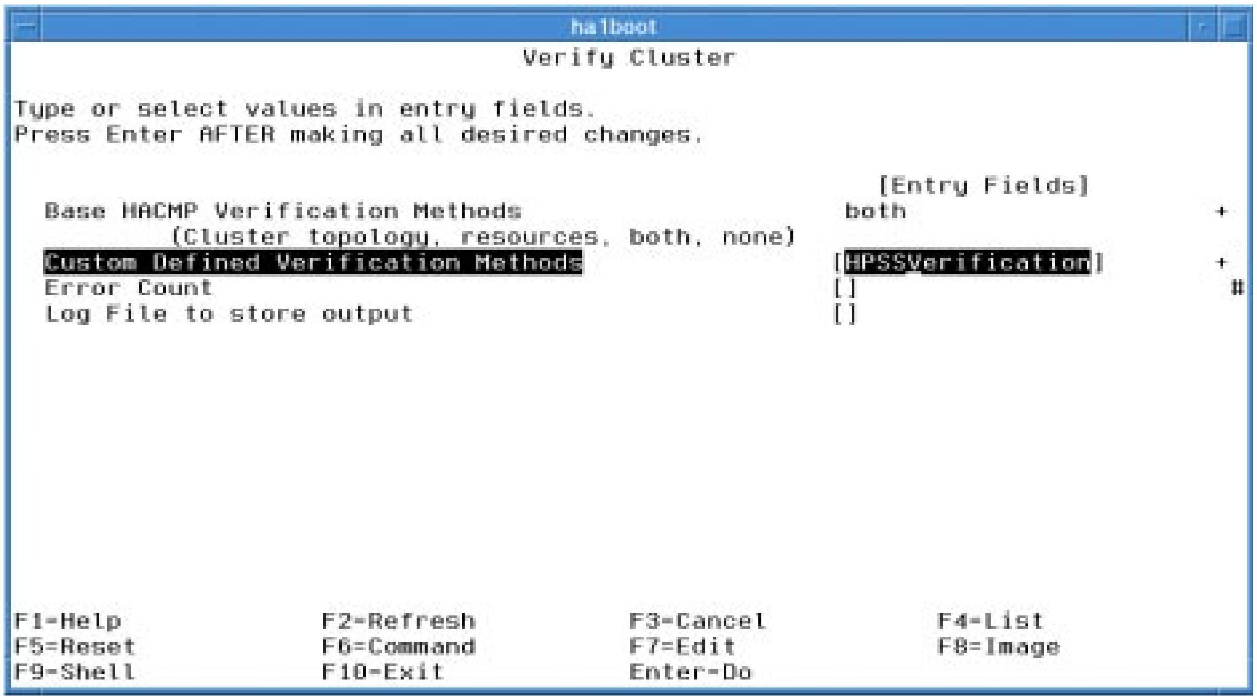
This SMIT path that requires shutting down both nodes independently is:

```
% smitty hacmp

Cluster Services
   -> Stop Cluster Services
```

G.10.3 Verify the Cluster

Once the HA HPSS verification method has been defined in HACMP (Section G.5.3: *Define HA HPSS Verification Method* on page 564), go to SMIT to verify your cluster:

```
% smitty hacmp

Cluster Configuration
   -> Cluster Verification
         -> Verify Cluster
```



```
                             Verify Cluster

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                      [Entry Fields]
  Base HACMP Verification Methods                     both                   +
          (Cluster topology, resources, both, none)
  Custom Defined Verification Methods                [HPSSVerification]      +
  Error Count                                        []                       #
  Log File to store output                           []

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure H-19 Verifying the Cluster

The output from this will include both HACMP's normal checks and the HA HPSS-specific checks. You will likely have to scroll through several pages of output to check for reported errors.

G.10.4 Synchronize the Cluster

G.10.4.1 Synchronize Topology

In order to synchronize topology changes to the cluster, go to SMIT:

```
% smitty hacmp

Cluster Configuration
   -> Cluster Topology
         -> Synchronize Cluster Topology
```

This will take you to the “Synchronize Cluster Topology” SMIT window. Accept the defaults by pressing **<Enter>**, and your topology should synchronize successfully. Of course there is always a chance that something will fail, so if it does, go back through this section and make sure that you followed all the steps correctly.

G.10.4.2 Synchronize Resources

To synchronize resource group changes to the cluster, go to SMIT:

```
% smitty hacmp

Cluster Configuration
   -> Cluster Resources
         -> Synchronize Cluster Resources
```

This will take you to the “Synchronize Cluster Resources” SMIT window. Accept the defaults by pressing **<Enter>**, and your resources should synchronize successfully. Of course there is always a chance that something will fail, so if it does, go back through this section and make sure that you followed all the steps correctly.

G.10.4.3 Synchronize HA HPSS Files

Whenever an administrator modifies a file in the /var/hahpss directory on one node, the changes will likely need to be propagated to the other node in the cluster. To do this, use the **/var/hahpss/hpss_sync.ksh** script:

The syntax is:

