
HPSS

User's Guide

**High Performance Storage System
Release 5.1**

December 2003 (Revision 1)

HPSS User's Guide

Copyright (C) 1992-2003 International Business Machines Corporation, The Regents of the University of California, Sandia Corporation, and Lockheed Martin Energy Research Corporation.

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 Lockheed Martin Energy Research Corporation, 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, Lockheed Martin Energy Research Corporation, 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.

Printed in the United States of America.

HPSS Release 5.1

December 2003 (Revision 1)

High Performance Storage System is a registered trademark of International Business Machines Corporation.



Preface

This High Performance Storage System (HPSS) User's Guide provides the necessary information for transferring files using HPSS. It is designed for HPSS users. In particular, the following interfaces are described:

- File Transfer Protocol (FTP) interface
- Parallel FTP (PFTP) interface
- HPSS Tar (HTAR)
- Network File System Version 2 (NFS V2)

Note: It is not the intent of this document to define the standard commands and subcommands provided by standard FTP, and NFS. Only interface extensions provided by HPSS are defined within the HPSS User's Guide.

Refer to the *HPSS Installation Guide*, *HPSS Management Guide*, and *HPSS SSM Guide* for descriptions of the interfaces provided to HPSS administrators. Refer to the *HPSS Programmer's Reference* for programming interfaces provided to the end user. Refer to the *HPSS Error Messages Manual* for a list of all HPSS error and advisory messages which are output by the HPSS software.

The *HPSS User's Guide* is structured as follows:

- **Chapter 1: Overview** - Provides an overview of each type of user interface, a summary of key storage concepts, and recommendations on usage.
- **Chapter 2: File Transfer Protocol (FTP)** - Defines the extensions to the standard FTP interface.
- **Chapter 3: Parallel File Transfer Protocol (PFTP)** - Defines the Parallel FTP (PFTP) interface.
- **Chapter 4: HPSS Tar (HTAR)** - Defines the HTAR interface.
- **Chapter 5: User Utilities** - Defines the set of utilities available to the general user.
- **Appendix A: Acronyms** - Provides a list of acronyms used in this document.
- **Appendix B: References** - Lists documents cited in the text as well as other reference materials.

Typographic and Keying Conventions

This document uses the following typographic conventions:

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

```
% sample command
```

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

```
# This is a comment
```

Italic *Italic* words or characters represent variable values to be supplied.

[] Brackets enclose optional items in syntax and format descriptions.

{ } Braces enclose a list of items to select in syntax and format descriptions.

Table of Contents

Chapter 1 Overview	7
1.1 User Interfaces	7
File Transfer Protocol (FTP)	7
Parallel FTP (PFTP)	7
HPSS Tar (HTAR)	8
Network File System Version 3 (NFS)	8
User Utilities	9
1.2 Storage Concepts	9
Class of Service	9
Storage Class	10
Storage Hierarchy	10
File Family	10
1.3 Interface Usage Considerations	10
1.4 User IDs	11
1.5 DCE User Accounts	12
Chapter 2 File Transfer Protocol (FTP)	13
2.1 Site Commands	13
Specifying a File's Class of Service - setcos	14
Changing a File's Group by ID - chgid	14
Changing a File's Group by Name - chgrp	15
Changing a File's Permissions - chmod	15
Changing a File's Owner by Name - chown	16
Changing a File's Owner by ID - chuid	16
Staging a File - stage	17
Setting the Desire Wait Options (for Migrated Files) - wait	17
Creating a Symbolic Link - symlink	18
Allocating space for files - quote allo64	18
2.2 List Directory Extensions	19
Chapter 3 Parallel File Transfer Protocol (PFTP)	21
3.1 Parallel FTP Client Transfers	24
Parallel FTP Client Configuration	25
3.2 Additional HPSS Commands	27
General Login messages (Examples)	28

Parallel append - pappend	28
Parallel file store - pput	29
Parallel file store - mpput	31
Parallel file retrieval - pget	32
Parallel file retrieval - mpget	33
Local File append - lfappend	34
Local File store - lfput	36
Local file retrieval - lfget	37
Multiple Local file store - mlfput	39
Multiple Local file retrieval - mlfget	40
Specify TCP socket based transfers - psocket	42
Specify transfer stripe width - setpwidth	42
Specify transfer block size - setpblocksize	43
Multinode Enable/Disable - multinode	44
Autoparallel Enable/Disable - autoparallel	45
Get Current Protocol Mode - getprot	46
Get Tuning Parameters - gettuningparms	46
Set the PDATA_ONLY protocol - pdata	48
Override the Non-AutoParallel mode - penable	49
Set the PDATA_AND_MOVER protocol - pmover	49
Set the Socket Buffer Size - setsockbufsize	50
Set the Transfer Buffer Size - setxferbufsize	51

Chapter 4 HPSS Tar (HTAR) 53

4.1 Usage	53
4.2 HTAR Action Flags	54
4.3 HTAR Options	55
4.4 HTAR Examples	58

Chapter 5 User Utilities 59

5.1 Utilities	59
HPSS ACL Editor - hacl	59
List information about HPSS - lshpss	68

Appendix A Acronyms 75

Appendix B References 77

The High Performance Storage System (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 for large storage capacities, and to use network-connected storage devices to transfer data at rates up to multiple gigabytes per second. Listed below are the user interfaces for accessing data from HPSS.

1.1 User Interfaces

1.1.1 File Transfer Protocol (FTP)

HPSS provides an industry-standard FTP user interface. Because 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.

All FTP commands are supported or properly rejected if the HPSS Parallel FTP Daemon does not implement a specific feature. In addition, the ability to specify Class of Service for is provided via the **quote site** or **site** commands. Additional site command options are provided for **chgrp**, **chgid**, **chmod**, **chown**, **chuid**, **stage**, **wait**, and **symlink**. The HPSS FTP Daemon supports access from any RFC-0959 conformant FTP Client. In addition, the **quote allo64** command is supported.

Passive connections are not supported. Also, to avoid potential difficulties, the user should explicitly change the data transfer type to binary. ASCII transfers are very inefficient.

Refer to the HPSS System Administration Guide for information on configuring PFTP.

1.1.2 Parallel FTP (PFTP)

The PFTP supports normal FTP plus extensions. It is built to optimize FTP performance for storing and retrieving files from HPSS by allowing the data to be transferred in parallel to the client. The interface provided to the user has syntax similar to FTP but with some extensions to allow the user to transfer data to and from HPSS across parallel communication interfaces. PFTP supports transfers either via TCP/IP. The FTP client communicates directly with HPSS Movers to transfer data.

The following constraints are imposed by PFTP.

- Pipes are not supported.
- Passive connections are only supported for parallel push type operations using the PDATA_PUSH protocol.
- ASCII transfers are not supported over the parallel interface because ASCII transfers insert characters. This makes it impossible to send the data in parallel. Since extra characters are inserted in the stream, there is no way to resolve data placement. Note that some FTP implementations default to ascii. If this is the case, it will be necessary to specify binary by entering the bin command.
- PFTP client access is supported only from nodes which support the HPSS PFTP client software.

1.1.3 *HPSS Tar (HTAR)*

HTAR is a non-DCE client utility which manipulates HPSS-resident archives by writing files to, or retrieving files from HPSS. Files written to HPSS are in the POSIX 1003.1 "tar" format and may be retrieved from HPSS or read by native tar programs.

The following constraints are imposed by HTAR:

- Reading from or writing to pipes is not supported.
- Appending entries to existing archives is not supported.
- Updating entries in existing archives is not supported.
- Removing entries from existing archives is not supported.
- The archival/extraction of symbolic links is not supported.
- There is no way to specify relative Index file pathnames that are not rooted in the Archive file directory without specifying an absolute path.
- The maximum size of a member in an HTAR archive is 8 GB (2^{33} bytes).

1.1.4 *Network File System Version 3 (NFS)*

The purpose of the NFS V3 server interface is to provide transparent access to HPSS name space object and bitfile data for client systems. Following a mount on the HPSS file system name, the user may access HPSS files using standard function calls and command interfaces.

The code written to implement the NFS V3 Server interface is written to the *Network File System Specification, RFC-1813*, DDN Network Information Center, SRI International, Menlo Park, Ca.

The NFS V3 Server interface and data structures are defined by RFC-1813.

Since there are no extensions or modifications to the NFS user interface, no additional interface information is provided in the remainder of this document.

The following constraints are imposed by the HPSS NFS V3 server:

- All files created using NFS are stored in a single Class of Service. The Class of Service used by NFS is defined by the HPSS administrator.
- NFS transfers are slower than the other HPSS interfaces, and are therefore not recommended for large file accesses.

1.1.5 *User Utilities*

The purpose of the HPSS user utilities is to provide the end user with information such as Access Control List (ACL) definitions and Class of Service definitions. In addition, the ability for a user to change his ACL definitions is provided.

The user utilities consist of these commands:

- **hacl** - edit HPSS Access Control Lists
- **lshpss** - list information for HPSS (Class of Service list, hierarchy list, storage class list, physical volumes, devices and drives, servers, Movers, and other metadata)

1.2 *Storage Concepts*

This section defines key HPSS storage concepts which have a significant impact on the usability of HPSS. Configuration of the HPSS storage objects and policies is the responsibility of your HPSS administrator.

1.2.1 *Class of Service*

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 characteristics such as minimum and maximum file size, transfer rate, access frequency, latency, and valid read or write operations. A file resides in a particular COS and the class is selected when the file is created. Underlying a COS is a storage hierarchy that describes how data for files in that class are to be stored in HPSS.

For the FTP and PFTP interfaces, the COS ID may be explicitly specified by using the “**site setcos**” command. If not specified, a default COS is used. For NFS, all files are created in the same COS. This COS is defined by your system administrator. Otherwise, the size of the file is used as hints for COS selection. Contact your HPSS administrator to determine the COSs which have been defined. The “**lshpss -cos**” command may also be used to list the defined COSs. Refer to Chapter 5 for information on the “**lshpss**” command.

Also, PFTP provides a feature to automatically store the local file size in the minimum and maximum file size fields of the COS. This feature is also provided for FTP clients which support the ALLO command. This allows the COS selection to be made according to file size. The HPSS administrator should ensure that COS definitions contain proper minimum and maximum file sizes in order for PFTP (FTP clients which support ALLO) to optimize storage utilization when transferring files to HPSS. Note: If the COS ID is explicitly set by using the “**site setcos**” command, that COS will be used regardless of file size.

A COS is implemented by a Storage Hierarchy of one to many Storage Classes. Storage Hierarchies and Storage Classes are not directly visible to the user, but are described below since they map to COS..

1.2.2 *Storage Class*

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

1.2.3 *Storage Hierarchy*

An HPSS storage hierarchy consists of multiple levels of storage with each level representing a different storage media (i.e., a storage class). Files are moved up and down the storage hierarchy via stage and migrate operations, respectively, based upon storage policy, usage patterns, storage availability, and user request. For example, a storage hierarchy might consist of a fast disk, followed by a fast data transfer and medium storage capacity robot tape system, which in turn is followed by a large data storage capacity, but relatively slow data transfer tape robot system. Files are placed on a particular level in the hierarchy depending on the migration policy and staging operations. Multiple copies of a file may also be specified in the migration policy. 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.

1.2.4 *File Family*

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 for files by associating a fileset with a family, and creating the files in that fileset. When a file is migrated from disk to tape, it is migrated to a tape virtual volume assigned to the family associated with the file. If no family is associated with the file, the file is migrated to the next available tape not associated with a family (actually to a tape associated with family zero). If no tape virtual volume is associated with the family, a blank tape is reassigned from family zero to the file's family. The family affiliation is preserved when tapes are repacked. Configuring file families is a System Administrator function.

1.3 *Interface Usage Considerations*

Guidance on when to use a particular HPSS interface is provided below. In general, PFTP provides the best data transfer performance. NFS is the slowest interface, and should not be the interface of choice for large HPSS data transfers.

Conditions in which a user might elect to use FTP are:

- Utilizes standard FTP interface - Users and applications familiar with FTP can access HPSS with the standard command set.
- Supports files greater than 2 gigabytes - FTP supports file sizes up to 2**64.
- Supports any FTP client platforms - FTP commands may be issued from any vendor nodes with an FTP interface. No specialized code is required.

Conditions in which a user might elect to use PFTP are:

- Provides faster file transfers than FTP - PFTP is a better performer than FTP since it provides the capability to stripe data across multiple client data ports.
- Supports files greater than 2 gigabytes - PFTP supports file sizes up to 2**64.
- Supports partial file transfer - PFTP provides options on the pget and pput commands to perform partial file transfers. This would be beneficial to users who want to extract pieces of large files.

Conditions in which a user might elect to use HTAR are:

- Allows for several small files to be viewed by HPSS as one large file. This is very desirable to users who store lots of small files since HPSS does not perform well with lots of small files.
- Since HTAR creates an index file when an archive file is created, files can be extracted quickly without having to rumage through the entire archive file since the index file stores the locations of each entry and is much smaller than the archive file, therefore much quicker to search.
- Since all of the archived file information is stored in the index file it is possible to view the contents of an archive file without having to stage the entire archive file from tape storage to the disk cache.

Conditions in which a user might elect to use NFS are:

- Provides standard system access - Files may be accessed and managed through standard system mechanisms without calling a special library or program to translate commands
- Eliminates multiple file instances - the need to maintain multiple instances of a file can be eliminated since files remain on the NFS server.
- Accesses limited to smaller files
- Supports any NFS client platforms - NFS access is supported from any client nodes with an NFS V2 or V3 interface. No specialized code is required.

1.4 User IDs

After the HPSS system is configured, the necessary accounts must be created for HPSS users. Contact your HPSS administrator to add an account.

For FTP / PFTP access, an FTP account must be created. The administrator can use the “**hpssuser -add <user> -ftp**” command to add a new FTP user.

For NFS access, a DCE user account must be created. The administrator can use the “**hpssuser -add <user> - dce**” command to add a new DCE account.

Users calling the utilities described in this document must be logged into DCE. As noted above, the administrator can use the “**hpssuser -add <user> -dce**” command to add a new DCE account.

1.5 DCE User Accounts

As mentioned in the previous section, the user utilities require the user be logged into DCE.

The following command syntax is used to issue a DCE login:

dce_login [*principal_name*] [*password*]

When this command is entered, the principal's identity is validated, and the network credentials are obtained. If *principal name* or *password* are not supplied, **dce_login** will prompt for them.

