HSI Best Practices for NERSC Users

Nicholas Balthaser and Damian Hazen
NERSC Division
Lawrence Berkeley National Laboratory
One Cyclotron Road
Berkeley, CA 94720

May 2011

This work was produced by the University of California, Lawrence Berkeley National Laboratory (LBNL) under Contract No. DE-AC02-05CH11231 with the Department of Energy Office of Science.

HSI Best Practices for NERSC Users

Nicholas Balthaser and Damian Hazen
NERSC Division
Lawrence Berkeley National Laboratory
One Cyclotron Road
Berkeley, CA 94720
{nabalthaser,dhazen}@lbl.gov

Abstract

HSI is a command-line interface for data storage and retrieval from HPSS. While HSI is the most frequently used HPSS client at NERSC, until now there has not been a convenient single document explaining how to best use HSI with the NERSC HPSS systems. This can present difficulties for NERSC users new to HPSS, especially since the NERSC HPSS system configuration and authentication mechanism are unique.

In this paper we explain how to obtain and install HSI, create a NERSC authentication token, and transfer data to and from the system. Additionally we describe methods to optimize data transfers and avoid common pitfalls that can degrade data transfers and storage system performance.

1 Introduction

HSI, the Hierarchical Storage Interface, provides a Unix-like command-line interface to the High Performance Storage System (HPSS) at NERSC. HSI supports most of the familiar native Unix FTP commands, as well as providing shell-like enhancements such as command-line editing, command history, and extended *ls* options to display HPSS-specific file attributes. HSI allows recursion for most commands, e.g. *chown* and *chmod*, accepts wildcards such as "*" to perform operations on multiple files, and supports ".netrc" functionality for automated logins. Many shell-like commands such as *mv* and *rm* are implemented as well.

Additional features include support for HPSS parallel I/O transfers, tunable read and write buffers, ability to read commands from a separate text file, accept input from a Unix pipe, and run in a non-interactive mode for use in shell scripts. HSI is supported on many Unix-like platforms including 32 and 64-bit Linux, FreeBSD, AIX, and Mac OS X [1].

HSI is licensed for use and freely downloadable for authorized NERSC users. However it makes use of proprietary HPSS source and is not open source software. It is supported by Gleicher Enterprises with additional site-specific modifications developed by LBNL/NERSC.

1.1 Text Conventions

Italic	is used for Unix filenames, commands, and shell functions. *Italic* is also used for dummy parameters that should be replaced with an actual value.
`Constant Width`	is used in examples to show the contents of files or the output from commands.
`Constant Bold`	is used in examples to show interaction between the user and the shell. Any text the user types is shown in **`Constant Bold`**.
`Constant Italic`	is used in displayed command lines for dummy parameters that should be replaced with an actual value.

2 Installation

Precompiled binary install packages are available on the NERSC software download page, *https://www.nersc.gov/users/data-and-networking/hpss/storing-and-retrieving-data/software-downloads/*. Authentication

with a valid NERSC LDAP username and password is required. Note that HSI is installed and supported on the NERSC compute platforms by the Storage Systems Group. It is only necessary to install HSI to use it on a remote system.

2.1 Selecting the Correct HSI Version

At NERSC the supported version of HSI is matched with the particular release of the HPSS software in use on the storage systems. Older versions of HSI and/or versions available from other HPSS sites are unlikely to be compatible with the storage systems at NERSC.

HSI has been compiled and tested at NERSC only for those operating system versions listed on the software download page. Choose the version that most closely matches your operating system, at least within the major release number. Installing HSI on operating system variants that differ by a minor release number may result in a working configuration, but be aware that the software has not been tested for that version of the OS. Installing HSI on a different major release of an OS is likely to result in a non-working installation. Shared library errors, core dumps, and authentication issues may result.

Contact NERSC Consulting at *consult@nersc.gov* for inquiries regarding HSI versions for operating systems not listed on the software download page.

2.2 Installation Example

In this example HSI 3.4.3 is installed on a RedHat 5.x machine.

1. Check the operating system version.

```
bash-4.0$ cat /etc/redhat-release
Red Hat Enterprise Linux Server release 5.4 (Tikanga)
```

2. Download the install bundle for the operating system version closest to the output of step 1. Check the md5 checksum of the downloaded install bundle to make sure it matches the checksum indicated on the NERSC software download page:

```
bash-4.0$ ls
hsi-3.4.3-rhel-5.3-6-i386.tar.gz
bash-4.0$ md5sum hsi-3.4.3-rhel-5.3-6-i386.tar.gz
0a09ac42a6f8d584e594c67aafaaf944  hsi-3.4.3-rhel-5.3-6-i386.tar.gz
```

3. Untar the install bundle and *cd* to the install subdirectory:

```
bash-4.0$ tar xzf hsi-3.4.3-rhel-5.3-6-i386.tar.gz
bash-4.0$ cd hsi-3.4.3-rhel-5.3-6-i386
```

4. The default install will place the HSI binary, configuration file, and man page into subdirectories of */usr/local* with firewall mode enabled. To perform a default install, run *make* followed by *make install*. Most systems require administrative access to install files in */usr/local*.

```
bash-4.0$ sudo bash
bash-4.0# make
bash-4.0# make install
```

5. Configurable options can be changed prior to installation by running the interactive *Configure* script distributed with the install bundle. A commonly changed configuration option is to change the base installation directory to enable installation as a non-administrative user. It is also common to restrict inbound NERSC data mover connections to a defined port range to facilitate use of host-based firewalls. Note that turning firewall mode off requires allowing inbound connections from NERSC data movers.

```
bash-4.0$ ./Configure
-------------------------------------------------
                      Options
-------------------------------------------------
  0. Firewall Mode.................  on
  1. HPSS Host Name................
  2. Use Port Range................  off
  3. Mover Port Range..............  7500-7599
  4. Base Installation Directory...  /usr/local
-------------------------------------------------


Enter the number of the item to change, or q[uit] > 0
```

To restrict inbound mover connections to a particular port range, select option 2 to enable Port Range and select a port range with option 3. A range of 100 or more ports is recommended. After exiting the *Configure* script run *make* and *make install* as above. Other options are described in the README file included with the install bundle.

2.3 Post-Install Configuration

Further HSI configuration is not typically necessary after running the *./Configure* script. However HSI can be further configured by means of an *.hsirc* file in the user's home directory and/or by setting environment variables by directly editing the */usr/local/bin/hsi* file. This file is a wrapper script used to set the NERSC-specific environment for the *hsi* binary.

The *~/.hsirc* file is used to override configuration set in the global *hsirc* file, installed by default at */usr/local/hpss/etc/hsirc*. The file is divided into global and site-specific stanzas [2]. The typical use would be to enable parallel transfers by turning firewall mode off by default, overriding the setting in the global file:

```
global:
    firewall = off
```

It is not recommended to override NERSC site-specific settings defined in the global *hsirc* file. Further *~/.hsirc* configuration information is available from Gleicher Enterprises at *http://www.mgleicher.us/GEL/hsi/hsirc_file.html*. Note that not all settings in the global *hsirc* file can be overridden.