```
hpss_sync.ksh <remote_node>
```

For example, to sync the current node (**hanode1**) with **hanode2**:

```
% cd /var/hahpss
% ./hpss_sync.ksh hanode2
Updating hanode2 with local changes:
   hpss_aix_error.ksh
   hpss_cluster_notify.ksh
   hpss_environment
   hpss_notify.ksh
   hpss_snapshot.ksh
   hpss_start.ksh
```

```
            hpss_start_list
            hpss_stop.ksh
            hpss_sync.ksh
            hpss_verify.ksh
        Update complete
```

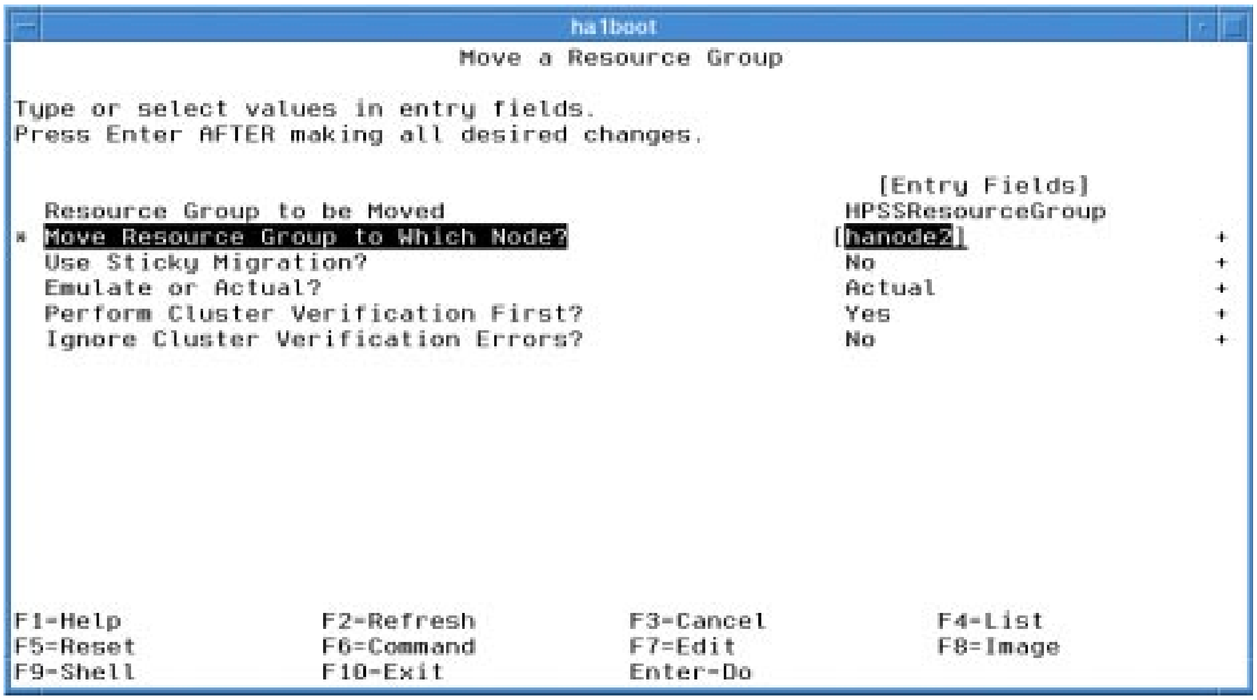
G.10.5 Move a Resource Group

It is often useful to have HACMP move a resource group to another node in the cluster. This will result in a short period of down time as HA HPSS is shutdown on the active node and brought up on the standby node, but it is a convenient way to free up a node for maintenance.

To move a resource group to another node, go to SMIT:

```
% smitty hacmp

Cluster System Management
   -> Cluster Resource Group Management
         -> Move a Resource Group
               [HPSSResourceGroup]
```

Choose the target node for the resource group and hit <Enter>:



```
                                   ha1boot
                           Move a Resource Group

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                         [Entry Fields]
  Resource Group to be Moved                            HPSSResourceGroup
* Move Resource Group to Which Node?                   [hanode2]              +
  Use Sticky Migration?                                 No                    +
  Emulate or Actual?                                    Actual                +
  Perform Cluster Verification First?                   Yes                   +
  Ignore Cluster Verification Errors?                   No                    +

F1=Help             F2=Refresh          F3=Cancel           F4=List
F5=Reset            F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure H-20 Moving a Resource Group

SMIT will then show the status of the resource group move. When the move is completed, all resources will be available on the target node.

Index