When the principal's DCE login context is no longer required, the following command may be used to destroy the login context and associated credentials:

% kdestroy

Other DCE commands which might be of interest to the user are:

- **klist** - list the primary principal and tickets held in the DCE credentials cache
- **kinit** - Refresh a DCE credentials cache

File Transfer Protocol (FTP)

This chapter specifies the HPSS FTP interface. FTP is supported from any FTP client platform.

HPSS supports the FTP command set for transferring files to and from HPSS. To use FTP, use the following syntax:

```
ftp <node_name> [<port_number>]
```

where,

node_name is the node name of the node where the HPSS FTP Daemon process resides

port_number is the port number for HPSS, as set up in `/etc/services`

At this point, any standard FTP command may be entered. Note: If the message "Load thread state failed" is received, contact your HPSS administrator. This message generally implies that either HPSS is not correctly configured, or some HPSS components may not be executing:

2.1 Site Commands

HPSS also supports the site commands listed below (e.g., "`site setcos 300`" or "`quote site setcos 300`").

Note: On some platforms, it may be necessary to specify **quote site** instead of **site**.

- **setcos**
- **chgid**
- **chgrp**
- **chmod**
- **chown** (valid only for "root" account)
- **chuid**

- **stage**
- **wait**
- **symlink**

2.1.1 *Specifying a File's Class of Service - setcos*

setcos is used to specify a class of service and has the following format:

```
quote site setcos <cos_id>
```

where,

cos_id is the Class of Service identifier (used when creating a new HPSS file during a put operation.)

Class of Service is used as a means for specifying the amount of parallelism or stripe width for a file. See your HPSS system administrator for the Class of Service identifiers defined for your site. If a Class of Service is not specified, a default is used.

In the example below, the following commands might be entered to put a large file to HPSS with a Class of Service identifier of 4. In this example, 4 might designate 4-way striping to 3490 tape.

```
ftp> quote site setcos 4
```

2.1.2 *Changing a File's Group by ID - chgid*

chgid is used to change the group ID of a file and has the following format:

```
quote site chgid <gid> <file>
```

where,

gid is the new group ID of the file

file is the name of the file.

The user must belong to the specified group and be the owner of the file, or be the root user.

Example: The following may be entered to change the group ID of myfile to group ID 210.

```
ftp> quote site chgid 210 myfile
```

2.1.3 Changing a File's Group by Name - *chgrp*

chgrp is used to change the group name of a file and has the following format:

```
quote site chgrp <group> <file>
```

where,

group is the new group name of the file, and *file* is the name of the file.

The user must belong to the specified group and be the owner of the file, or be the root user.

Example: The following may be entered to change the group of *myfile* to group *mygroup*.

```
ftp> quote site chgrp mygroup myfile
```

2.1.4 Changing a File's Permissions - *chmod*

chmod is used to change the mode of a file and has the following format:

```
quote site chmod <mode> <file>
```

where,

mode is the new octal mode number of the file

file is the name of the file.

Mode is constructed from the OR of the following modes:

0400	read by owner
0200	write by owner
0100	execute (search in a directory) by owner
0040	read by group
0020	write by group
0010	execute (search in a directory) by group
0004	read by others
0002	write by others
0001	execute (search in a directory) by others

Note: The following mode values are not supported:

4000	set user ID on execution
2000	set group ID on execution
1000	sticky bit

Only the owner of the file or root user can change its mode.

Example: The following may be entered to change the mode of myfile to read, write by owner and group.

```
ftp> quote site chmod 0660 myfile
```

2.1.5 Changing a File's Owner by Name - *chown*

chown is used to change the owner of a file and has the following format:

```
quote site chown <owner> <file>
```

where,

owner is the new owner of the file

file is the name of the file.

Only the root user can change the owner of a file.

Example: The following may be entered to change the owner of /home/smith/myfile to jones.

```
ftp> quote site chown jones /home/smith/myfile
```

2.1.6 Changing a File's Owner by ID - *chuid*

chuid is used to change the uid of a file and has the following format:

```
quote site chuid <uid> <file>
```

where,

uid is the new uid of the owner of the file

file is the name of the file.

Only the root user can change the uid of a file.

Example: The following may be entered to change the uid of /home/smith/myfile to 201.

```
ftp> quote site chuid 201 /home/smith/myfile
```

2.1.7 Staging a File - stage

stage is used to initiate a stage of a migrated file (e.g. from tape to disk). The user can initiate the stage and then return at a later time to initiate the file transfer using the FTP **get** or PFTP **pget** commands:

```
quote site stage <file>
```

where,

file is the name of the file.

Example: The following may be entered to stage file /home/smith/myfile.

```
ftp> quote site stage /home/smith/myfile
```

2.1.8 Setting the Desired Wait Options (for Migrated Files) - wait

wait is used to notify the HPSS PFTP Daemon :

```
quote site wait <option>
```

where,

option is one of the following values:

-1 or inf(inite) - wait forever for the file to be staged. Do not return from the **get** / **pget** command to complete until the file has been transferred or a transfer error has occurred.

0 - do not wait for the file to be staged. If the file has been migrated, return the appropriate message and initiate the stage. The user will return later to reissue the **get** / **pget** command.

n (where n is an integer) - wait the specified period (in seconds) for the file requested by a **get** / **pget** command to complete. Either transfer the file if the file is staged within the specified period or return a reply to notify the user to try again later.

Example: The following may be entered to wait for files to be staged.

```
ftp> quote site wait -1
```

The following table describes the behaviour the customer should expect from FTP when issuing the stage/wait commands. Note: ONLY Classes of service utilizing the "Stage on Background" will exhibit predictable results.

Stage/Wait Behaviour

Wait Time	File Condition	Command	Behaviour /Message
No Wait	Archived	quote site stage xyz	"File xyz is being retrieved from archive."

No Wait	Not Archived	quote site stage xyz	“File xyz is currently ready for other processing.”
Wait ###	Archived	quote site stage xyz	Wait for period then receive message: “File xyz is currently ready for other processing.” or “File xyz is currently ready for other processing.” if the file is staged in the time frame allowed
Wait ###	Not Archived	quote site stage xyz	“File xyz is currently ready for other processing.”
No Wait	Archived	get xyz	“File xyz is being retrieved from archive.”
No Wait	Not Archived	get xyz	Transfers Data as expected.
Wait ###	Archived	get xyz	Wait for period then receive message: “File xyz is being retrieved from archive.” or transfers data as expected if file is staged in the time allowed.
Wait ###	Not Archived	get xyz	Transfers file as expected.

2.1.9 Creating a Symbolic Link - symlink

symlink is used to create a symbolic link.

```
quote site symlink <path/file> <link>
```

where,

path/file refers to the destination

link refers to the local filename.

Example: The following may be entered to create a link names sys_passwd in the local directory pointing to /etc/passwd.

```
ftp> quote site symlink /etc/passwd sys_passwd
```

A **dir** command will show sys_passwd -> /etc/passwd.

2.1.10 Allocating space for files - quote allo64

The quote allo64 command is used to specify the size of a file for space allocation.

```
quote allo64 <size>
```

where,

size is a string representing the size of the file. The size may be a decimal number less than 2⁶⁴ or may be in the form 1MB (1048576). No spaces are allowed between the decimal number and the magnitude representation string. Accepted magnitude representation strings are:

KB (kilobyte = 1024),

MB (megabyte = 1048576),

GB (gigabyte = 1073741824),

TB (terabyte = 1099511627776),

PB (petabyte = 1125899906842624).

The magnitude representation string is case independent. The decimal component may contain up to two decimal points of precision. NOTE: 1005.03 will truncate to 1005 if no magnitude representation string is specified. Similar truncations will occur for excess precision specifications.

This command provides a 64-bit extension to the standard **quote allo size** command. NOTE: the **quote allo size** command only accepts decimal values for size. Both these commands are helpful for providing hints for non-parallel "put" commands.

Example: The following may be entered to specify the file size of 8 gigabytes.

```
ftp> quote allo64 8GB
```

2.2 List Directory Extensions

FTP supports the **ls** command to list the contents of a directory. Standard options supported are: **ls**, **ls -l**, **ls -a**, and **ls -F**. In addition to the standard **ls** options generally provided, HPSS also provides a **-lh** option. If **-lh** is specified, then a long directory listing is generated. However, in place of the owner field (field #3) and group field (field #4) listed for the **-l** option, the Class of Service identifier and Account Code are listed.

Example:

```
ftp> ls -lh
-rw-rw---- 1 1          198      157286400 May 13 1996 TEST
-rw-r--r-- 1 1          160           32768 May 16 1996 prod1
-rw-r--r-- 1 1          160           32768 May 16 1996 prod10
-rw-r--r-- 1 1          160           32768 May 16 1996 prod11
-rw-r--r-- 1 1          160           32768 May 16 1996 prod12
-rw-r--r-- 1 1          160           32768 May 16 1996 prod13
-rw-r--r-- 1 1          160           32768 May 16 1996 prod14
-rw-r--r-- 1 1          160           32768 May 16 1996 prod15
-rw-r--r-- 1 1          160           32768 May 16 1996 prod151
-rw-r--r-- 1 1          160           32768 May 16 1996 prod152
-rw-r--r-- 1 1          160           32768 May 16 1996 prod153
-rw-r--r-- 1 1          160           32768 May 16 1996 prod154
-rw-r--r-- 1 1          160           32768 May 16 1996 prod155
-rw-r--r-- 1 1          160           32768 May 16 1996 prod156
-rw-r--r-- 1 1          160           32768 May 16 1996 prod157
-rw-r--r-- 1 1          160           32768 May 16 1996 prod158
-rw-r--r-- 1 1          160           32768 May 16 1996 prod159
```


Parallel File Transfer Protocol (PFTP)

This chapter specifies the HPSS PFTP interface. In order to use PFTP, the PFTP client code must be compiled and supported on the client platform.

PFTP supports the FTP command set plus some additional commands (refer to the next subsection). To use PFTP, the user enters one of the following commands:

```
pftp_client [-bStringSize] [-c] [-d] [-e] [-g] [-h] [-i] [-m] [-n]
[-p] [-t] [-v] [-w###] [-BStringSize] [-C###] [-Rstring]
[-SsizeString] [Host [Port]]
```

```
krb5_gss_pftp_client [-bStringSize] [-c] [-d] [-e] [-g] [-h] [-i] [-m]
[-n] [-p] [-t] [-v] [-w###] [-BStringSize] [-C###] [-Rstring]
[-SsizeString] [Host [Port]]
```

where,

Table 3-1 PFTP Client Command-Line Options

Option	Description
-b	Sets the PDATA protocol Blocksize. StringSize is the size specification in the format: Digit(s)Magnitude; e.g. 1MB.
-c	Sets “Child” mode. This provides the ability to “emulate” a tty and interactive mode when executing the client in a “batch” mode.
-d	The standard FTP debug specification.
-e	Sets “Echo” mode. When running the client in batch mode, this causes the client to echo each command into the output file providing a helpful record of commands interleaved with the return messages.
-g	Disables the expansion of metacharacters in file names. Interpreting metacharacters can be referred to as expanding (sometimes called globbing) a file name. See the glob subcommand.
-h	Specifie to use the original HPSS protocol, PDATA_AND_MOVER, regardless of the default specified in the HPSS.conf file
-i	Turns off interactive prompting during multiple file transfers. See the prompt, mget, mput, and mdelete subcommands for descriptions of prompting during multiple file transfers.
-k	Kerberos ONLY option to specify an alternate Kerberos Realm for the PFTP Daemon.
-m	This argument will enable multinode processing. By default, multinode processing is disabled. Multinode will be ignored if NO multinode specification for this client / daemon pair is specified in the HPSS.conf file.
-n	Prevents an automatic login on the initial connection. Otherwise, the ftp command searches for a \$HOME/.netrc entry that describes the login and initialization process for the remote host. See the user subcommand.
-p	Specifies to use the HPSS protocol (PDATA_ONLY) for parallel transfers regardless of the default in the HPSS.conf file.
-t	The standard FTP trace specification.
-v	Displays all the responses from the remote server and provides data transfer statistics. This display mode is the default when the output of the ftp command is to a terminal, such as the console or a display.
-w	This argument will set the pwidth. The pwidth value must be specified immediately following this argument.
-B	Sets the Parallel Blocksize for parallel transfers. StringSize is the size specification in the format: Digit(s)Magnitude; e.g. 1MB.
-C	Sets the default Class of Service (COS) for the session. The argument is a valid string representation of a decimal COS. COS names are NOT accepted.

Table 3-1 PFTP Client Command-Line Options

Option	Description
-P	Specifies to use the PDATA_PUSH protocol. This will be overridden if another protocol is specified in the .netrc file or if an explicit specification of another protocol is made by the user.
-R	Used to specify the valid ports for parallel transfers. This is useful in instances where network filters are invoked which provide port ranges for TCP traffic. The syntax is “start_range-end_range”. These values are translated into the environment variable “HPSS_PFTPC_PORT_RANGE=ncacn_ip_tcp[start_range-end_range]” environment variable which is parsed by the client as necessary.
-S	Sets the maximum open / close socket size for HPSS parallel transfers. An artificial maximum of 250GB (subject to change) is compiled in. Results may be smaller than the specified value based on a number of external HPSS constraints. The default is 1.5GB. StringSize is the size specification in the format: Digit(s)Magnitude; e.g., 200GB.
Host	The node where the HPSS FTP Daemon process resides.
Port	The port number for the HPSS FTP Daemon, as set in /etc/services.

The local administrator may opt to define a **pftp** program link that points to **pftp_client**.

An additional variant of the **pftp_client**, **krb5_gss_pftp_client** may be built by the customer site. Contact your site representative for details. These clients utilize the MIT Kerberos GSS facilities for authentication and reply processing. The GSS-based clients are used to provide credential authentication facilities (password-less authentication) between the client and the HPSS GSS Parallel FTP Daemon using either Kerberos or DCE credentials for authentication. Since the Generic Security Service (GSS) versions of the Parallel FTP Client only relate to the authentication process, these clients should behave identical to the non-GSS versions after authentication. MIT Kerberos is available from MIT and will NOT be supplied by the HPSS project. NOTE: The HPSS (GSS) Parallel FTP Client and Daemon are incompatible with the Kerberos-based FTP features provided by IBM with AIX 4.x. The HPSS (GSS) Parallel FTP Client and Daemon **are** compatible with the MIT FTP processes.

The GSS version of the Parallel FTP Client requires MIT Kerberos and/or compatible Client software (headers / libraries). Neither IBM nor the HPSS development team are obligated to continue the GSS PFTP in the future.