Direct editing of the HSI wrapper script can be done to change configuration options without re-running *Configure*. However in most cases it is easier and less error-prone to re-run the install with different configuration options. Run *make clean* in the install directory, followed by *./Configure* and *make install* as above.

2.4 Firewall Configuration

To enable parallel transfers with HSI, Firewall Mode should be turned off using either the *Configure* script, the *~/.hsirc* file, or at run-time (see Section 5.1). Running in parallel mode requires that hosts accept incoming connections from NERSC data movers. However many operating systems are configured to deny incoming traffic by default when a host-based firewall is enabled. The standard behavior is for data movers to connect back to the source host on random high-numbered ports. Restricting the port range with the *Configure* script as above is better security practice as it does not require the source host to accept inbound connections on all high-numbered ports.

2.4.1 iptables Example

On CentOS and RedHat Linux the standard firewall is iptables which is configured with the */etc/sysconfig/iptables* file. This file should be configured to grant access for TCP connections originating from the 128.55.80.192/27 and 128.55.128.0/22 networks. A sample configuration to allow such connections is below. Note that wrapped lines are assumed to be continuous:

```
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp -s 128.55.80.192/27
  --dport 7500:7599 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp -s 128.55.128.0/22
  --dport 7500:7599 -j ACCEPT
```

Administrative access is required to edit the file. After editing the ruleset it must be reloaded with "*/etc/init.d/iptables restart*". For configuration information for other host-based firewalls consult your operating system documentation.

3 Authentication

NERSC uses a site-specific token-based authentication mechanism. An encrypted token is generated by the user and placed in an ASCII *.netrc* file. The file is placed in the top level of the user's home directory on the source (client) host. HSI reads the file on startup and the HPSS Gatekeeper validates the token. Assuming validation is successful the user is allowed access to the storage system. This enables authentication without interactively typing a username and password, which is necessary for running HSI in scripts and batch jobs. **Note:** tokens are matched to the user's IP address. A token can only be used within the class C network for which it was generated. For example, if a token is generated for 128.99.1.1, it will work from any system with an IP address of 128.99.1.x.

3.1 Authentication Token Description

The HSI authentication token is a base-64 encoded string which includes the username, token generation timestamp, the IP address of the user's machine, and an encrypted checksum. A version number is prepended to the string, currently "*02*." Token data and login information are looked up in the NERSC LDAP directory and validated by the HPSS Gatekeeper when a user attempts to log into the storage system [3].

3.2 The *.netrc* File

The *.netrc* file is divided into stanzas. Each stanza contains a newline-separated *machine*, *login*, and *password* combination valid for a particular HSI, FTP, or PFTP server. The HPSS *.netrc* parameters are the same as the standard Unix FTP *.netrc* parameters: *machine* is the HPSS server the user logs into, which will be "*archive*" or "*archive.nersc.gov*" for most NERSC users. "*login*" is the user's login name in plain text. "*password*" is the user's encrypted authentication token. Stanzas are separated by a blank line. An example *.netrc* file is below:

```
machine archive.nersc.gov
login joeuser
password 02UPMUezYJ/Urc7ypflk7M8KHLITsoGN6ZIcfOBdBZBxn+BViShg==

machine ftp.nersc.gov
login anonymous
password joeuser@nersc.gov
```

3.2.1 Generation of the *.netrc*

The LBNL/NERSC Storage Systems Group has developed an automated token generation service. Upon initial HSI login to the NERSC HPSS system, if the user has no *.netrc* file or if the *.netrc* does not contain a valid stanza for the HPSS system, the session is redirected to the auth service. The user is prompted for a password, which is the user's NIM/LDAP password. If the password is entered correctly a valid stanza for the user is appended to the *.netrc* file using the IP address and username from the session. The *.netrc* file is created at the top level of the user's home directory if it does not already exist.

Authentication tokens for alternate IP addresses must be generated manually via the *https://nim.nersc.gov/* website. Log into the NIM website and choose "Generate HPSS Token" from the "Actions" dropdown menu. By default the *.netrc* text is generated for the IP address of the HTTP session, i.e. the machine where the user is running the web browser. A link is provided for generating a token using an alternate IP address. Using a text editor, copy and paste the *.netrc* text from the

token generator into a *.netrc* file at the home directory top level.

For more token generation information and examples, see *http://www.nersc.gov/users/data-and-networking/hpss/getting-started/hpss-passwords/*.

3.2.2 *.netrc* File Permissions

The *.netrc* file must be read/write for the user only. On a Unix/Linux system this is accomplished with:

```
bash-4.0$ chmod 600 .netrc
```

Incorrect file permissions will invalidate the file, resulting in failed logins.

4 Transferring Data

HSI has an interactive mode similar to command-line FTP. HSI also includes support for several non-interactive transfer modes, multiple simultaneous storage system connections, and HPSS-to-HPSS transfers.

4.1 Connecting to NERSC HPSS Systems

Assuming the *.netrc* is set up correctly, simply typing *hsi* from any NERSC compute platform will connect to the *archive* system. The *archive* system is the designated HPSS system for user data.

```
bash-4.0$ hsi
[Authenticating]
A:/home/j/joeuser->
```

From offsite use "*-h*" to specify a hostname:

```
bash-4.0$ hsi -h archive.nersc.gov
[Authenticating]
A:/home/j/joeuser->
```

4.2 Interactive Transfers

The interactive command-line interface supports many of the same commands as FTP, as well as recursive operations and command-line editing. Note that there is no need to specify *binary* mode transfers in HSI. For a summary of HSI commands type *help* at the HSI prompt or see the NERSC HSI Usage page at: *http://www.nersc.gov/users/data-and-networking/hpss/storing-and-retrieving-data/clients/hsi/*.

The simplest use case is to transfer a file between the current directory on a source host and the current directory on the storage system. To transfer a file to the storage system the syntax is *put* <*filename*>:

```
A:/home/j/joeuser-> put myfile
put  'myfile' : '/home/j/joeuser/myfile' ( 2097152 bytes, 31445.8 KBS
(cos=4))
```

To retrieve the file the syntax is simply *get* <*filename*>.

```
A:/home/j/joeuser-> get myfile
get  'myfile' : '/home/j/joeuser/myfile' (2010/12/19 10:26:49 2097152 bytes,
46436.2 KBS )
```

To specify full pathnames, rename files during transfers, or otherwise differentiate between local files and storage system files, HSI uses a "*:*" (colon character). The character must be surrounded by whitespace. The syntax is: "(*put*|*get)*

local_file : hpss_file". Note that unlike FTP, the local filename is always placed on the left side of the colon character regardless of the direction of the transfer:

```
A:/home/j/joeuser-> put local_file : hpss_file
A:/home/j/joeuser-> get local_file : hpss_file
```

Wildcards, e.g. '*' and '?', are supported for transfer operations on pattern matches:

```
A:/home/j/joeuser-> put .bashr?
put  '.bashrc' : '/home/j/joeuser/.bashrc' ( 3607 bytes, 131.5 KBS (cos=4))
```

Wildcards can be useful for transferring multiple files that match a certain pattern using *mput* and *mget*:

```
A:/home/j/joeuser-> prompt
prompting turned off

A:/home/j/joeuser-> mput .bash*
mput  '.bash_history' : '/home/j/joeuser/.bash_history' ( 12718 bytes, 578.8
KBS (cos=4))
mput  '.bash_profile' : '/home/j/joeuser/.bash_profile' ( 2100 bytes, 76.5
KBS (cos=4))
mput  '.bash_profile.ext' : '/home/j/joeuser/.bash_profile.ext' ( 523 bytes,
24.0 KBS (cos=4))
mput  '.bashrc' : '/home/j/joeuser/.bashrc' ( 3607 bytes, 182.0 KBS (cos=4))
mput  '.bashrc.ext' : '/home/j/joeuser/.bashrc.ext' ( 711 bytes, 33.9 KBS
(cos=4))
```

4.2.1 Recursive Operations

Recursive operations are supported using "*-R*" for many HSI commands, including:

```
cget, chgrp, chmod, chown, cput, delete, get, ls,
mdelete, mget, mput, put, rm, stage, touch
```

For example, the following command allows group read access for all files below the user's *tmp* directory in the storage system:

```
A:/home/j/joeuser->  chmod -R g+r ./tmp/
```

Recursive *put* and *mput* are not recommended for reasons explained in Section 6.

4.2.2 History and Command-Line Editing

HSI supports the *history* command to view previous commands:

```
A:/home/j/joeuser-> history
   2 get myfile
   3 put .bashr?
   4 prompt
   5 mput bash*
   6 chmod -R g+r ./tmp/
```

Up-arrow and down-arrow keys can be used to scroll through the command history.

Command-line editing is supported in either *emacs* or *vi* mode. The default, if not specified in *$HOME/.editrc*, is *vi* mode. To specify an edit mode at runtime use the "*-E <editor>*" argument:

```
bash-4.0$ hsi -E emacs
```

For more information about the *.editrc* file see: *http://www.mgleicher.us/GEL/hsi/editrc_man_page.html* or the *editrc* man page distributed with HSI.

4.2.3 Running Local Commands from HSI

Operations on the local filesystem are supported using the following commands, with output occurring in the HSI session:

```
lcd, lls, lmkdir, lpwd
```

The "*!*" (exclamation point) can be used to perform shell operations on the local system. Text following the exclamation point is executed in the shell on the local system, with output occurring in the HSI session:

```
A:/home/j/joeuser-> !cd && pwd
/global/homes/j/joeuser
```

4.2.4 Sleepers

"Sleepers" are a NERSC-specific mechanism for limiting access to the storage systems during downtime. When sleepers are set on the NERSC compute platforms HSI and PFTP jobs attempting to access storage will wait. Usually jobs resume safely after sleepers are removed [4]. Interactive access will display an "HPSS disabled" message when sleepers are set:

```
bash-4.0$ hsi
*** HSI - Waiting for <HPSS disabled> flag to clear ***
```

Note that the sleepers mechanism is only available on the NERSC compute platforms. It does not work on remote clients.

The *hpss_avail* utility on the NERSC compute platforms can be used to check sleepers status. *hpss_avail* takes a storage system name as an argument and returns either *1* if sleepers are set, i.e. system unavailable, or *0* if sleepers are not set.

```
bash-4.0$ hpss_avail archive; echo $?
0
```

For more information type "*hpss avail help*" on the command line, or see the NERSC sleepers page at: *http://www.nersc.gov/users/data-and-networking/hpss/storing-and-retrieving-data/sleepers/*.

4.3 Non-Interactive Transfers

HSI supports four non-interactive modes for ease of use in shell scripts and batch jobs: one-line mode, command file, here document, and reading from standard input

4.3.1 One-Line Mode

Multiple commands are given to HSI on a single line, separated by the *";"* (semicolon) character. HSI executes the commands serially and exits. Note that the shell may impose a line length limitation.

```
bash-4.0$ hsi -q "mkdir mydir; cd mydir; put myfile; ls -l"
[Authenticating]
mkdir: /home/j/joeuser/mydir
put  'myfile' : '/home/j/joeuser/mydir/myfile' ( 2097152 bytes, 28748.6 KBS
(cos=4))
/home/j/joeuser/mydir:
-rw-------   1 joeuser     joeuser        2097152 Dec 19 14:51 myfile
```

4.3.2 Command File

HSI executes commands specified in a separate command file with one command per line. The syntax is *hsi "in command_file"*:

```
bash-4.0$ cat mycommands.txt
mkdir mydir
cd mydir
put myfile
ls -l
quit

bash-4.0$ hsi "in mycommands.txt"
[Authenticating]
mkdir mydir
mkdir: /home/j/joeuser/mydir
cd mydir
put myfile
put  'myfile' : '/home/j/joeuser/mydir/myfile' ( 2097152 bytes, 39813.8 KBS
(cos=4))
ls -l
/home/j/joeuser/mydir:
-rw-------   1 joeuser     joeuser        2097152 Dec 19 14:56 myfile
quit
```

4.3.3 Here Document

A "here document" is a type of shell redirection which instructs the shell to read input from the current source until a line containing a specified delimiter with no other characters or whitespace is seen. All lines read up to that point are then used as standard input for the initial command:

```
bash-4.0$ hsi <<EOF
mkdir mydir
cd mydir
put myfile
ls -l
quit
EOF
[Authenticating]
mkdir: /home/j/joeuser/mydir
put  'myfile' : '/home/j/joeuser/mydir/myfile' ( 2097152 bytes, 19360.0 KBS
(cos=4))
/home/j/joeuser/mydir:
-rw-------   1 joeuser     joeuser        2097152 Dec 19 15:08 myfile
```

4.3.4 Standard Input

In a manner similar to the "here document" above, HSI can accept commands from standard input using a *"|"* (pipe) character:

```
bash-4.0$ echo 'mkdir mydir; cd mydir; put myfile; ls -l; quit' | hsi -q
[Authenticating]
mkdir: /home/j/joeuser/mydir
put  'myfile' : '/home/j/joeuser/mydir/myfile' ( 2097152 bytes, 24626.0 KBS
(cos=4))
/home/j/joeuser/mydir:
-rw-------   1 joeuser     joeuser        2097152 Dec 19 15:15 myfile
```

4.4 Multiple Connections and HPSS to HPSS Transfers

HSI has the capability to establish concurrent connections to multiple HPSS systems within a single HSI session. Note that at NERSC this is not currently a supported option and the following example is for illustrative purposes only. Please contact NERSC Consulting at *consult@nersc.gov* before attempting HPSS-to-HPSS transfers.