As a courtesy to HPSS customers, the Parallel FTP Client code is available for compilation at customer sites upon request. The Parallel FTP Client code will be provided as a `clients_port.tar.Z` file (tarred and compressed) containing all components required to build the applications. Hardware/Software dependencies are the individual HPSS customers responsibility. This explicitly denies any support requirement on IBM or the HPSS Development/Support personnel for any modifications made by the customer. No DCE software is required to build the HPSS Parallel FTP Client.

The HPSS PFTP Client has been successfully compiled on: Cray UNICOS, Hewlett-Packard HPUX (32 / 64 Bit), Silicon Graphics IRIX (32 / 64 Bit), Sun Solaris (32 / 64 Bit), Intel Paragon OSF (discontinued), Intel Teraflop OSF, Linux Intel (32 / 64 Bit), Compaq Alpha, and IBM AIX 4.x (32 / 64 Bit). Ports to other hardware/software components are the responsibility of the remote site. These sites will be asked to share their ports with the HPSS development team (and other HPSS facilities); however, neither IBM nor the HPSS Development Team accepts any obligation to incorporate any hardware/software ports into the distribution source. No site specific features (local mods) added to the Parallel FTP client by customer sites will be incorporated into the PFTP client without the modification of the HPSS license.

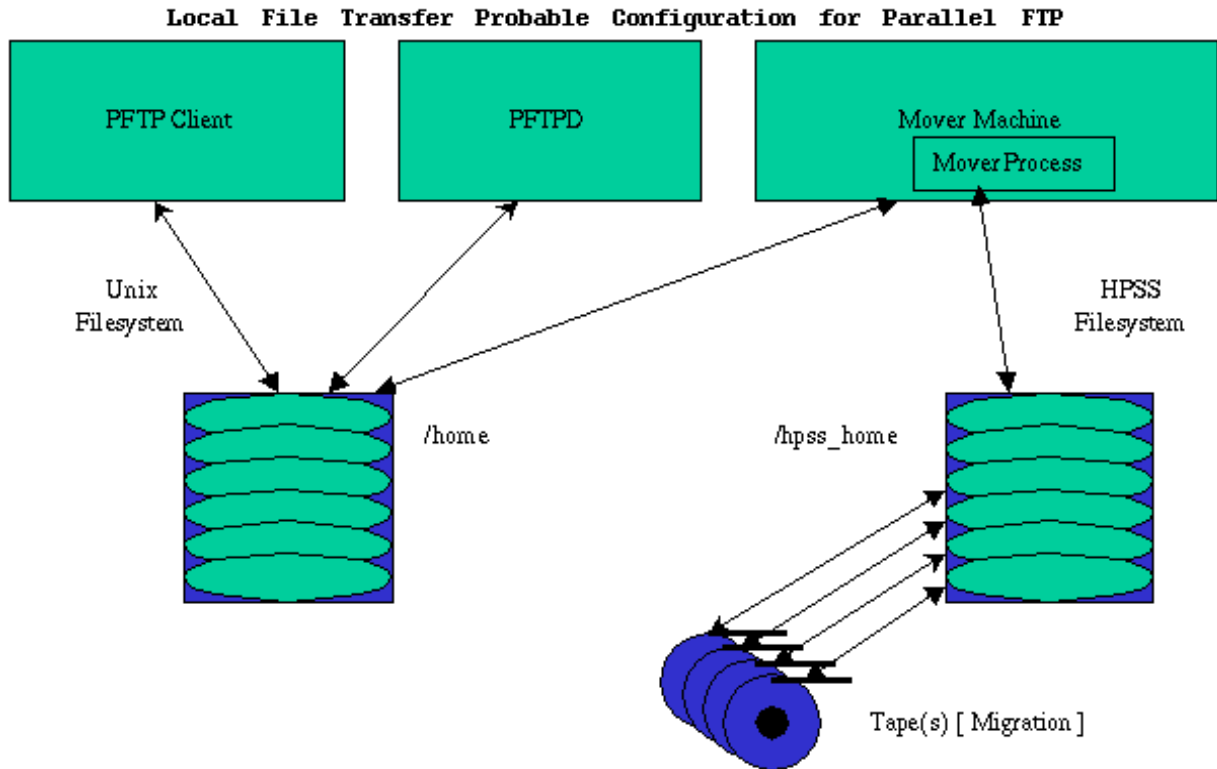
The GSS versions of the Parallel FTP Client, **krb5_gss_pftp_client**, require MIT Kerberos. Neither IBM nor the HPSS development team declare the Kerberos code suitable for any specific purpose nor are they obligated to repair or support customers using this code.

The GSS HPSS Parallel FTP Daemon, **hpss_pftp_amgr** and **auth_krb5gss** executables, are available for IBM AIX 4.x and Solaris 8.

Note: If the message "Load thread state failed" is received, contact your HPSS administrator. This message generally implies that either HPSS is not correctly configured, or some HPSS components may not be executing.

3.1 Parallel FTP Client Transfers

Parallel transfers involve the creation of child processes to transfer the data between the source and the destination. This process may be either local spawned PFTP client children, remotely initiated PFTP "children," or the combination of both. When the `pwidth` value is set and a valid multinode configuration file does not exist or multinode has not been activated, the PFTP client will provide parallel data paths to



the Movers by spawning multiple processes on the client node using one or more network interfaces (NICs).

The multinode option supports spawning the client processes across multiple machines / nodes and/or multiple interfaces on the remote machines / nodes. This multinode option may be beneficial on processors which support shared file systems, such as GPFS on the IBM SP. Note: if multinode is used in a non-shared file system, the multinode file transfer to the client will be spread across multiple, separate files, which is not the desired behaviour. The client nodes which participate in a multinode transfer are selected from the HPSS.conf configuration file which contains entries with control, and optionally, data interface names or addresses. The number of nodes selected from the configuration file is based on the pwidth value. The starting node is selected using an offset of which is maintained by the PFTP client.

3.1.1 Parallel FTP Client Configuration

The PFTP client requires configuration in the HPSS.conf file to provide optimal performance characteristics.

3.1.1.1 HPSS.conf File

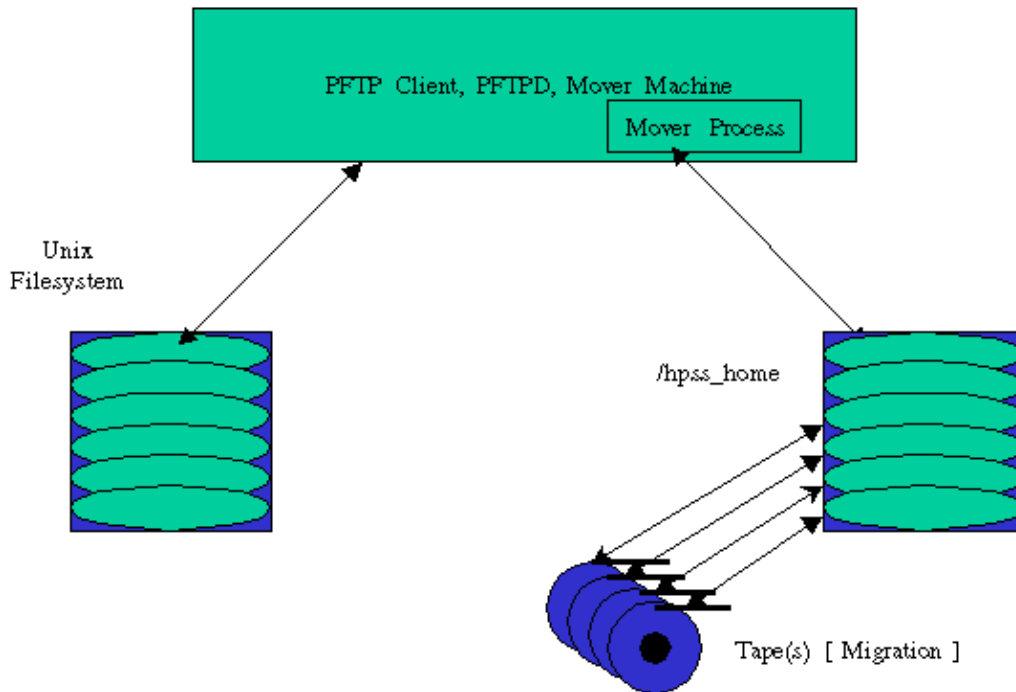
This configuration file is used to specify performance optimization parameters for the PFTP components, the HPSS Movers, and potentially Site specific applications. For details of the implementation of the HPSS.conf file, contact your local HPSS administrator.

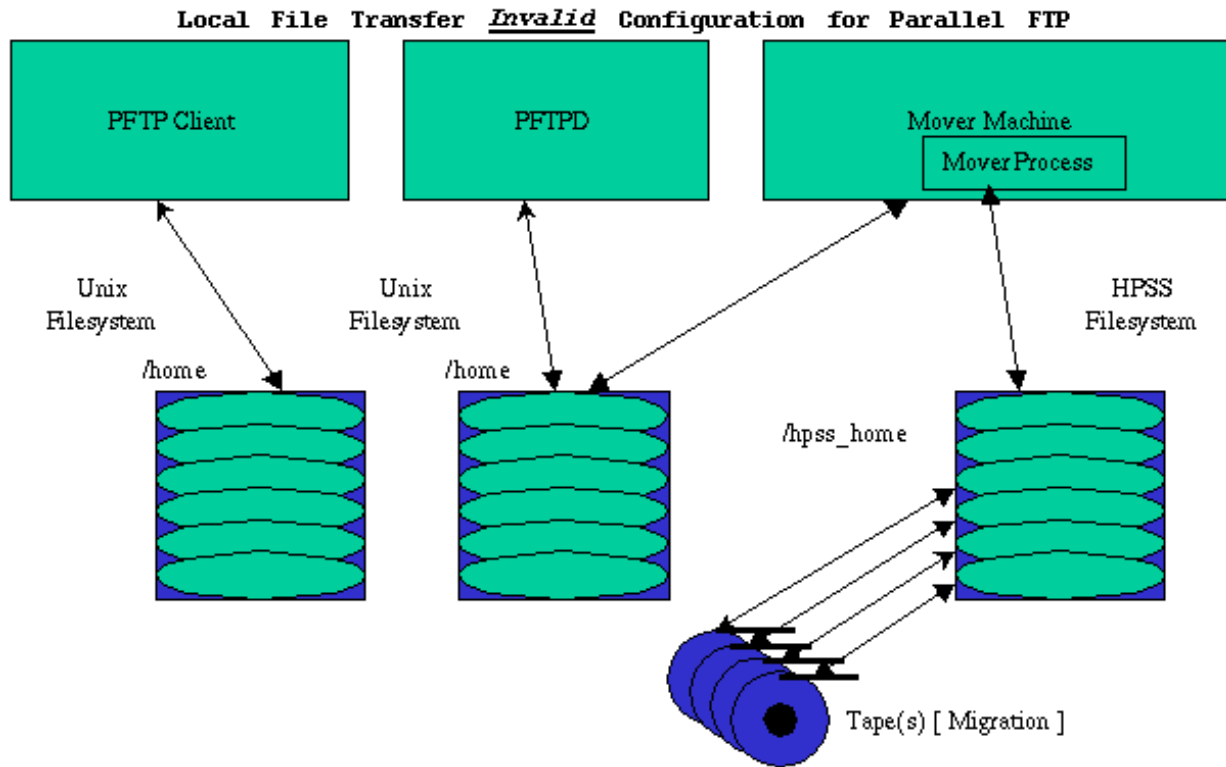
3.1.1.2 Local File Functions

The Local File Functions represent performance enhancements using the HPSS parallel protocols where both the HPSS file and the Unix source/destination file are "globally available" to the mover(s) and the PFTP client processes (e.g., GPFS filesystems.) The local file path must be specified in the file: /var/hpss/etc/hpss_mvr_localfilepath.conf. The specified path specified must exist for each PFTP Client/Mover machine.

Configuration Criteria:

Local File Transfer Optimal Configuration for Parallel PTF





3.2 Additional HPSS Commands

All FTP extensions described in Chapter 2 are supported by PFTP. In addition, the following commands (abbreviations) are supported by PFTP:

- pappend (papp)
- pput, mpput (ppu, mpp)
- pget, mpget (pge, mpg)
- lfappend (lfa)
- lfput, mlftp (lfp, mlfp)
- lfget, mlftp (lfg, mlfg)
- psocket (psock)
- setpwidth (setpw)
- setpblocksize (setpb)
- multimode (multi)

Chapter 3 Parallel File Transfer Protocol (PFTP)

- autoparallel (autop)
- getprot (getp)
- gettuningparms (gettun)
- pdata (pdat)
- penable (pen)
- pmover (pmov)
- setsockbufsize (sets)
- setxferbufsize (setx)



*Pipes are NOT supported for the pget and pput commands even if the HPSS.conf specifies a valid pipe directory. To use pipes it is mandatory that you specify the **autop** command to disable automatic substitution of parallel commands and then explicitly use the **get** or **put** commands.*

3.2.1 General Login messages (Examples)

```
Connected to water.clearlake.ibm.com.
220-#
220-#   HPSS Parallel FTP Daemon on water
220-#
220-
220 water FTP server (HPSS 4.3 PFTPD V1.1.4 Tue May 29 14:55:58 CDT 2001) ready.
Name (water:whrahe):
331 Password required for /.../water_cell.clearlake.ibm.com/whrahe.
Password:
230 User /.../water_cell.clearlake.ibm.com/whrahe logged in.

Remote system type is UNIX.
Using binary mode to transfer files.
**** NOTE: Server supports Parallel Features      ****
****           Auto-Parallel Substitution Enabled. ****
**** NOTE: Protocol set to PDATA_AND_MOVER      ****
Multinode is Disabled.

ftp>
```

3.2.2 Parallel append - pappend

Synopsis

```
pappend local_file [remote_file]
```

Description

The **pappend** command transfers a file from the local machine to HPSS. The transfer starts at the end of the remote file and continues until the entire file is moved or until an error occurs.

Parameters

local_file - Identification of the file to transfer on the local machine.

remote_file - Optional file name to the remote file. If not supplied then the remote (HPSS) file name defaults to be the same as the local file name.

Return strings

Output shows the amount of data transferred and any error conditions.

Error conditions

Connection Failures - data transfer connection malfunction.

Network Failures - data transfer malfunction.

Allocation Failures - no space on remote machine for file.

Error codes may also be returned from HPSS. The most common error codes are:

- -5 - an I/O error occurred.
- -28 - no space remaining in the associated storage class.

See also

RFC-0959.

Notes

none.