At sites that support 3rd-party transfers, files can be copied directly from one HPSS system to another without first staging the data at the intermediate HSI client system. To make use of HPSS-to-HPSS transfers the user must have an account on both HPSS systems, the HSI client system must be able to connect to both HPSS systems, and there must be network connectivity between the mover nodes on the source HPSS system and the HSI Gateway server process on the destination system [5]. HPSS-to-HPSS transfers are highly dependent on compatible software levels, network configuration, and authentication mechanisms between participating sites.

When working with multiple HPSS systems HSI treats each connection as a "logical drive" or drive letter. Each connection has a separate context associated with the drive letter. It is possible to switch between the active connection by specifying its drive letter.

Assuming authentication has been set up for the second HPSS system, to establish a concurrent connection to the second system use the *connect* command from the initial HSI session. In this example *joeuser* connects first to the NERSC *archive* system, then establishes a concurrent connection on drive letter *H:* to the *regent* (*hpss*) system. The password is the encrypted string from the user's *.netrc* file:

```
bash-4.0$ hsi -q
[Authenticating]
A:/home/j/joeuser-> connect -q -l joeuser -s hpss
[joeuser]Password:
H:/home/j/joeuser->
```

After the second connection is established the active connection can be switched back to the *archive* system by specifying drive letter "*a:*":

```
H:/home/j/joeuser-> a:
A:/home/j/joeuser->
```

Active connections can be listed with "*lscon*". Files can be moved from one system to another with "*cp*". In this example "*/home/j/joeuser/mydir/myfile*" is copied from the *archive* system to the "*/home/j/joeuser/myhpssdir*" directory on the *regent* (*hpss*) system:

```
A:/home/j/joeuser-> h:
H:/home/j/joeuser-> mkdir myhpssdir
mkdir: /home/j/joeuser/myhpssdir
H:/home/j/joeuser-> a:
A:/home/j/joeuser-> cd mydir
A:/home/j/joeuser/mydir-> ls -l
/home/j/joeuser/mydir:
-rw-------   1 joeuser     joeuser        2097152 Jan  1 12:41 myfile
A:/home/j/joeuser/mydir-> cp ./myfile h:/home/j/joeuser/myhpssdir/
cp  'A:/home/j/joeuser/mydir/myfile' to 'H:/home/j/joeuser/myhpssdir/myfile'
(2011/0                              1/01 12:41:39 2097152 bytes, 16007.8 KBS
(cos=4))
A:/home/j/joeuser/mydir-> h:
H:/home/j/joeuser-> ls -l myhpssdir/

/home/j/joeuser/myhpssdir:
-rw-------   1 joeuser     joeuser        2097152 Jan  1 12:48 myfile
```

The second session can be disconnected with "*close*":

```
H:/home/j/joeuser-> a:
A:/home/j/joeuser-> close h:
A:/home/j/joeuser->
```

Further information about using HSI with multiple HPSS systems is available from Gleicher Enterprises at *http://www.mgleicher.us/GEL/hsi/hsi_reference_manual_2/using_hsi_with_multiple_hps.html*. Options documented by Gleicher Enterprises may not be supported on the NERSC HPSS systems.

5 Optimization

The single-file transfers in the Section 4 examples above should require no further user optimization when initiated from NERSC compute platforms. The NERSC Storage Systems Group configures HSI to handle common usage at the NERSC site. However more in-depth knowledge about the NERSC tape archives, HSI usage, and other data transfer infrastructure may be required to improve transfer performance for certain use cases. Such use cases include retrieving files distributed across multiple tapes, WAN transfers, and situations where pre-staging files onto disk cache may be a consideration.

5.1 Parallel Mode Transfers

On high-speed networks use of parallel transfer mode increases available transfer bandwidth. Parallel transfer mode is enabled by default on the NERSC compute platforms, however it disabled by default when HSI is installed from software packages distributed on the NERSC software download page. For most networks the default settings are appropriate, however for clients within the NERSC network or on other high-speed networks transfer performance may improve by enabling parallel mode transfers. At run-time this is done by turning off firewall mode:

```
bash-4.0$ hsi 'firewall -off'
A: firewall mode set OFF, I/O mode set to normal (parallel=on),
autoscheduling currently set to OFF
```

Firewall mode can also be disabled during installation or by editing the ~/*.hsirc* file. See Sections 2.2 and 2.3.

5.2 Tape and File Ordering

When retrieving large numbers of files distributed across numerous tapes, inefficient use of the system can have a detrimental effect on I/O performance as well as putting undue wear on tape drives and robotics, i.e. "thrashing." It can also have a negative impact on system performance for other archive users and cause long waits for data. It is best for both transfer rates and data availability to organize such requests in a way that minimizes both tape mounts and repositioning on individual tapes. This can be accomplished by gathering tape volume and position information for each file, then organizing transfers based on this information [6]. The following example illustrates a manual method for ordering file requests by tape volume. To help automate this process a sample Perl script is available on the NERSC web site at *http://www.nersc.gov/users/data-and-networking/hpss/usage-examples/hsi-tape-ordering-script/* .

The HSI "*ls -X*" or *"ls -P*" options can be used to list extended attributes for files in the archive, including tape volume and position:

```
bash-4.0$ hsi -q 'ls -X /home/j/joeuser/mydir/myfile'
/home/j/joeuser/mydir:
-rw-r-----   1 joeuser     joeuser         5        12345 TAPE    1048576000
Jun 11  2010 myfile
Storage   VV   Stripe
 Level   Count  Width  Bytes at Level
----------------------------------------------------------------------------
 0 (disk)   0       4                                    (no data at this level)
 1 (tape)   1       1  1048576000
  VV[ 0]:   Object ID: 8a7c31ea-694c-11df-b691-0004ac493f6a
            ServerDep: e65fcc4a-7833-11d2-a346-0004ac493f6a
  Pos:    42  PV List: EM318000
 2 (tape)   0       0                                    (no data at this level)
 3 (tape)   0       0                                    (no data at this level)
 4 (tape)   0       0                                    (no data at this level)
```

The first line indicates that the file is not resident on disk, i.e. "*TAPE*." This is also indicated in the line starting with "*0 (disk) 0*". The other relevant information is that *myfile* is at position 42 on tape volume EM318000.

The "*ls -P*" option gives a more concise summary (note, wrapped lines are continuous):

```
bash-4.0$ hsi -q 'ls -P /home/j/joeuser/mydir/myfile'
FILE    /home/j/joeuser/mydir/myfile        1048576000      1048576000
  4       EM318000        5       0       1       06/11/2010      15:00:17
  01/02/2011      20:12:54
```

In the listing above, field 2 is the filename, field 5 is tape position, and field 6 is the volume name.

Volume and position information is obtained for each file to be retrieved. Then transfers can be grouped on a per-tape basis in ascending position order, reducing excessive tape mounts and repositioning. For large numbers of files this can be time consuming, however it can be scripted as in the following examples.

If all files to be retrieved are located conveniently under one directory, a list of filenames, tape positions, and volumes can be obtained with a single "*ls -P*" command, such as:

```
bash-4.0$ hsi -q 'ls -P /home/j/joeuser/mydir/' 2>&1 | grep FILE \
  | awk '{printf("%s  %10s    %s\n",$2,$5,substr($6,0,6))}' \
  > pv.list
```

"*pv.list*" will resemble the following:

```
bash-4.0$ cat pv.list
/home/j/joeuser/mydir/myfile00        3536    EA8541
/home/j/joeuser/mydir/myfile01           1    EM4502
/home/j/joeuser/mydir/myfile02           3    EM4502
...
/home/j/joeuser/mydir/myfile10         176    EA1445
```

If the files are not located under a single directory, put the filenames to be retrieved into a text file list:

```
bash-4.0$ cat files.txt
/home/j/joeuser/mydir/myfile00
/home/j/joeuser/mydir1/myfile01
/home/j/joeuser/mydir2/myfile02
...
/home/j/joeuser/mydir10/myfile10
```

The list of files, tape positions, and volumes can be generated and placed in "*pv.list*" by running "*ls -X*" in a loop similar to the following:

```
bash-4.0$ for file in `cat files.txt`
> do
> echo -n "$file" >> pv.list
> hsi -q "ls -X $file" 2>&1 | grep 'PV List' >> pv.list
> done
```

Note that standard output and standard error are combined because HSI often uses standard error for its output. "*pv.list*" should look like the following example:

```
bash-4.0$ cat pv.list
/home/j/joeuser/mydir/myfile00  Pos: 3536   PV List: EA854100
  Pos:140790   PV List: ED000100
/home/j/joeuser/mydir1/myfile01  Pos:   1   PV List: EM450200
/home/j/joeuser/mydir2/myfile02  Pos:   3   PV List: EM450200
...
/home/j/joeuser/mydir10/myfile10  Pos: 176   PV List: EA144500
  Pos:90603   PV List: ED000400
```

Lines containing volumes beginning with "*ED*" should be removed as these are dual-copy tapes. Next, per-tape lists can be

generated in position order using "*pv.list*" as input:

```
bash-4.0$ for vol in `awk '{print $6}' pv.list | sort -u`
> do
> grep $vol pv.list | sort -n +2 -3 | awk '{print $1}' > ${vol}.list
> done
```

Note that the fields in "*pv.list*," and thus the command line above, will be slightly different if the list was generated with "*ls -P*" as in the first example.

The command line above will generate file lists called "<*volume*>.*list*" in position order similar to the following:

```
bash-4.0$ cat EM450200.list
/home/j/joeuser/mydir1/myfile01
/home/j/joeuser/mydir2/myfile02
/home/j/joeuser/mydir3/myfile03
...
```

File lists can then be converted to HSI command files (See Section 4.3.2) and submitted to HSI for per-volume file retrieval in position order:

```
bash-4.0$ cat EM450200.cmd
get /home/j/joeuser/mydir1/myfile01
get /home/j/joeuser/mydir2/myfile02
get /home/j/joeuser/mydir3/myfile03
quit
bash-4.0$ hsi -q "in EM450200.cmd"
```

While it is straightforward to organize file retrieval on a per-volume basis as above, it is more efficient to store and retrieve large numbers of files by aggregating with HTAR. HTAR file aggregates are usually stored on contiguous volumes. HTAR also generates an index file for faster retrieval. More information about HTAR is available on the NERSC website at *http://www.nersc.gov/users/data-and-networking/hpss/storing-and-retrieving-data/clients/htar/*.

5.3 WAN Transfers

HSI is tunable for WAN transfers on a per-destination basis in the "*Network Options*" section of the *HPSS.conf* file. Note that this is not a user-configurable file on the NERSC compute platforms. Furthermore the HPSS mover protocol is not optimized for WAN transfers, and WAN transfer tuning can be a time-intensive iterative process which is beyond the scope of this paper. At this time we recommend that NERSC users perform WAN transfers using run-time configurable tools such as *bbcp* or *globus-url-copy* from the NERSC Data Transfer (DTN) nodes to the destination site. Files would first be transferred from the HPSS archives to global */scratch* or */project* filesystems on the DTN nodes using HSI. For example, to copy *myfile* from a NERSC DTN global */scratch* to an ORNL DTN using *bbcp* with 8 network streams and a 1024Kb window size, the command is:

```
bash-4.0$ cd $SCRATCH
bash-4.0$ /usr/common/usg/bin/bbcp -v -s 8 -w 1024k \
   -T "ssh -x -a -oFallBackToRsh=no %I -l %U %H /opt/public/bin/bbcp" \
   ./myfile dtn01.ccs.ornl.gov:/tmp/work/joeuser/
```

Further information about the NERSC Data Transfer nodes is available for NERSC users at *http://www.nersc.gov/systems/data-transfer-nodes/*. More information about *bbcp* is available at *http://www.nersc.gov/users/data-and-networking/transferring-data/bbcp/*. Additionally there is an ESnet knowledge base site dedicated to WAN data transfer tuning at *http://fasterdata.es.net/*.

5.4 Staging

"*Staging*" refers to copying files from relatively slow tape media to the HPSS disk cache for faster access. With data pre-staged to disk, it will presumably be available for immediate use by jobs running on the compute platforms, avoiding waits

for tape mounts and positioning. For small amounts of data or small files staging may work as intended, however for large amounts of data staging may incur a performance penalty.

The HPSS disk cache is used for all file access in the storage system, i.e. both storage and retrieval. It is a shared resource of finite size, and it is not possible to allocate space on a per-user basis. When the cache is full the oldest files are purged. Due to the size of the cache and the fact that it is used concurrently by many users, when staging large amounts of data the cache may purge the data before it is needed by the application. This is especially likely during periods of high system activity, or if the total amount of data to be staged is larger than the cache. When this happens the application will have to wait for the data to be read from tape again, causing further delays [7].

A better approach would be to retrieve the data to the NERSC Global Filesystem (NGF), either global */scratch* or */project*, before the application runs on the compute system. Global */scratch* is available on most NERSC compute platforms and users can request temporary quota increases that should be sufficient for most applications. See the NERSC Filesystem page at *http://www.nersc.gov/users/data-and-networking/file-systems/* for more information.