Examples

1. Append local file testfile to the same file name in the user's HPSS home directory.

```
ftp> pappend testfile
```

2. Append local file testfile to HPSS file prod1 in the current working directory.

```
ftp> pappend testfile prod1
```

3.2.3 Parallel file store - pput

Synopsis

```
pput [-l local_offset] [-r remote_offset] [-s size] local_file [remote_file]
```

Description

The **pput** command transfers a file from the local machine to HPSS. If offsets and size of transfer are not specified, the transfer starts at the beginning of the local file and continues until the entire file is moved or until an error occurs. However, flexibility is provided to perform partial file transfers by specifying local file offset, remote file offset, and size of transfer. The *local_offset*, *remote_offset*, and *size* may be specified using a decimal and magnitude representation string.

The normal **pput** command functions just like the standard ftp **put** command and transfers an entire file.

Parameters

-l *local_offset* - Optional byte offset into the local file where the transfer is to begin.

-r *remote_offset* - Optional byte offset into the remote file where the data is to be placed.

-s *size* - Optional byte size of the amount of data to transfer.

local_file - Identification of the file to transfer on the local machine.

remote_file - Optional file name to the remote (HPSS) file. If not supplied then the remote file name defaults to be the same as the local file name.

Return strings

Output shows amount of data transferred and any error conditions.

Error conditions

Connection Failures - data transfer connection malfunction.

Network Failures - data transfer malfunction.

Allocation Failures - no space on remote machine for file.

Error codes may also be returned from HPSS. The most common error codes are:

- -5 - an I/O error occurred.
- -28 - no space remaining in the associated storage class.

See also

RFC-0959.

Notes

none.

Examples

1. Transfer local file testfile to the user's HPSS home directory.

```
ftp> pput testfile
```

2. Transfer local file testfile to HPSS file prod1 in the current working directory.

```
ftp> pput testfile prod1
```

3. Transfer 1MB from offset 1MB of local file testfile to offset 0 of HPSS file /home/bob/prod1.

```
ftp> pput -l 1048576 -r 0 -s 1048576 testfile /home/bob/prod1.
```

4. Transfer all local files which begin with "test" to the user's HPSS home directory using a pipe and tar (bundling).

```
ftp> pput " | tar cf - ./test*" my_test.tar
```

Parallel Pipes are NOT supported! Specify the "autop" command to disable automatic parallel command substitution and use the "put" command!



3.2.4 Parallel file store - mpput

Synopsis

```
mpput local_files
```

Description

The **mpput** command expands the files specified in the *local_files* parameter at the local host and copies the indicated files to HPSS. The **mpput** command functions just like the standard ftp **mput** command.

Parameters

local_files - Identification of the files to transfer on the local machine.

Return strings

Output shows the amount of data transferred and any error conditions.

Error conditions

Connection Failures - data transfer connection malfunction.

Network Failures - data transfer malfunction.

Allocation Failures - no space on remote machine for file.

Error codes may also be returned from HPSS. The most common error codes are:

- -5 - an I/O error occurred.
- -28 - no space remaining in the associated storage class.

See also

RFC-0959.

Notes

none.

Examples

1. Transfer all local files in the current directory to the user's HPSS home directory.

```
ftp> mpput *
```

2. Transfer all local files which begin with test in directory /usr/bob to the user's HPSS home directory.

```
ftp> lcd /usr/bob
```

```
ftp> mpput test*
```

3.2.5 Parallel file retrieval - pget

Synopsis

```
pget [-r remote_offset] [-l local_offset] [-s size] remote_file [local_file]
```

Description

The **pget** command transfers a file to the local machine from HPSS. If offsets and size of transfer are not specified, the transfer starts at the beginning of the remote file and continues until the entire file is moved or until an error occurs. However, flexibility is provided to perform partial file transfers by specifying remote file offset, local file offset, and size of transfer. The *local_offset*, *remote_offset*, and *size* may be specified using a decimal and magnitude representation string. See Section 2.1.10 for use of this notation.

The standard **pget** command transfers entire files similar to the standard **ftp** get command.

Parameters

-r remote_offset - Optional byte offset where transfer is to begin in the remote file.

-l local_offset - Optional parameter where the data is transferred in the local file.

-s size - Optional number of bytes to transfer.

remote_file - Identification of the file to transfer from the remote (HPSS) host.

local_file - Optional file name to the local file. If not supplied then the local file name defaults to be the same as the remote file name.

Return strings

Output shows the amount of data transferred and any error conditions.

Error conditions

Connection Failures - data transfer connection malfunction.

Network Failures - data transfer malfunction.

Allocation Failures - no space on local machine for file.

Error codes may also be returned from HPSS. The most common error code is:

- -5 - an I/O error occurred.

See also

RFC-0959.

Notes

none.

Examples

1. Transfer HPSS file /home/bob/prod1 to the user's local directory.

```
ftp> pget /home/bob/prod1
```

2. Transfer HPSS file prod1 in the current working directory to local file testfile1.

```
ftp> pget prod1 testfile1
```

3. Transfer 1MB from offset 0 of HPSS file /home/bob/prod1 to offset 1048576 of local file testfile.

```
ftp> pget -r 0 -l 1048576 -s 1048576 /home/bob/testfile1 testfile
```

4. Transfer and untar a tar file into the user's current working directory using a pipe and tar (unbundling).

```
ftp> pget my_test.tar " | tar xf -"
```

Parallel Pipes are NOT supported! Specify the "autop" command to disable automatic parallel command substitution nad use the "get" command!



3.2.6 Parallel file retrieval - mpget

Synopsis

```
mpget remote_files
```

Description

Chapter 3 Parallel File Transfer Protocol (PFTP)

The **mpget** command expands the *remote_files* parameter at the remote (HPSS) host and copies the indicated HPSS files to the current directory on the local host. The **mpget** command functions just like the standard ftp **mget** command.

Parameters

remote_files - Identification of the files to transfer from the remote (HPSS) host.

Return strings

Output shows the amount of data transferred and any error conditions.

Error conditions

Connection Failures - data transfer connection malfunction.

Network Failures - data transfer malfunction.

Allocation Failures - no space on remote machine for file.

Error codes may also be returned from HPSS. The most common error code is:

- -5 - an I/O error occurred.

See also

RFC-0959.

Notes

none.

Examples

1. Transfer all files in HPSS directory /home/bob to the user's local directory.

```
ftp> cd /home/bob
ftp> mpget *
```

2. Transfer all HPSS files which begin with test in directory /home/bob to the user's local directory.

```
ftp> cd /home/bob
ftp> mpget test*
```

3.2.7 Local File append - *lfappend*

Synopsis

```
lfappend local_file [remote_file]
```

Description

The **lfappend** is a performance optimized Parallel FTP Client command used to append a "globally available" file into HPSS using the "parallel" protocols. IF the input file is not available to the mover(s) machine (NOT "globally available") the transfer will fail because the mover(s) will NOT be able to locate the desired file. The difference between lfappend and pappend is that the mover(s) involved in the transfer, will not use the network to move the data providing improved performance. The mover(s) machine MUST be correctly configured. The file: /var/hpss/etc/ hpss_mvr_localfilepath.conf MUST exist on each "local file" aware mover machine and MUST contain entries specifying which directories are eligible for "local file" transport.

The **lfappend** command transfers a file from the local machine to HPSS. The transfer starts at the end of the remote file and continues until the entire file is moved or until an error occurs.

Parameters

local_file - Identification of the "globally available" file to transfer.

remote_file - Optional file name to the remote file. If not supplied then the remote (HPSS) file name defaults to be the same as the "globally available" file name.

Return strings

Output shows the amount of data transferred and any error conditions.

Error conditions

Connection Failures: data transfer connection malfunction.

Network Failures: data transfer malfunction.

Allocation Failures: no space on remote machine for file.

Error codes may also be returned from HPSS. The most common error codes are:

- -5 - an I/O error occurred.
- -28 - no space remaining in the associated storage class.

See also

RFC-0959.

Notes

none

Examples

1. Append "globally available" file testfile to the same file name in the user's HPSS home directory.

```
ftp> lfappend testfile
. . . (Information returned by the PFTP Daemon)
ftp>
```

2. Append "globally available" file testfile to HPSS file prod1 in the current working directory.

```
ftp> lfappend testfile prod1
. . . (Information returned by the PFTP Daemon)
ftp>
```

3.2.8 Local File store - lfput

Synopsis:

```
lfput [-l local_offset] [-r remote_offset] [-s size] local_file [remote_file]
```

Description

The **lfput** is a performance optimized Parallel FTP Client command used to transfer a "globally available" file into HPSS using the "parallel" protocols. IF the file is not available to the mover(s) machine (NOT "globally available") the transfer will fail because the mover(s) will NOT be able to locate the desired file. The difference between lfput and pput is that the mover(s) involved in the transfer, will not use the network to move the data providing improved performance. The mover(s) machine MUST be correctly configured. The file:

`/var/hpss/etc/hpss_mvr_localfilepath.conf` MUST exist on each "local file" aware mover machine and MUST contain entries specifying which directories are eligible for "local file" transport.

If offsets and size of transfer are not specified, the transfer starts at the beginning of the local file and continues until the entire file is moved or until an error occurs. However, flexibility is provided to perform partial file transfers by specifying local file offset, remote file offset, and size of transfer. The *local_offset*, *remote_offset*, and *size* may be specified using a decimal and magnitude representation string. See Section 2.1.10 for use of this notation. The normal **lfput** command functions just like the standard ftp **put** command and transfers an entire file.

Parameters

-l *local_offset* - Optional byte offset into the "globally available" file where the transfer is to begin.

-r *remote_offset* - Optional byte offset into the remote file where the data is to be placed.

-s *size* - Optional byte size of the amount of data to transfer.

local_file - Identification of the "globally available" file to transfer. (MUST be available to mover(s) machines)

remote_file - Optional file name to the remote (HPSS) file. If not supplied then the remote file name defaults to be the same as the "globally available" file name.

Return strings

Output shows amount of data transferred and any error conditions.

Error conditions

Connection Failures - data transfer connection malfunction.

Network Failures - data transfer malfunction..

Allocation Failures - no space on remote machine for file.

Error codes may also be returned from HPSS. The most common error codes are:

- -5 - an I/O error occurred.
- -28 - no space remaining in the associated storage class.

See also

RFC-0959.

Notes

none.

Examples

1. Transfer local file testfile in the current working directory of the client to the user's HPSS home directory.

```
ftp> cd ~
. . . (Information returned by the PFTP Daemon)
ftp> lfput testfile
. . . (Information returned by the PFTP Daemon)
ftp>
```

2. Transfer local file testfile in the current working directory of the client to HPSS file prod1 in the current working directory.

```
ftp> lfput testfile prod1
. . . (Information returned by the PFTP Daemon)
ftp>
```

3. Transfer 1MB from offset 1MB of local file testfile in the current working directory of the client to offset 0 of HPSS file /home/bob/prod1 with a new name testfile2..

```
ftp> lfput -l 1048576 -r 0 -s 1048576 testfile
/home/bob/prod1/testfile2
. . . (Information returned by the PFTP Daemon)
ftp>
```

3.2.9 Local file retrieval - lfget

Synopsis

```
lfget [-r remote_offset] [-l local_offset] [-s size] remote_file [local_file]
```

Description

The **lfget** is a performance optimized Parallel FTP Client command used to transfer an HPSS file into a "globally available" file using the "parallel" protocols. IF the current working directory or the specified

directory is not available to the mover(s) machine (NOT "globally available") the transfer will fail because the mover(s) will NOT be able to locate the desired file. The difference between `lfget` and `pget` is that the mover(s) involved in the transfer, will not use the network to move the data providing improved performance. The mover(s) machine MUST be correctly configured. The file: `/var/hpss/etc/hpss_mvr_localfilepath.conf` MUST exist on each "local file" aware mover machine and MUST contain entries specifying which directories are eligible for "local file" transport.

The `lfget` command transfers a file from HPSS to a "globally available" directory on the mover machine(s). If offsets and size of transfer are not specified, the transfer starts at the beginning of the remote file and continues until the entire file is moved or until an error occurs. However, flexibility is provided to perform partial file transfers by specifying remote file offset, local file offset, and size of transfer. The `local_offset`, `remote_offset`, and `size` may be specified using a decimal and magnitude representation string. See Section 2.1.10 for use of this notation.

The standard `lfget` command transfers entire files similar to the standard `ftp` get command.

Parameters

`-r remote_offset` - Optional byte offset where transfer is to begin in the remote file.

`-l local_offset` - Optional parameter where the data is transferred in the local file.

`-s size` - Optional number of bytes to transfer.

`remote_file` - Identification of the file to transfer from the remote (HPSS) host.

`local_file` - Optional file name to the local file. If not supplied then the local file name defaults to be the same as the remote file name.

Return strings

Output shows the amount of data transferred and any error conditions.

Error conditions

Connection Failures - data transfer connection malfunction.

Network Failures - data transfer malfunction.

Allocation Failures - no space on local machine for file.

Error codes may also be returned from HPSS. The most common error code is:

- -5 - an I/O error occurred.

See also

RFC-0959.

Notes

none.

Examples

1. Transfer HPSS file /home/bob/prod1 to the user's local directory.

```
ftp> lfget /home/bob/prod1
. . . (Information returned by the PFTP Daemon)
ftp>
```

2. Transfer the HPSS file prod1 in the current working directory to local file testfile1.

```
ftp> lfget prod1 testfile1
. . . (Information returned by the PFTP Daemon)
ftp>
```

3. Transfer the HPSS file testfile into the "globally available" directory /home/bob renaming the file to testfile.

```
ftp> lfget prod1 /home/bob/testfile1 testfile
. . . (Information returned by the PFTP Daemon)
ftp>
```

4. Transfer 1MB from offset 0 of HPSS file /home/bob/prod1 to offset 1048576 of local file testfile.

```
ftp> lfget -r 0 -l 1048576 -s 1048576 /home/bob/testfile1 testfile
. . . (Information returned by the PFTP Daemon)
ftp>
```

3.2.10 Multiple Local file store - mlftp

Synopsis

mlftp *local_files*

Description

The **mlftp** is a performance optimized Parallel FTP Client command used to transfer multiple "globally available" files into HPSS using the "parallel" protocols. IF the file(s) is not available to the mover(s) machine (NOT "globally available") the transfer will fail because the mover(s) will NOT be able to locate the desired file. The difference between mlftp and mpput is that the mover(s) involved in the transfer, will not use the network to move the data providing improved performance. The mover(s) machine MUST be correctly configured. The file: /var/hpss/etc/hpss_mvr_localfilepath.conf MUST exist on each "local file" aware mover machine and MUST contain entries specifying which directories are eligible for "local file" transport.