6 Common Issues

There are a number of common network-related performance and configuration issues that can occur when connecting to the NERSC HPSS systems from offsite, including problems induced by firewalls or NAT gateways at the remote site. Other common performance issues can be related to HSI and/or storage system use cases such as long-running transfers, streaming data via Unix pipelines, or data sets involving massive numbers of small files. Many of these types of issues have workarounds involving alternate methods of using HSI.

6.1 Networking

Offsite networks commonly implement firewalls or NAT gateways which can inadvertently block or degrade transfer performance. Other common issues include multi-homed hosts and latency or network bottlenecks. Note that firewall mode is the default setting for downloaded clients, so explicitly enabling firewall mode should not be necessary in most cases.

6.1.1 Firewalls

A firewall blocking an HSI transfer often appears to be a working connection in which control commands (e.g. "*ls*", "*cd*", etc.) function but data transfers hang. To enable parallel transfers with HSI, remote hosts must accept incoming connections from HPSS data movers (see Section 2.4, "Firewall Configuration"). If opening a port range for HPSS mover traffic is not possible, HSI "firewall" mode can be enabled with the "*firewall -on*" command as in the example below. Firewall mode uses a “store-and-forward” I/O mode instead of the normal mode in which HPSS movers connect back to the HSI client [8]. On the NERSC HPSS systems firewall mode will restrict the flow of data traffic between the HSI client machine and the storage system User Interface machine on a high-numbered port. For large files firewall mode may incur a performance penalty since parallel transfers are disabled in this mode.

```
bash-4.0$ hsi 'firewall -on; get /dev/null : /home/j/joeuser/myfile'
A: firewall mode set ON, I/O mode set to extended (parallel=off),
autoscheduling currently set to OFF
get  '/dev/null' : '/home/j/joeuser/myfile' (2009/09/22 11:15:18 3236744047
bytes, 53656.9 KBS )
```

6.1.2 Muti-homed Hosts

A "multi-homed" host is a host with two or more network interfaces configured. When HSI firewall mode is set to "*-off*", HPSS movers will attempt to connect to the client machine on the interface bound to the name returned by the "*hostname*" command. If the client machine's hostname is bound to a non-routable IP address, e.g. 10.x.x.x or 192.168.x.x, parallel transfers will fail or perform badly. On Unix-like machines hostname-to-IP-address mapping can usually be checked with the "*hostname*" command and the */etc/hosts* file:

```
bash-4.0$ hostname
myhost.berkeley.edu
bash-4.0$ grep myhost /etc/hosts
192.168.10.101         myhost.berkeley.edu      myhost
```

In this example the hostname *myhost* is bound to a non-routable IP address so parallel transfers will not work correctly. This behavior can be overridden at install time by setting Option 1 in the *Configure* script, "*HPSS Host Name*", to the machine's routable hostname or IP address. After installation this behavior can be overridden by directly editing the *hsi* script to set the "*OVERRIDE_HPSS_HOSTNAME*" variable to the appropriate routable hostname or IP address. If the multi-homed host does not have a routable IP address the only option is to use firewall mode.

6.1.3 NAT Gateways

"NAT" stands for "Network Address Translation," which is an IP masquerading technique often used to route traffic between a non-routable internal network and external sites or the Internet. In a typical NAT configuration all traffic from the non-routable network is routed through a NAT gateway machine, which dynamically readdresses outgoing IP packets so they appear to originate from a single IP address on the gateway.

In the typical NAT configuration described above it is most often the case that connections from external hosts cannot be initiated to hosts on the non-routable network. Therefore parallel HSI transfers will not work in such a NAT environment because the HPSS movers will not be able to connect back to the HSI client machine on the internal network. HSI users in a NAT environment must set firewall mode to "*-on*", disabling parallel transfers.

6.1.4 Network Performance

The NERSC Networking and Storage Systems Groups work to optimize network performance between the storage systems and compute platforms. Network performance issues on the NERSC internal network should be reported to *consult@nersc.gov*.

Troubleshooting network issues occurring between offsite machines and the NERSC storage systems can be complicated, involving many potential bottlenecks and network hops. While such troubleshooting is beyond the scope of this paper, online resources are available. ESnet offers a Knowledge Base and live I/O test hosts available for use in determining the location(s) of network performance issues. The Network Troubleshooting Overview page is located at *http://fasterdata.es.net/ts_overview.html*, and an explanation and usage examples for the I/O test hosts are located at *http://fasterdata.es.net/disk_pt.html*.

6.2 Storage System Usage

Many performance issues can result from non-optimal use of storage system resources. Tape-backed storage systems generally do not work well with large numbers of small files, due in part to the sequential nature of the storage media. Retrieving such data sets can induce latency due to time spent mounting and positioning tapes. Additionally HPSS-specific storage concepts such as "Class of Service" may inadvertently cause issues if not used appropriately.

6.2.1 HPSS Classes of Service (COS) and Unix Pipelines

An HPSS COS is an abstraction of storage system characteristics that allows users to select a type of service based on performance, space, and functionality requirements [10]. At NERSC COS is used to determine optimal storage resources, i.e. type of disk and tape media, for files of various sizes. COS determination is mostly transparent to storage system users and is based on the size of file(s) being stored.

In the NERSC environment, in order to determine COS, and thus the optimal type of disk and tape media, the HPSS software must be able to determine the size of the file(s) being stored before being allocated in the system. This can be a problem when data is written to HSI or other storage system clients via a Unix pipeline (pipe). While streaming data via

pipes mitigates the need for large amounts of free disk space to stage data, it does not allow HSI to determine the size of the data being stored. Lacking file size information, HPSS will allocate data to a default COS, which may not have adequate resources for storing the amount of data being transferred. The data could be allocated in such a way that it is very slow to retrieve, and/or the transfer may fail after it is partially completed.

There are several workarounds to avoid the use of Unix pipelines with HSI, although all involve workflow changes. Pipelines are most often used to accept the standard output of filesystem backup commands such as *tar*, *dump*, or *cpio*. The easiest workaround is to write the backup archive to local disk or NGF filesystems (e.g. global */scratch* or */project*) before transferring it to permanent storage with HSI. As this may not be an option for some users the Storage Systems Group makes HTAR available. HTAR stores or retrieves a tar-compatible archive in HPSS without first storing it on local disk, while handling file size information for allocation in the appropriate COS. HTAR is the recommended workaround if there is not enough filesystem space available for local staging.

If neither alternative will work a third option is to write the existing pipeline to FTP or PFTP, using the HPSS *ALLO64* command to pass file size information before the data is stored (see example below). This allows data to be stored without first staging to local disk, however the user must provide a method to accurately predetermine the number of bytes to be transferred to the storage system. Additionally, troubleshooting transfer issues involving Unix pipelines can be difficult, as HPSS will not be able to log any information about the command writing to the pipe. Note that this option is not currently supported on the NERSC *franklin* or *hopper* compute platforms.

```
bash-4.0$ pwd
/global/homes/j
bash-4.0$ du -sk ./joeuser
7526124 ./joeuser
bash-4.0$ echo "7526124 * 1024" | bc
7706750976
bash-4.0$ /usr/bin/ftp -n -v archive <<EOF
> user joeuser <crypted string>
> bin
> prompt off
> quote allo64 7706750976
> put "| tar cf - ./joeuser" /home/j/joeuser/joeuser.tar
> quit
> EOF
bash-4.0$
```

6.2.2 Long-running Transfers

Long-running transfers can be failure-prone for reasons such as transient network issues, scheduled system maintenance, unscheduled downtime, etc. HSI does not have the capability to resume interrupted transfers, so any interruption will result in having to start the transfer over. Additionally, data cannot be completely migrated to tape and purged from the disk cache until the transfer is completed, which may cause administrative problems with HPSS. It is recommended to break transfers into groups that can be completed in 24 hours or less, if possible.

6.2.3 Library and Tape Subsystem Issues

There are several use cases that can put an undue load on the tape subsystem, causing performance issues and possible delays for all storage system users. Generally these issues can be mitigated by combining small files into larger archive files (aggregates) that are more efficient to store and retrieve from tape media.

One of the worst ways to store data in HPSS is to recursively copy a large filesystem directory tree with HSI. While HSI permits this and there will be no apparent issues with the transfer or data storage, the thousands of small files will be scattered across dozens of tapes as they are migrated to tape from disk cache at periodic intervals. Even if tape and file ordering are done properly on retrieval (Section 5.1), retrieving thousands of files stored in this manner will cause a lot of tape mount and positioning activity. As tape drives are allocated on a first-come, first-served basis, this type of retrieval workload will not only be slow to process, it will likely cause delays for all storage system users by allocating most of the available tape drives. Excessive tape mount activity can also put undue wear on library components such as robotics and pass-through-ports.

Users with this type of small-file workload often attempt to "pre-stage" the data to disk cache prior to running a compute job in order to mitigate delays caused by tape mounts. Pre-staging attempts to read all the data back from tape to disk cache in order to make it available quickly for an application running on one of the compute platforms. For reasons described in Section 5.3, pre-staging is a failure-prone method of optimizing file retrieval. Not only will it cause excessive mount activity and poor system performance as described above, it is likely that some of the staged data will be purged from the cache before being used by the application, causing additional delays as it is re-read from tape.

Both of the above use cases can be mitigated by combining small files into larger aggregates with utilities such as HTAR, *tar*, *cpio*, or *dump*. Currently the largest cartridge size is 1TB, so it is possible that aggregates approaching this size will fit entirely on a single tape. Additionally HTAR generates a file index for fast retrieval.

In the event that thousands of small files were stored non-optimally and must be retrieved from tape, use of the tape and file ordering method described in Section 5.1 is the best retrieval method. As a courtesy, prior to starting such a workload on the storage system please notify the Storage Systems Group by contacting *consult@nersc.gov*. We may be able to help optimize the transfer.

6.3 Other Issues

For various reasons related to HPSS configuration, internals, or HSI development issues, HSI users may run into several other issues explained in this section.

6.3.1 HPSS Session Limits

By default the HPSS Gatekeeper is configured to allow a maximum of 15 concurrent sessions per user. This number can be administratively decreased in the event user activity is causing system problems. Please contact NERSC Consulting if more than 15 concurrent sessions for a single user or login are needed.

6.3.2 Large Directory Listing

Each HPSS system is backed by a single database instance that holds all information about stored files (metadata). Every user interaction with the system causes some amount of database query activity. One of the most database-intensive HSI commands is long file listing, i.e. "*ls -l*". Directories containing more than a few thousand files can become difficult to work with due to the amount of database activity incurred. Performing a long listing of a directory with 24,000 files takes over 2 minutes in this example:

```
bash-4.0$ time hsi -q 'ls -l /home/n/nickb/24k-files/' > /dev/null 2>&1

real    2m28.346s
user    0m6.692s
sys     0m0.940s
```

For ease of system use we suggest maintaining smaller directories if possible, and combining large numbers of files into aggregates as explained in Section 6.2.3.

6.3.3 File Checksums

HSI does not have the capability to perform file checksum verification at this time although this feature has been proposed. Users requiring checksum verification will have to perform this step manually with a separate utility such as *sha1sum* or *md5sum*, then store a list of checksums with the corresponding filenames.

HTAR provides a Cyclic Redundancy Checksum (CRC) option that can be used during archive creation to verify the member files of an archive.

7 Reporting Problems

To report a issues with HSI, the NERSC HPSS systems, or for further assistance please contact NERSC Consulting at 510-486-8611, toll-free at 800-666-3772 #3, or send email to *consult@nersc.gov*.

To report errors, omissions, or issues with the accuracy of this document please send email to *nabalthaser@lbl.gov*.

8 Appendix

The following materials are provided for reference.

A: HSI TCP Conversation Diagrams

The following TCP port diagrams illustrate HSI transfers in both firewall and normal modes.

In Figure 1 a file is downloaded in firewall mode. The client connection is initiated on a random high-numbered port (51100) communicating with the HPSS Gateway Daemon on port 7597. The data connection occurs between port 33794 on the Gateway server and port 44909 on the client system. Data is forwarded from the movers to the Gateway server.