The **mlftp** command expands the files specified in the *local_files* parameter at the local host and copies the indicated files to HPSS. The **mlftp** command functions just like the standard ftp **mput** command.

Parameters

local_files - Identification of the files to transfer on the local machine.

Return strings

Output shows the amount of data transferred and any error conditions.

Error conditions

Connection Failures - data transfer connection malfunction.

Network Failures - data transfer malfunction.

Allocation Failures - no space on remote machine for file.

Error codes may also be returned from HPSS. The most common error codes are:

- -5 - an I/O error occurred.
- -28 - no space remaining in the associated storage class.

See also

RFC-0959.

Notes

none.

Examples

1. Transfer all "globally available" files in the current directory to the user's HPSS home directory.

```
ftp> cd ~
. . . (Information returned by the PFTP Daemon)
ftp> prompt(<== Toggles File Prompting)
. . .
ftp> mlfput *
. . . (Information returned by the PFTP Daemon)
ftp>
```

2. Transfer all "globally available" files which begin with test in directory /usr/bob to the user's PSS home directory.

```
ftp> cd ~
. . . (Information returned by the PFTP Daemon)
ftp> mlfput /usr/bob/test*
. . . (Information returned by the PFTP Daemon)
ftp>
```

3.2.11 Multiple Local file retrieval - *mlfget*

Synopsis

```
mlfget remote_files
```

Description

The **mlfget** is a performance optimized Parallel FTP Client command used to transfer an HPSS file into a "globally available" file using the "parallel" protocols. IF the current working directory or the specified directory is not available to the mover(s) machine (NOT "globally available") the transfer will fail because the mover(s) will NOT be able to locate the desired file. The difference between **mlfget** and **mpget** is that the mover(s) involved in the transfer, will not use the network to move the data providing improved performance. The mover(s) machine MUST be correctly configured. The file: `/var/hpss/etc/hpss_mvr_localfilepath.conf` MUST exist on each "local file" aware mover machine and MUST contain entries specifying which directories are eligible for "local file" transport.

The **mlfget** command expands the *remote_files* parameter at the remote (HPSS) host and copies the indicated HPSS files to the current directory on the local host. The **mlfget** command functions just like the standard ftp **mget** command.

Parameters

remote_files - Identification of the files to transfer from the remote (HPSS) host.

Return strings

Output shows the amount of data transferred and any error conditions.

Error conditions

Connection Failures - data transfer connection malfunction.

Network Failures - data transfer malfunction.

Allocation Failures - no space on remote machine for file.

Error codes may also be returned from HPSS. The most common error code is:

- -5 - an I/O error occurred.

See also

RFC-0959.

Notes

none.

Examples

1. Transfer all files in HPSS directory `/home/bob` to the "globally available" current working directory

```
ftp> mlfget /home/bob/*
. . . (Information returned by the PFTP Daemon)
ftp>
```

2. Transfer all HPSS files which begin with `test` in directory `/home/bob` to the "globally available" current working directory.

```
ftp> mlfget /home/bob/test*
. . . (Information returned by the PFTP Daemon)
```

ftp>

3.2.12 *Specify TCP socket based transfers - psocket*

Synopsis

psocket

Description

The **psocket** command is used to specify to the FTP client code that any parallel transfers are now to be done using connection based sockets (TCP).

Parameters

none.

Return strings

"Parallel transfers will now go over sockets."

Error conditions

none.

See also

RFC-0959.

Notes

psocket is the default.

3.2.13 *Specify transfer stripe width - setpwidth*

Synopsis

setpwidth *stripe_width*

Description

The **setpwidth** command is used to specify the size of the client side stripe to the FTP client code.

Parameters

stripe_width - The width of the PFTP client-side stripe. The width can have a value of 1 through 16. The default width is 1. The stripe width from the PFTP client perspective is the number of client processes spawned to handle the data transfers. Stripe width from the server perspective is the number of volumes the file is striped across.

A general guideline would be to set *stripe_width* to an even divisor of the number of volumes the file is striped across. For example, if the Class of Service for a file were set up for a 4-way stripe, suggested values for *stripe_width* might be 2 or 4.

If the stripe width of the file is unknown, consult your HPSS administrator or follow the following steps to determine the stripe width.

1. Enter the **lshpss -cos** command to list Class of Service information. From the entry with the *CID* value equal to your Class of Service ID, locate the *HID* (hierarchy ID) field.
2. Enter the **lshpss -hier** command to list hierarchy information. From the entry matching the *HID* value above, locate the *StorageClassID* field.
3. Enter the **lshpss -sc** command to list the storage class information. From the entry matching the *StorageClassID* value above, locate the *W* field. This is the stripe width for the storage class of the file.

Return strings

"Parallel stripe width set to [stripe width]."

Error conditions

"Bad width value [stripe width]."

See also

RFC-0959.

Notes

none.

Example

1. Set the stripe width to 4.

```
ftp> setpwidth 4
```

3.2.14 Specify transfer block size - *setpblocksize*

Synopsis

```
setpblocksize block_size
```

Description

The **setpblocksize** command is used to specify the block size to be used for parallel transfers. The *block_size* may be specified using a decimal and magnitude representation string. See Section 2.1.10 for use of this notation. The maximum blocksize is 16mb.

Parameters

block_size - The number of bytes to be transferred to each element of the stripe before data is sent to the next element. The current allowable transfer sizes range from 1 through 16MB. The default block size is 256KB

A general guideline would be to set *block_size* to the virtual volume block size. Consult your HPSS administrator or follow the following steps to determine the virtual volume blocksize.

1. Enter the **lshpss -cos** command to list Class of Service information. From the entry with the *CID* value equal to your Class of Service ID, locate the *HID* (hierarchy ID) field.
2. Enter the **lshpss -hier** command to list hierarchy information. From the entry matching the *HID* value above, locate the *StorageClassID* field.
3. Enter the **lshpss -sc** command to list the storage class information. From the entry matching the *StorageClassID* value above, locate the *VVBlk* field. This is the virtual volume block size.

Return strings

"Parallel block size set to [block size]."

Error conditions

"Bad block size value [block size]."

See also

RFC-0959.

Notes

none.

Example

1. Set the transfer block size to 8 MB.

```
ftp> setpblocksize 8388608
```

or

```
ftp> setpblocksize 8MB
```

3.2.15 Multinode Enable/Disable - multinode

Synopsis

multinode

Description

The **multinode** command is used to enable/disable the "desire" to perform a parallel file transfer using multiple nodes. When multinode is enabled, the **pftp_client** will process the multinode configuration

file. If the process cannot obtain a single node to perform the parallel transfer, then the transfer will occur using non-multinode parallel method.

Parameters

None.

Return strings

“Processing the multinode list, please wait.....”

“Multinode is on.”

or

“Multinode is off.”

Error conditions

“Configuration File I/O Problems: Without nodes, files cannot be transferred using the multiple node capability.”

See also

None

3.2.16 *Autoparallel Enable/Disable - autoparallel*

Synopsis

autoparallel

Description

The **autoparallel** command is used to enable/disable the automatic mapping of non-parallel commands to parallel commands; e.g., get ==>pget. In autoparallel mode (enabled), transfers involving files smaller than the "Auto Parallel Size" specification in the HPSS.conf will NOT be auto-mapped.

Parameters

None.

Return string

Automatic Substitution of Parallel Commands Disabled

Daemon supports Parallel Features - Auto-Parallel Substitution Enabled

Error conditions

?Invalid command

See also

HPSS.conf(7)

Examples

```
ftp> autop
Automatic Substitution of Parallel Commands Disabled
```

or

```
Daemon supports Parallel Features - Auto-Parallel Substitution Enabled
```

3.2.17 *Get Current Protocol Mode - getprot*

Synopsis

```
getprot
```

Description

Display the current Parallel protocol mode.

Parameters

None.

Return strings

Current Parallel Protocol is PDATA and MOVER to MOVER

Current Parallel Protocol is PDATA ONLY

Error conditions

?Invalid command ==> Older Client?

See also

None.

Examples

```
ftp> getprot
Current Parallel Protocol is PDATA and MOVER to MOVER
Current Parallel Protocol is PDATA ONLY
```

3.2.18 *Get Tuning Parameters - gettuningparms*

Synopsis

```
gettun hostname/IP Addr
```

Description

Display the (transfer) parameters between the Client and other hosts (default is the PFTP Daemon host.)

Parameters

None.

Return strings

See Example below.

Error conditions Warning Preceding without HPSS.conf (-2) (Observed at Login Time)

HPSS.conf parsing errors.

See also

None.

Example

```
ftp> gettun
Effective Tuning parameters from saux22 to sair031.sandia.gov
  Using PDATA_AND_MOVER protocol
  Using 4.1 Protocol
  Parallel Transfer Size = 2147483647
  Transfer Buffer Size = 16777216
  Parallel Block Size = 262144

  Parallel Network Width = 1
  No Interfaces Found (ret_code = -2)
  Using Default Interface for 1 stripes(s)
  Multinode is disabled
```

or

```
Multinode Enabled:
Processing the multinode list, please wait.....
Using 1 remote node(s) from the following:
Control Interface ==> Data Interface:
water ==> water

Using Network Options
  Using "Default" destination characteristics
  PdataSockBufSize = 1048576 based on user input
  recv_socksize = send_socksize = 1048576 based on
  PdataSockBufSize
  Writesize = 524288
```

or

NOTE: Parallel Pipes are NOT supported regardless of the following:



```
recv_socksize = 262144 based on Network Options
send_socksize = 262144 based on Network Options
PdataSockBufSize = 262144 based on send_socksize
```

```
RFC1323 is turned on
TCPNoDelay is turned on
PipeFileSize TOO Large reset to 2147483647
Pipe Files NOT supported on this machine - Open Failed 2
```

or

```
Pipe Files are supported on this machine
Pipe File = /copylvol/.pftp_pipes26404
Pipe File Size = 1073741824
```

3.2.19 Set the *PDATA_ONLY* protocol - *pdata*

Synopsis

pdata

Description

Explicitly request the *PDATA_ONLY* protocol.

Parameters

None.

Return strings

```
**** NOTE: Protocol set to PDATA_ONLY **** (At logon time)
```

```
215 Parallel protocol is PDATA_ONLY
```

Error conditions

?Invalid command==> Older Client?

See also

None.

Examples

1. Set protocol to PDATA_ONLY (Failure)

```
ftp> pdata
Server does NOT support command==> Older Server?
?Invalid command==> Older Client?
ftp>
```

3.2.20 *Override the Non-AutoParallel mode - penable*

Synopsis

penable

Description

Explicitly specify parallel transfers. This is useful when transferring data to a Parallel FTP Daemon which is unaware of the PRL command. This command is included for backward compatibility only and may be discontinued at any time.

Parameters

None.

Return strings

None

Error conditions

?Invalid command==> Older Client?

See also

None.

Example

1. Set parallel enable

```
ftp> penable
ftp>
```

3.2.21 *Set the PDATA_AND_MOVER protocol - pmover*

Synopsis

pmover

Description

Chapter 3 Parallel File Transfer Protocol (PFTP)

Explicitly specify parallel transfers to use the PDATA_AND_MOVER protocol (Original protocol) regardless of what is specified in the HPSS.conf file.

Parameters

None.

Return strings

215 Parallel protocol is PDATA_AND_MOVER

Error conditions

ftp> pmover

Server does NOT support command==> Older Server?

?Invalid command==> Older Client?

See also

HPSS.conf(7)

Examples

1. Set PDATA_AND_MOVER protocol

```
ftp> pmover
215 Parallel protocol is PDATA_AND_MOVER
ftp>
```

3.2.22 Set the Socket Buffer Size - setsockbufsize

Synopsis

setsock *SizeString*

Description

Set the desired socket buffer size. Useful when no HPSS.conf file exists or the client/mover combination is NOT in the HPSS.conf file. When entered without a *SizeString*, the command returns the Socket Buffer Size in effect.

Parameters

SizeString; e.g., "1MB"

Return strings

Socket Buffer Size = 1048576.

Error conditions

PdataSockBufSize reset equal or below sb_max (1048576)

See also

None.

Example

1. Set Socket Buffer Size

```
ftp> setsock 4mb (Above system max)
PdataSockBufSize reset equal or below sb_max (1048576)
ftp>
```

2. Set Socket Buffer Size

```
ftp> setsock 512kb
ftp>
```

3. Set Socket Buffer Size (no argument)

```
ftp> setsock
Socket Buffer Size = 524288.
ftp>
```

3.2.23 Set the Transfer Buffer Size - *setxferbufsize*

Synopsis

setxferbufsize *SizeString*

Description

Set the desired transfer buffer sizes. Useful when no HPSS.conf file exists or the client/daemon combination is NOT in the HPSS.conf file. When entered without a *SizeString*, the command returns the Transfer Buffer Size in effect.

Parameters

SizeString; e.g., "4MB"

Return strings

PdataBufferSize = 4194304

Error conditions

?Invalid command==> Old Client?

See also

None.

Example

1. Set Transfer Buffer Size

```
ftp> setxfer 40MB (Max is 32MB)
ftp>
```

2. Display effective Transfer Buffer Size (no argument)

```
ftp> setxfer
Socket Buffer Size = 33554432.
ftp>
```

HPSS Tar (HTAR)

This chapter specifies the HTAR interface. HTAR is supported on any platform that supports the Non-DCE Client API.

Before running HTAR, you must source the environment variables 'HPSS_HOSTNAME' and 'HPSS_NDCG_SERVERS' in order for the Non-DCE Client API to talk to the appropriate Non-DCE Client Gateway (NDCG). To do this, perform the following commands before invoking HTAR (or add them to your profile):

```
% export HPSS_HOSTNAME=<hostname of node where htar is being evoked>
% export HPSS_NDCG_SERVERS=<hostname of root HPSS node>/
<port number set in the NDC Gateway's server configuration>
```

4.1 Usage

HTAR supports a subset of the tar command set for archiving/extracting files into/out of HPSS or the local file system. To use HTAR, use the following syntax:

```
htar    -{c|t|x|X} -f Archive [-?] [-B] [-E] [-L inputlist]
        [-h] [-m] [-o] [-d debuglevel] [-p] [-v] [-V] [-w]
        [-I {IndexFile | .suffix}]
        [-Y [Archive COS ID][:Index File COS ID]]
        [-S Bufsize] [-T Max Threads]
        [Filespec | Directory ...]
```

where:

- 'Archive' is the name of the archive file being created or extracted.
- 'inputlist' is a file that contains the list of files to be archived.
- 'debuglevel' is the level of desired debug output (default = 0 = none).
- 'IndexFile' is the name of the index file.
- 'Archive COS ID' is the desired COS ID of the HPSS archive file being created.
- 'Index File COS ID' is the desired COS ID of the HPSS index file being created.

- 'Bufsize' is the desired size of the I/O buffer to use when creating an archive (default is 4MB).
- 'Max Threads' is the maximum number of threads to use when creating an archive (default is 20).
- 'Filespec' is the list of filenames to archive or extract.
- 'Directory' is the name of the directory to archive or extract.

4.2 HTAR Action Flags

HTAR must be invoked with exactly one action flag. Omitting the action flag or using more than one will result in an error. The valid action flags are listed below.

-c

Creates a new HPSS-resident archive, and writes the local files specified by one or more Fileparameters into the archive. Warning: any pre-existing archive file will be overwritten without prompting. This behavior mimics that of the AIX tar utility.

-t

Lists the files in the order in which they appear in the HPSS-resident archive. Listable output is written to standard output; all other output is written to standard error.

-x

Extracts the files specified by one or more File parameters from the HPSS-resident archive. If the File parameter refers to a directory, the htar command recursively extracts that directory and all of its subdirectories from the archive.

If the File parameter is not specified, htar extracts all of the files from the archive. If an archive contains multiple copies of the same file, the last copy extracted overwrites all previously extracted copies. If the file being extracted does not already exist on the system, it is created. If you have the proper permissions, then htar command restores all files and directories with the same owner and group IDs as they have on the HPSS tar file. If you do not have the proper permissions, then files and directories are restored with your owner and group IDs.

-X

Builds a new index file by reading the entire tar file. This operation is used either to reconstruct an index for tar files whose Index File is unavailable (e.g., accidentally deleted), or for tar files that were not originally created by htar.

4.3 HTAR Options

-?

Displays htar's verbose help

-B

Displays block numbers as part of the listing (-t option). This is normally used only for debugging.

-d debuglevel

Sets debug level (0 - N) for htar. 0 disables debug, 1 - n enable progressively higher levels of debug output. 5 is the highest level; anything > 5 is silently mapped to 5. 0 is the default debug level.

-E

If present, specifies that a local file should be used for the file specified by the "-f Archive" option. If not specified, then the archive file will reside in HPSS.

-f Archive

Uses Archive as the name of archive to be read or written. Note: This is a required parameter for htar, unlike the standard tar utility, which uses a built-in default name.

If the Archive variable specified is - (minus sign), the tar command writes to standard output or reads from standard input. If you write to standard output, the -I option is mandatory, in order to specify an Index File, which is copied to HPSS if the Archive file is successfully written to standard output. [Note: this behavior is deferred - reading from or writing to pipes is not supported in the initial version of htar].

-h

Forces the htar command to follow symbolic links as if they were normal files or directories. Normally, the tar command does not follow symbolic links.

-I index_name

Specifies the index file name or suffix. If the first character of the index_name is a period, then index_name is appended to the Archive name, e.g. "-f the_htar -I .xndx" would create an index file called "the_htar.xndx". If the first character is not a period, then index_name is treated as a relative pathname for the index file (relative to the Archive file directory) if the pathname does not start with "/", or an absolute pathname otherwise.

The default directory for the Index file is the same as for the Archive file. If a relative Index file pathname is specified, then it is appended to the directory path for the Archive file. For example, if the Archive file resides in HPSS in the directory "projects/prj/files.tar", then an Index file specification of "-I projects/prj/files.old.idx" would fail, because htar would look for the file in the directory "projects/prj/projects/prj". The correct specification in this case is "-I files.old.idx".

-L InputList

Writes the files and directories listed in the "InputList" file to the archive. Directories named in the InputList file are not treated recursively. For directory names contained in the InputList file, the tar command writes only the directory entry to the archive, not the files and subdirectories rooted in the directory. Note that "home directory" notation ("~") is not expanded for pathnames contained in the InputList file, nor are wildcard characters, such as "*" and "?".

-m

Uses the time of extraction as the modification time. The default is to preserve the modification time of the files. Note that the modification time of directories is not guaranteed to be preserved, since the operating system may change the timestamp as the directory contents are changed by extracting other files and/or directories. htar will explicitly set the timestamp on directories that it extracts from the Archive, but not on intermediate directories that are created during the process of extracting files.

-o

Provides backwards compatibility with older versions (non-AIX) of the tar command. When this flag is used for reading, it causes the extracted file to take on the User and Group ID (UID and GID) of the user running the program, rather than those on the archive. This is the default behavior for the ordinary user. If htar is being run as root, use of this option causes files to be owned by root rather than the original user.

-p

Says to restore fields to their original modes, ignoring the present umask. The setuid, setgid, and tacky bit permissions are also restored to the user with root user authority.

-S bufsize

Specifies the buffer size to use when reading or writing the HPSS tar file. The buffer size can be specified as a value, or as kilobytes by appending any of "k", "K", "kb", or "KB" to the value. It can also be specified as megabytes by appending any of "m" or "M" or "mb" or "MB" to the value, for example, 23mb.

-T max_threads

Specifies the maximum number of threads to use when copying local member files to the Archive file. The default is defined when htar is built; the release value is 20. The maximum number of threads actually used is dependent upon the local file sizes, and the size of the I/O buffers. A good approximation is usually

$$(\text{average file size}) / (\text{buffersize})$$

If the `-v` or `-V` option is specified, then the maximum number of local file threads used while writing the Archive file to HPSS is displayed when the transfer is complete.

-V

"Slightly verbose" mode. If selected, file transfer progress will be displayed in interactive mode. This option should normally not be selected if verbose (`-v`) mode is enabled, as the outputs for the two different options are generated by separate threads, and may be intermixed on the output.

-v

"Verbose" mode. For each file processed, displays a one-character operation flag, and lists the name of each file. The flag values displayed are:

"a" - file was added to the archive

"x" - file was extracted from the archive

"i" - index file entry was created (Build Index operation)

-w

Displays the action to be taken, followed by the file name, and then waits for user confirmation. If the response is affirmative, the action is performed. If the response is not affirmative, the file is ignored.

-Y auto / [Archive CosID][:IndexCosID]

Specifies the HPSS Class of Service ID to use when creating a new Archive and/or Index file. If the keyword `auto` is specified, then the HPSS hints mechanism is used to select the archive COS, based upon the file size. If `-Y cosID` is specified, then `cosID` is the numeric COS ID to be used for the Archive File.

If `-Y :IndexCosID` is specified, then `IndexCosID` is the numeric COS ID to be used for the Index File. If both COS IDs are specified, the entire parameter must be specified as a single string with no embedded spaces, e.g. `"-Y 40:30"`.

4.4 HTAR Examples

1. To write the files "file1" and "file2" to a new archive called "files.tar" in the current HPSS home directory, enter:

```
% htar -cvf files.tar file1 file2
a file1
a file2
a /tmp/HTAR_CF_CHK_20890_1063223592
Create complete. 1,730,048 bytes written, max threads: 4
Transfer time: 0.742 seconds (2.331 MB/s)
```

2. To extract all files from the "project1/src" directory in the Archive file called "proj1.tar", and use the time of extraction as the modification time, enter:

```
%htar -xvm -f proj1.tar project1/src
x project1/src
x project1/src/file1, 847668 bytes, 1657 media blocks
x project1/src/file2, 879332 bytes, 1719 media blocks
Extract complete. total bytes read: 1,729,024 in 0.836 seconds
(2.067 MB/s )
```

3. To display the names of the files in the "out.tar" archive file within the HPSS home directory, enter:

```
%htar -tvf proj1.tar
drwxrwxr-x hpss/hpss          0 2003-09-10 14:56 project1/
drwxrwxr-x hpss/hpss          0 2003-09-10 14:57 project1/src/
-rwxrwxr-x hpss/hpss      847668 2003-09-10 14:51
project1/src/file1
-rwxrwxr-x hpss/hpss      879332 2003-09-10 14:51
project1/src/file2
-rw----- hpss/hpss          256 2003-09-10 14:58
/tmp/HTAR_CF_CHK_20923_1063223886
% htar -tvf out.tar
```

4. To create the index file for the archive file "file.tar", which exists in the HPSS home directory, enter:

```
%htar -Xvf files.tar
[build index] Overwrite existing index file 'files.tar.idx'? [Y/N] Y
i file1
i file2
i /tmp/HTAR_CF_CHK_20890_1063223592
```

HPSS provides a set of utilities for administrators and users. The majority of the HPSS utilities are for administrators, and are defined in the HPSS Administration Guide. Those utilities applicable to users are documented in this chapter. There are also man pages for these utilities.

5.1 Utilities

The user utilities defined in the chapter are:

hacl

lshpss

5.1.1 HPSS ACL Editor - hacl

What follows is the HPSS manual page for hacl:

NAME

hacl - manage HPSS access control lists

SYNTAX

hacl [-x] [-v] [command ...]

DESCRIPTION

The hacl utility is used to manage access control lists (ACLs) on HPSS files. In particular, hacl can replace an ACL with a different one, add new ACL entries or update existing ones, delete selected ACL entries, clear out an entire ACL, and display ACLs. To make it easier to change ACLs, hacl can also copy an ACL from an existing file to another file.

If a command is supplied on the execute line, then the command is executed immediately and control is returned to the shell.

Otherwise hacl enters interactive mode. In this mode, the user can change working directories, look at the files in the current directory, try out acl changes, and see the results of the

change. Interactive mode may also be used to read commands from a script.

Access control lists are made up of individual ACL entries separated by "white space". White space characters include commas, semicolons, and newlines, as well as spaces and tabs. Each ACL entry takes the form "<type>:<key>:<perms>", where:

<type>	the ACL entry type (e.g., "user_obj")
<key>	principal and/or cell name
<perms>	the permissions (e.g., "rwx")

The <type> field can be any of the standard DCE ACL entry types, including user_obj, group_obj, other_obj, user, group, foreign_user, foreign_group, foreign_other, any_other, mask_obj, unauthenticated, user_obj_delegate, group_obj_delegate, other_obj_delegate, user_delegate, group_delegate, foreign_user_delegate, foreign_group_delegate, foreign_other_delete, and any_other_delegate.

The <key> field takes the form /<cell>/<principal>. The <cell> or <principal> may or may not be required, depending on the ACL entry type. The <cell> can be specified with either a cell name (e.g., /.../hpss.ca.sandia.gov) or a cell id (e.g., /87654321).

Similarly, users can be specified either by name (e.g., hilary) or by uid (e.g., 20021), and groups can be specified by name or gid. Normally the user will use names rather than ids, but ids are provided to deal with situations where the cell, principal, and/or group have been removed from the security registry.

The <perms> field may or may not be required, depending on the command. When required, the field is made up the characters "rwxcid", signifying respectively read, write, execute, control, insert, and delete. To specify that a permission is not wanted, use a hyphen (-) or just omit the character corresponding to that permission.

Many commands can optionally act on the initial container or initial object ACL of a directory. To specify one of these, use the -ic or -io flag on the command line. For example, to clear the initial container ACL on mydirectory:

```
$ hacl clear -ic -f mydirectory
```

The -ic and -io flags are mutually exclusive. If neither is present on the command line, then the object ACL itself will be processed.

Many commands require that a list of ACL entries be entered. The list can be specified in one of two ways using the -a or -A flags. The flags are mutually exclusive.

If the -a flag is used, then <entries> is an explicit list of ACL entries to be processed. If the -A flag is used, then <filea> is the name of Unix file that contains the ACL entries. The format

of the file is flexible, but it is recommended that each ACL entry appear on a separate line.

Many commands also require the user to specify the list of files that are to be processed. The list can be specified in one of three ways using the `-B`, `-f`, and `-F` flags. The flags are mutually exclusive.

If the `-f` flag is used, it must be the last flag on the command line. In this case, `<files>` is a list of files to be processed. The list may contain files whose names begin with a hyphen. With the exception of the `"ls"` command, the list may not contain wildcards or escape characters.

If the `-F` flag is used, then `<filef>` is the name of a Unix file that contains the names of the HPSS files to be processed. The file must be formatted with one file to a line. Leading blank characters are ignored, but embedded and trailing blanks are considered to be part of the file name. Asterisks (`*`), question marks (`?`), and escapes (`\`) are not translated but rather are treated as part of the file name.

In addition a bulk file processing option MAY be available in some future version of `hacl`. This option, selected by the `-B` flag, allows `hacl` to change ACLs more efficiently. To use the option, a Unix file must first be prepared containing the HPSS nameserver handle and the name of each file that is to be processed. To prepare the Unix file, use `hacl's "ls -h"` command. Once the file has been prepared, use `hacl's -B` flag, naming the Unix file as `<fileb>`. One of the examples below shows how to do this in more detail.

To designate input from `stdin`, use a hyphen (`-`) for the `<filea>`, `<fileb>`, or `<filef>` parameters. Needless to say, only one of these parameters can be set to `stdin` at a time. Moreover, the hyphen cannot be used in interactive mode.

The `"ls"` is the only `hacl` command that accepts wildcards and escape characters. The asterisk (`*`) and question mark (`?`) have the same interpretation as they do in most Unix shells. The backslash (`\`) is an escape character that can be used to represent troublesome characters. The following escape sequences are recognized:

<code>\t</code>	the tab character
<code>\'</code>	the apostrophe (<code>'</code>)
<code>\"</code>	the double quote (<code>"</code>)
<code>\(blank)</code>	the blank () character
<code>\nnn</code>	the ASCII character given by 1-3 octal digits
<code>\\</code>	the backslash (<code>\</code>) character itself
<code>\(other)</code>	unchanged

Escape sequences are translated before wildcards are evaluated, which has the practical effect that there is no simple way to specify a file name that contains an asterisk or question mark.

Note that file names can either be specified as absolute or relative path names. When hacl starts, it determines the user's default working directory from HPSS. However, if the environment variable `HACL_DEFAULT_DIR` is set, then its value will determine the default directory. The directory can be changed by using hacl's "cd" command.

Some commands support a `-q` flag. If the flag is missing, then information is printed out for every file that is processed. For example:

```
$ hacl copy -m model -f file.01 file.02
  file.01      OK
  file.02      Cannot change ACL. (No such name)
```

If the `-q` flag is present, only the commands that fail will be reported. This makes it easier to see errors. For example:

```
$ hacl copy -q -m model -f file.01 file.02
  file.02      Cannot change ACL. (No such name)
```

By default, when hacl encounters an error, it continues running, processing other files and hacl commands until a quit or exit command is seen. However, this can be dangerous if hacl is being run from a script. For example, if a "cd" command fails, hacl might end up processing files in the wrong directory. To prevent this from happening, always use the `-x` flag when running from a script. The `-v` flag should also be used so that hacl's messages can be related to the commands that generated them.

PARAMETERS

command	Optional one-liner command and parameters.
-x	Terminate commands as soon as an error occurs.
-v	Echo hacl commands to stdout.

COMMAND SUMMARY

acl	Description of the syntax for access control lists
cd	Change the working directory
clear	Clear the entire ACL for one or more files
copy	Copy the ACL from one file to one or more other files
echo	Echo the arguments to standard output
exit	Leave the program
help	Get help on how to use hacl
ls	List the files in a directory
purge	Purge unused ACL perms to account for mask object
pwd	Print the current working directory
remove	Remove selected ACL entries from one or more files
replace	Replace entire ACL with new ACL for one or more files
quit	Leave the program
show	Show the ACL for one or more files
update	Update selected ACL entries for one or more files
usage	Describe hacl's command line

COMMAND DETAILS

acl

Description of the syntax for access control lists

This is not a command per se, but rather a topic for which help is available. Typing the command "help acl" results in a description of the syntax for ACLs.

cd [<dir>]

Change the working directory

<dir> New working directory

If <dir> is missing, this command changes the current working directory to be the user's default working directory.

clear [-ic | -io] [-q] [-B <fileb> | -F <filef> | [-f] <files>]
 Clear the entire ACL for one or more files

-ic the ACL is an initial container ACL
 -io the ACL is an initial object ACL
 -q don't report successes
 -B bulk ACL change for files listed in <fileb>
 <fileb> a Unix text file listing file names and NS handles
 -F file names are determined by reading <filef>
 <filef> a Unix text file listing file names
 -f file names explicitly listed in <files>
 <files> an explicit list of files

This command deletes all ACL entries except the user_obj, group_obj, other_obj entries.

See the discussion section for a description of the flags.

copy [-ic | -io] [-q] -m <filem> [-B <fileb> | -F <filef> | \ [-f] <files>]
 Copy the ACL from one file to one or more other files

-ic the ACL is an initial container ACL
 -io the ACL is an initial object ACL
 -q don't report successes
 -m change ACL to that given by "model" file <filem>
 <filem> an HPSS file whose ACL will be copied
 -B bulk ACL change for files listed in <fileb>
 <fileb> a Unix text file listing file names and NS handles
 -F file names are determined by reading <filef>
 <filef> a Unix text file listing file names
 -f file names explicitly listed in <files>
 <files> an explicit list of files

The -m flag is used to specify the name of the HPSS file whose ACL is to be copied. This flag is required.

See the discussion section for a description of the other flags.

echo [<text>]
Echo the arguments to standard output

<text> the message text to be echoed

exit
Leave the program

help [-v] [<topic>]
Get help on how to use hacl

-v print more details
<topic> topic for which help is desired

If <topic> is omitted, print a list of topics for which help is available.

ls [-a] [-h] [-l] [-f] [<files>]
List the files in a directory

-a list files whose names begin with a dot (.)
-h list file handles needed by the bulk ACL commands
-l list owner's cell name, user name, and group name
-f list the files named in <files>

The -f flag must be last on the command line. The <files> argument can contain wildcards and escape characters. If one of the files named is a directory, the directory itself will be listed rather than the files in that directory. If <files> is omitted, then all of the files in the current working directory will be listed.

The "ls" command is the only hacl command that accepts wildcards. To use wildcards on other commands, the output of the "ls" command must be directed into a file or a pipe and then read back into hacl using the -F flag. For example, this will show the ACLs for all files whose names end in ".dat":

```
$ hacl ls -f '*.dat' | hacl show -F -
```

If the -h flag is used, the output of the "ls" command will be in the right format to use as input to one of the bulk ACL operations. For example, the following command copies the ACL in myfile.dat to all the files in the current working directory:

```
$ hacl ls -h -f '*' | hacl copy -m myfile.dat -B -
```

purge [-ic | -io] [-q] [-B <fileb> | -F <filef> | [-f] <files>]
Purge unused ACL permissions to account for the mask object

-ic the ACL is an initial container ACL
-io the ACL is an initial object ACL
-q don't report successes
-B bulk ACL change for files listed in <fileb>

```

<fileb>  a Unix text file listing file names and NS handles
-F       file names are determined by reading <filef>
<filef>  a Unix text file listing file names
-f       file names explicitly listed in <files>
<files>  an explicit list of files

```

See the discussion section for a description of the flags.

pwd

Print the current working directory

```

remove [-ic | -io] [-q] [-a <entries> | -A <filea>] \
      [-B <fileb> | -F <filef> | -f <files>]
Remove selected ACL entries from one or more files

```

```

-ic      the ACL is an initial container ACL
-io      the ACL is an initial object ACL
-q       don't report successes
-a       ACL entries are explicitly listed
<entries> explicit list of ACL entries
-A       ACL is determined by reading a file
<filea>  Unix text file listing the new ACL entries
-B       bulk ACL change for files listed in <fileb>
<fileb>  a Unix text file listing file names and NS handles
-F       file names are determined by reading <filef>
<filef>  a Unix text file listing file names
-f       file names explicitly listed in <files>
<files>  an explicit list of files

```

See the discussion section for a description of the flags.

```

replace [-ic | -io] [-m <type>] [-q] [-a <entries> | -A <filea>] \
      [-B <fileb> | -F <filef> | -f <files>]
Replace entire ACL with new ACL for one or more files

```

```

-ic      the ACL is an initial container ACL
-io      the ACL is an initial object ACL
-m       adjust mask object according to <type>
<type>  either "calc" or "nocalc"
-q       don't report successes
-a       ACL entries are explicitly listed
<entries> explicit list of ACL entries
-A       ACL is determined by reading a file
<filea>  Unix text file listing the new ACL entries
-B       bulk ACL change for files listed in <fileb>
<fileb>  a Unix text file listing file names and NS handles
-F       file names are determined by reading <filef>
<filef>  a Unix text file listing file names
-f       file names explicitly listed in <files>
<files>  an explicit list of files

```

See the discussion section for a description of the flags.

quit

Leave the program

```
show [-e] [-ic | -io] [-B <fileb> | -F <filef> | [-f] <files>]
Show the ACL for one or more files
```

```
-e          show effective permissions considering mask object
-ic        the ACL is an initial container ACL
-io        the ACL is an initial object ACL
-B         bulk ACL change for files listed in <fileb>
<fileb>   a Unix text file listing file names and NS handles
-F         file names are determined by reading <filef>
<filef>   a Unix text file listing file names
-f         file names explicitly listed in <files>
<files>   an explicit list of files
```

See the discussion section for a description of the flags.

```
update [-ic | -io] [-m <type>] [-q] \
      [-a <entries> | -A <filea>] \
      [-B <fileb> | -F <filef> | -f <files>]
Update selected ACL entries for one or more files
```

```
-ic        the ACL is an initial container ACL
-io        the ACL is an initial object ACL
-m         adjust mask object according to <type>
<type>    either "calc" or "nocalc"
-q         don't report successes
-a         ACL entries are explicitly listed
<entries> explicit list of ACL entries
-A         ACL is determined by reading a file
<filea>   Unix text file listing the new ACL entries
-B         bulk ACL change for files listed in <fileb>
<fileb>   a Unix text file listing file names and NS handles
-F         file names are determined by reading <filef>
<filef>   a Unix text file listing file names
-f         file names explicitly listed in <files>
<files>   an explicit list of files
```

See the discussion section for a description of the flags.

This command can be used to add new entries to an ACL as well as to change existing entries.

```
usage
Describe hacl's command line
```

EXAMPLES

```
To show the ACL on a file:
$ hacl show myfile.dat
```

```
To show the effective ACL on a file:
$ hacl show -e myfile.dat
```

```
To show the initial container ACL on a directory:
```

```
$ hacl show -ic mydir
```

To copy an ACL from a model file to several other files:

```
$ hacl copy -m model.file -f file.00 file.01 file.02
```

To copy an ACL to a list of files specified using wildcards:

```
$ hacl ls '*.dat' | hacl copy -m model.file -F -
```

To update a single ACL entry on a single file:

```
$ hacl update user:hilary:rwx file.00
```

To work with several ACL entries and files:

```
$ hacl update -a user:joe:rwx user:ann:cid -f f1 f2 f3
```

To run hacl interactively and work with mydir/file.00:

```
$ hacl
> show mydir/file.00
> cd mydir
> pwd
> ls
> update -a user:hilary:rwx -f file.00
> show file.00
```

To establish a default working directory (using ksh):

```
$ export HACL_DEFAULT_DIR=/home/hilary/mydir
$ hacl
> show file.00
```

To use an ACL specified in a Unix file:

```
$ cat myacls
user:joe:rwx
user:ann:cid
$ hacl update -A myacls -f f1 f2 f3
```

To modify the files listed in a Unix file:

```
$ cat myfiles
f1
f2
f3
$ hacl update -a user:joe:rwx user:ann:cid -F myfiles
```

To run hacl using a script:

```
$ cat myscript
update update -A myacls -f f1 f2 f3
update -a user:joe:rwx user:ann:cid -F myfiles
$ hacl -v -x < myscript
```

To generate and use the info needed by a bulk ACL command

```
$ hacl ls -h '*.01' > mybulk
$ cat mybulk
000002370000023700000000af4a81017ec823feee7511d1lab4708005a4726ef
```

```
acl.01
```

```
0000023e0000023e00000000273381017ec823feee7511d1lab4708005a4726ef
```

```
ftp.01
```

```
$ hacl update -A myacls -B mybulk
```

FILES

SEE ALSO

SCCS

```
static char SccsId[] = " @(#)33 1.2 man/hacl.7, gen, 5.1,  
5.1.0.0 2/26/03 10:00:17";
```

5.1.2 List information about HPSS - lshpss

What follows is the HPSS manual page for lshpss:

NAME

lshpss - List information about HPSS

SYNTAX

lshpss [options ...]

DESCRIPTION

This utility displays various HPSS resources like Class of Service, Hierarchy, and Storage Class.

The lshpss utility reads metadata directly from DB2. In order for it to work, the user must be logged in as a user that has permission to read HPSS metadata, such as hpss or the DB2 instance owner.

OPTIONS

-glob	Show Global Configuration Data
-subsys	Show Storage Subsystems
-cos	Show Class of Service list
-hier	Show Hierarchy list
-sc	Show Storage Class list
-migp	Show Migration Policies
-purgep	Show Purging Policies
-vol	Show Physical Volumes
-dev	Show Mover Devices
-drv	Show PVL Drives
-svr	Show HPSS Servers
-mvr	Show HPSS Movers
-acct	Show Accounting Policies
-logp	Show Log Policies
-locp	Show Location Policies
-fset	Show Filesets
-ffam	Show File Families
-core	Show Core Servers
-pvr	Show PVRs
-logd	Show Log Daemons
-logc	Show Log Clients
-nfsd	Show NFS Daemons

```

-listmeta      List metadata
-all          Display all of the above

-dumpmeta     Dump all metadata for HPSS to separate files
-dump <table> Dump one metadata table

-s <database schema name>
              If this is not specified, the contents of the
              environment variable HPSS_MM_SCHEMA_NAME will be used

-g <global config database>
              If this is not specified, the contents of the
              environment variable HPSS_GLOBAL_DB_NAME will be
used

-h           Show this help message

```

EXAMPLE

The following command will list all the resources :

```
$ lshpss -all
```

To list classes of service and purge policies :

```
$ lshpss -cos -purgep
```

SEE ALSO

SCCS

```
man/lshpss.7, gen, 5.1  10/30/03  09:44:01
```

5.1.2.1 Sample Output

- Class of Service List -

COS Hier	Optim	File Size	Xfer Avg
ID ID Name	Access	Min / Max	Rate Lat
(s) SC Flags	Size		kB/s
1	1	4MB	512MB
4096 0 O RWAM-		1/	
2	2	4MB	512MB
4096 0 O RWAM-		1/	
3	3	4MB	100GB
4096 0 A RWA--		2MB/	
4	4	4MB	1TB
4096 0 B RWA--		4MB/	
5	5	4MB	1TB
4096 0 O RWA-F		1/	

Chapter 5 User Utilities

6	6	lwt-lwt (fm-2c-2r)	4MB	1/	1TB
4096	0	N RWA-F			
7	7	lwt-lwt (fmp-1c-1r)	4MB	1/	1TB
4096	0	O RWA-F			
8	8	lwt-lwt (fmp-1c-2r)	4MB	1/	1TB
4096	0	N RWA-F			
9	9	lwt-lwt (vm-1c-1r)	4MB	1/	1TB
4096	0	O RWA-F			
10	10	lwt-lwt (vmwf-1c-1r)	4MB	1/	1TB
4096	0	N RWA-F			
11	11	2wt (nm)	4MB	1/	1TB
4096	0	N RWA-F			
12	12	fred: 1wd (H12)	4MB	1/	1TB
4096	0	N RWAM-			
13	13	fred: 2wd (H13)	4MB	1/	1TB
4096	0	N RWAM-			
14	14	fred: 4wd (H14)	4MB	1/	1TB
4096	0	N RWAM-			
15	15	sally: 1wd (H15)	4MB	1/	1TB
4096	0	N RWAM-			
16	16	sally: 2wd (H16)	4MB	1/	1TB
4096	0	N RWAM-			
17	17	sally: 4wd (H17)	4MB	1/	1TB
4096	0	N RWAM-			
18	18	mover3: 1wd (H18)	4MB	1/	1TB
4096	0	N RWAM-			
19	19	mover3: 2wd (H19)	4MB	1/	1TB
4096	0	N RWAM-			
20	20	mover3: 4wd (H20)	4MB	1/	1TB
4096	0	N RWAM-			
21	21	Util Test	4MB	1/	1TB
4096	0	N RWAM-			
22	22	Util Test (LTO)	4MB	1/	1TB
4096	0	N RWAM-			
23	23	Recovery Test	4MB	1/	1TB
4096	0	N RWAM-			
24	24	Shelf Tape COS 1	4MB	1/	1TB
4096	0	N RWAM-			
25	25	Shelf Tape COS 2	4MB	1/	1TB
4096	0	N RWAM-			
27	27	raidzone: 1wd	4MB	1/	1TB
4096	0	N RWAM-			
28	28	raidzone: 2wd	4MB	1/	1TB
4096	0	N RWAM-			
29	29	raidzone: 4wd	4MB	1/	1TB
4096	0	N RWAM-			
30	30	256wd->lwt	4MB	1/	8MB
4096	0	O RWA-F			
116	116	sally: 1w LTO Only (H116)	4MB	1/	1TB
4096	0	N RWAM-			
117	117	ginger: 1w 3592 Only (H117)	4MB	1/	1TB
4096	0	N RWAM-			
118	118	Greg LTO tape only COS 118	4MB	1/	1TB
4096	0	N RWAM-			

```

119 119 mover3: 1w ampex (H119)          4MB          1/          1TB
4096  0  N RWAM-
120 120 mover3: 1w Sony GY-8240 (H120)    4MB          1/          1TB
4096  0  N RWAM-
121 121 1w 9940B only (H121)             4MB          1/          1TB
4096  0  N RWAM-
122 122 fred: 1w STK 9840 tape (H122)     4MB          1/          1TB
4096  0  N RWAM-
123 123 Util Test (STK)                  4MB          1/          1TB
4096  0  N RWAM-
124 124 1w STK 9940B (H124)              4MB          1/          1TB
4096  0  N RWAM-
125 125 1w STK 9940B tape (H125)         4MB          1/          1TB
4096  0  N RWAM-

```

```

-----
-----

```

SC=Stage Code (N=none, O=on open, A=async, B=background)

Flags: RWAMF

R=Enable read

W=Enable write

A=Enable append

M=Enforce max file size

F=Force selection

- Hierarchy List -

Hier	ID Description	#	Lev	Storage Class	IDs
1	1wd-1wt-1wt (dm-2c-1r)	3	1	-> 101	-> 103
2	1wd-1wd-1wt (dm-dm-1c-1r)	3	2	-> 3	-> 109
3	2wd-1wt (dm-1c-2r)	2	4	-> 109	
4	4wd-1wt (dm-1c-2r)	2	5	-> 109	
5	1wt-1wt (fm-2c-1r)	2	103	-> 109	
6	1wt-1wt (fm-2c-2r)	2	104	-> 109	
7	1wt-1wt (fmp-1c-1r)	2	105	-> 109	
8	1wt-1wt (fmp-1c-2r)	2	106	-> 109	
9	1wt-1wt (vm-1c-1r)	2	107	-> 109	
10	1wt-1wt (vmwf-1c-1r)	2	108	-> 109	
11	2wt (nm)	1	110		
12	fred: 1wd (H12)	1	12		
13	fred: 2wd (H13)	1	13		
14	fred: 4wd (H14)	1	14		
15	sally: 1wd (H15)	1	15		
16	sally: 2wd (H16)	1	16		
17	sally: 4wd (H17)	1	17		
18	mover3: 1wd (H18)	1	18		
19	mover3: 2wd (H19)	1	19		
20	mover3: 4wd (H20)	1	20		
21	Util Test	1	111		
22	Util Test (LTO)	1	112		
23	Recovery Test	3	113	-> 111	-> 109
24	Shelf Tape 1	1	114		

Chapter 5 User Utilities

```

25 Shelf Tape 2                1 115
27 raidzone: 1wd              1 27
28 raidzone: 2wd              1 28
29 raidzone: 4wd              1 29
30 256wd->1wt                 2 30 -> 109
116 sally: 1w LTO Only (H116) 1 116
117 ginger: 1w 3592 Only (H117) 1 117
118 Greg Hierarchy 118        1 118
119 mover3: 1w ampex (H119)   1 119
120 mover3: 1w Sony GY-8240 (H120) 1 120
121 1w 9940B only (H121)      1 121
122 fred: 1w STK 9840 tape (H122) 1 122
123 Util Test (STK)           1 123
124 1w STK 9940B (H124)       1 124
125 1w STK 9940B tape (H125)  1 125

```

- Disk Storage Class List -

Media	Thresh	Mig	Prg	SC	Block	ID	Name	Type	Xfer Rate	S	Seg	Size	Avg #	PV	Est	SSegs	Size	Str	VVBS
Wid	Size	Wrn	Crt	ID	ID				(kB/s)	min	max								
1	1wd (H1-L1)							Default	3072	1MB/	16MB	4	4GB	1MB					
1	4kB	80/	90	1	1			Default	3072	1MB/	16MB	4	4GB	1MB					
1	2 1wd (H2-L1)							Default	3072	1MB/	16MB	4	4GB	1MB					
1	4kB	80/	90	1	1			Default	3072	1MB/	16MB	4	4GB	1MB					
1	3 1wd (H2-L2)							Default	3072	1MB/	16MB	4	4GB	1MB					
1	4kB	80/	90	1	1			Default	3072	1MB/	16MB	4	4GB	1MB					
1	4 2wd (H3-L1)							Default	6144	2MB/	64MB	4	4GB	1MB					
2	4kB	80/	90	2	2			Default	6144	2MB/	64MB	4	4GB	1MB					
2	5 4wd (H4-L1)							Default	12288	4MB/	256MB	4	4GB	1MB					
4	4kB	80/	90	2	3			Default	12288	4MB/	256MB	4	4GB	1MB					
12	fred: 1wd (H12)							Default	3072	1MB/	2MB	4	4GB	1MB					
1	4kB	80/	90	0	0			Default	3072	1MB/	2MB	4	4GB	1MB					
13	fred: 2wd (H13)							Default	6144	2MB/	4MB	4	4GB	1MB					
2	4kB	80/	90	0	0			Default	6144	2MB/	4MB	4	4GB	1MB					
14	fred: 4wd (H14)							Default	12288	4MB/	8MB	4	4GB	1MB					
4	4kB	80/	90	0	0			Default	12288	4MB/	8MB	4	4GB	1MB					
15	sally: 1wd (H15)							Default	3072	1MB/	2MB	4	4GB	1MB					
1	4kB	80/	90	0	0			Default	3072	1MB/	2MB	4	4GB	1MB					
16	sally: 2wd (H16)							Default	6144	2MB/	4MB	4	4GB	1MB					
2	4kB	80/	90	0	0			Default	6144	2MB/	4MB	4	4GB	1MB					
17	sally: 4wd (H17)							Default	12288	4MB/	8MB	4	4GB	1MB					
4	4kB	80/	90	0	0			Default	12288	4MB/	8MB	4	4GB	1MB					
18	mover3: 1wd (H18)							Default	3072	1MB/	2MB	4	4GB	1MB					
1	4kB	80/	90	0	0			Default	3072	1MB/	2MB	4	4GB	1MB					
19	mover3: 2wd (H19)							Default	6144	2MB/	4MB	4	4GB	1MB					
2	4kB	80/	90	0	0			Default	6144	2MB/	4MB	4	4GB	1MB					

20 mover3: 4wd (H20)	Default	12288	4MB/	8MB	4	4GB	1MB
4 4kB 80/ 90 0 0							
27 raidzone: 1wd	Default	3072	40MB/	40MB	4	219997MB	
1MB 1 4kB 80/ 90 0 0							
28 raidzone: 2wd	Default	6144	2MB/	4MB	4	219997MB	
1MB 2 4kB 80/ 90 0 0							
29 raidzone: 4wd	Default	12288	4MB/	8MB	4	219997MB	
1MB 4 4kB 80/ 90 0 0							
30 256wd	Default	262144	1MB/	4MB	4	32kB	4kB
256 4kB 80/ 90 1 0							
113 Recovery Test	Default	3072	1MB/	1MB	4	512MB	1MB
1 4kB 80/ 90 1 0							

 VVBS=Virtual volume block size

- Tape Storage Class List -

Threshold Max	Mig Prg		Media Est		Xfer				
SC			Media	Block PV	Str	Rate			
(volumes) VVs to Pol Pol			Type	Size	Size	Wid	VVBS	(kB/s)	BBTM
ID Name									
Warn/Crit Write ID ID									
101 1w 3590E-DL (c1)			3590E DL T	256kB	80GB	1	1MB	14336	512
2/ 1 5 0 0									
102 1w 3590E-DL (c2)			3590E DL T	256kB	80GB	1	1MB	14336	512
2/ 1 5 0 0									
103 1w 3590E-DL (fm-1r)			3590E DL T	256kB	80GB	1	1MB	14336	512
2/ 1 5 3 0									
104 1w 3590E-DL (fm-2r)			3590E DL T	256kB	80GB	1	1MB	14336	512
2/ 1 5 4 0									
105 1w 3590E-DL (fmp-1r)			3590E DL T	256kB	80GB	1	1MB	14336	512
2/ 1 5 5 0									
106 1w 3590E-DL (fmp-2r)			3590E DL T	256kB	80GB	1	1MB	14336	512
2/ 1 5 6 0									
107 1w 3590E-DL (vm-1r)			3590E DL T	256kB	80GB	1	1MB	14336	512
2/ 1 5 7 0									
108 1w 3590E-DL (vmwf-1r)			3590E DL T	256kB	80GB	1	1MB	14336	512
2/ 1 5 8 0									
109 1w 3590E-DL (nm)			3590E DL T	256kB	80GB	1	1MB	14336	512
2/ 1 5 0 0									
110 2w 3590E-DL Tape Only			3590E DL T	256kB	80GB	2	1MB	28672	512
2/ 1 2 0 0									
111 Util Test			3590E DL T	256kB	80GB	1	1MB	14336	512
10/ 5 10 3 0									
112 Util Test (LTO)			3580 Gen1	256kB	100GB	1	1MB	15360	1024
10/ 5 10 0 0									
114 Shelf Tape 1			3590E DL T	256kB	80GB	1	1MB	14336	512
10/ 5 10 0 0									
115 Shelf Tape 2			3590E DL T	256kB	80GB	1	1MB	14336	512

Chapter 5 User Utilities

```
10/  5    10  0  0
116 sally: 1w LTO Tape          3580 Gen1  256kB 100GB  1  1MB 15360 1024
10/  5    10  0  0
117 ginger: 1w 3592 Tape (H117)  3592 Tape  256kB 300GB  1  1MB 40960 1024
10/  5    10  0  0
118 gregs lto tape only 118      3580 Gen1  256kB 100GB  1  1MB 15360 1024
10/  5    10  0  0
119 mover3: 1w ampex (H119)      DST 312 Md   1MB 150GB  1  1MB 13312 1024
10/  5    10  0  0
120 mover3: 1w Sony GY-8240 (H120) GY-8240 Lg 256kB 200GB  1  1MB 24576 1024
10/  5    10  0  0
121 1w 9940B (H121)              9940B Tape 256kB 200GB  1  1MB 10240 1024
10/  5    10  0  0
122 fred: 1w STK 9840 tape (H122) 9840 Tape  256kB  20GB  1  1MB 10240 1024
10/  5    10  0  0
123 Util Test (STK)              9840 Tape  256kB  20GB  1  1MB 10240 1024
10/  5    10  0  0
124 1w STK 9940B (H124)          9940A Tape 256kB  60GB  1  1MB 10240 1024
10/  5    10  0  0
125 1w STK 9940B tape (H125)     9940B Tape 256kB 200GB  1  1MB 10240 1024
10/  5    10  0  0
```

```
-----
BBTM=Blocks between tape marks
VVBS=Virtual volume block size
```



Appendix A

Acronyms

ACL	Access Control List
ACLS	Automated Cartridge System Library Software (Science Technology Corporation)
AIX	Advanced Interactive Executive
API	Application Program Interface
CDS	Cell Directory Server
COS	Class of Service
DCE	Distributed Computing Environment
FTP	File Transfer Protocol
GSS	Generic Parallel File System
gid	Group Identifier
GPFS	IBM General Parallel File System
GSS	Generic Security Service
HPSS	High Performance Storage System
HTAR	HPSS Tar
IBM	International Business Machines Corporation
LaRC	Langley Research Center
LANL	Los Alamos National Laboratory
LLNL	Lawrence Livermore National Laboratory
NASA	National Aeronautics and Space Administration
ORNL	Oak Ridge National Laboratory

Appendix A Acronyms

ID	Identifier
IEEE	Institute of Electrical and Electronics Engineers
I/O	Input / Output
IP	Internet Protocol
NFS	Network File System
OSF	Open Software Foundation
PFTP	Parallel File Transfer Protocol
RISC	Reduced Instruction Set Computer
SNL	Sandia National Laboratories
SP	Scalable Processor
TCP	Transmission Control Protocol
uid	User Identifier
VV	Virtual Volume
XDSM	X/Open Data Storage Management
XFS	File system by SGI (not actually an acronym)

References

1. *File Transfer Protocol, RFC-0959*, October 1985.
2. *HPSS Error Messages Manual*.
3. *HPSS Programmer's Reference*.
4. *HPSS Installation Guide*.
5. *HPSS Management Guide*.
6. *HPSS SSM Guide*.
7. *Installing, Managing, and Using the IBM AIX Parallel I/O File System*, Document Number H34-6065-00.
8. *Network File System Specification, RFC-1094*, DDN Network Information Center, SRI International, Menlo Park, Ca.
9. *OSF DCE User's Guide and Reference*, Prentice Hall, Englewood Cliffs, N. J.
10. *POSIX 1003.1-1990 Tar Standard*.

Index

A

ACL editor, 59
allocating space for files, 18

B

block size
 specifying, 43

C

chgid, 14
chgrp, 15
chmod, 15
chown, 16
chuid, 16
class of service (COS)
 concept, 9
class of servicee (COS)
 specifying 14

D

DCE user accounts, 12

F

file family, 10
File Transfer Protocol (FTP), 7, 13

G

group
 changing by ID, 14
 changing by name, 15

H

hacl, 59
HPSS.Conf configuration file, 25
HTAR, 53

I

interfaces
 usage considerations, 10
 user, 7

K

krb5_gss_pftp_client, 21

L

lfappend, 34

lfget, 37

lfput, 36

list directory extensions, 19

list information about HPSS, 68

local file append, 34

local file retrieval, 37

local file store, 36

ls -lh, 19

lshpss, 68

M

mlfget, 40

mlfput, 39

mpget, 33

mpput, 31

multinode

- command, 44

- enable / disable, 44

multiple local file retrieval, 40

multiple local file store, 39

N

Network File System Version 3 (NFS V3), 8

O

owner

- changing by ID, 16

- changing by name, 16

P

pappend, 28

parallel append, 28

parallel file retrieval, 32, 33

parallel file store, 29, 31

Parallel File Transfer Protocol, 21

Parallel FTP (PFTP), 7

permissions

- changing, 15

pftp_client, 21

pget, 32

pput, 29

psocket, 42

Q

quote allo64, 18

S

setcos, 14

setpblocksize, 43

setpwidth, 42

site commands, 13

stage, 17

staging a file, 17

storage class, 10

storage concepts, 9

storage hierarchy, 10

stripe width

 specifying, 42

symbolic link

 creating, 18

symlink, 18

T

TCP socket based transfers

 specifying, 42

U

user IDs, 11

utilities, 9, 59

W

wait for file to be staged, 17

wait, 17