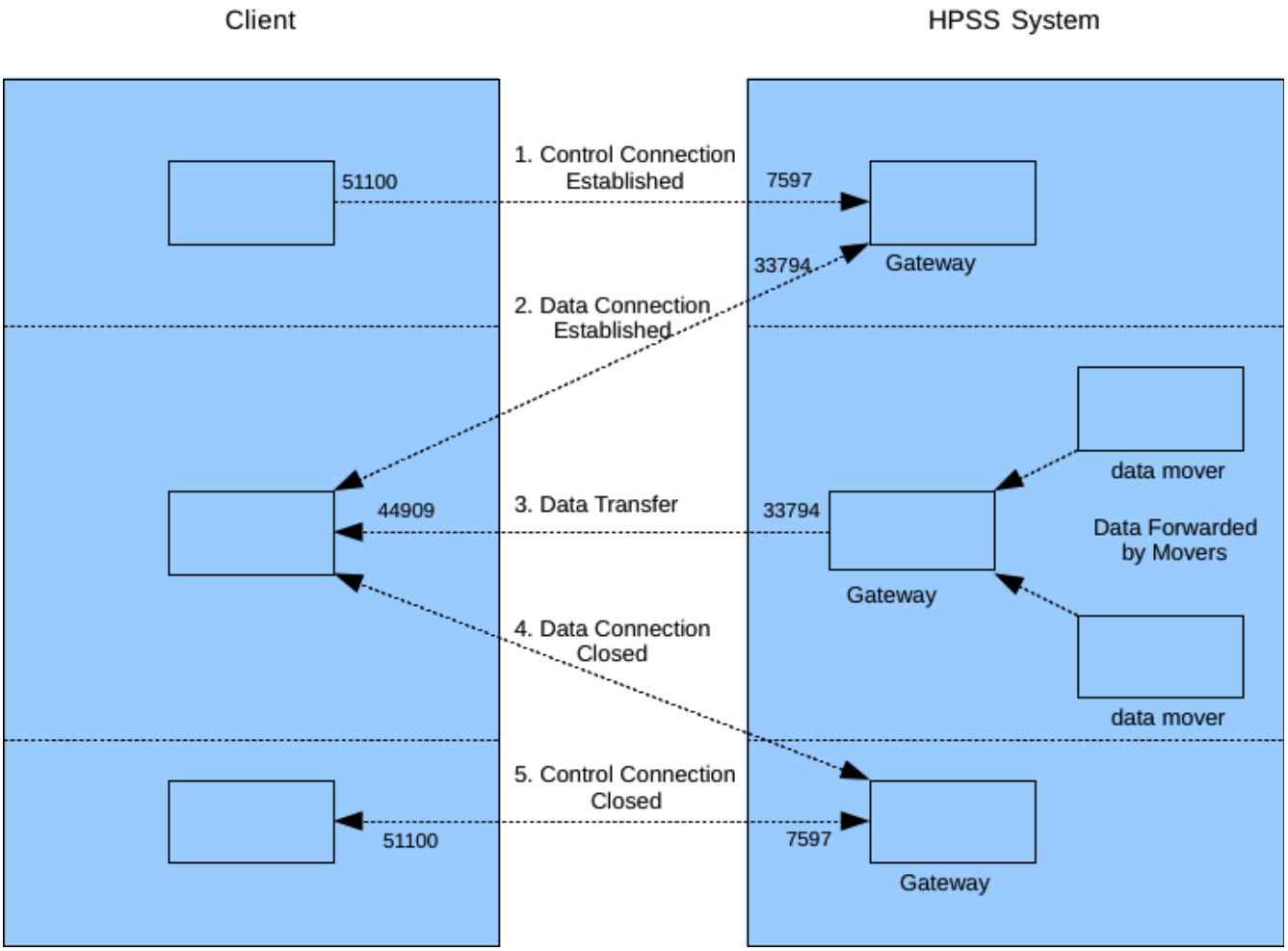

Figure 1: Firewall Mode Data Transfer

In Figure 2 a file is downloaded in normal mode. The client connection is initiated on a random high-numbered port (51632) communicating with the HPSS Gateway Daemon on port 7597. After the control connection is established, HPSS data movers connect back to the client and transfer data in parallel on high-numbered ports.

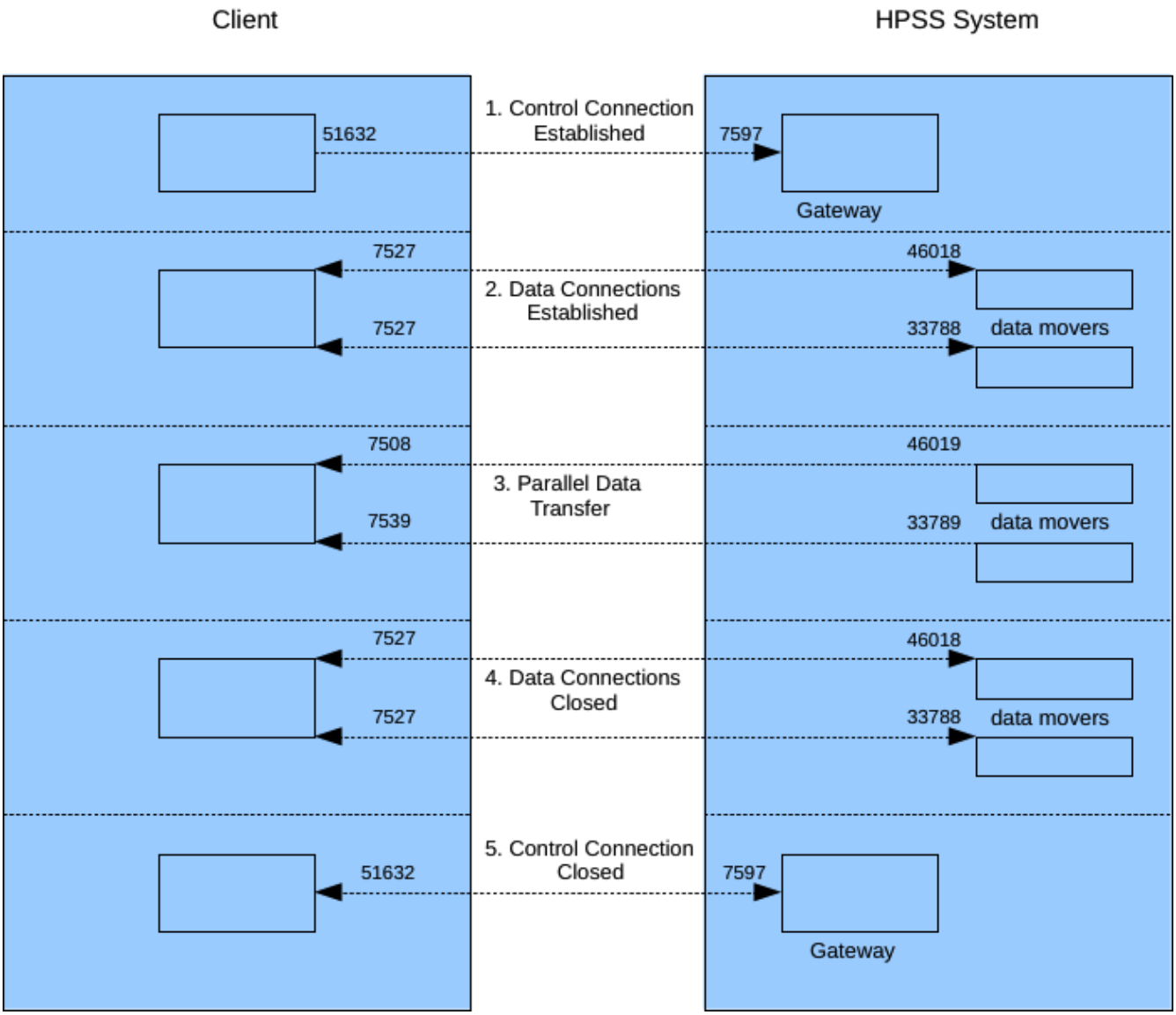

Figure 2: Normal Mode Data Transfer

B: HSI Exit Codes

Exit status 0 indicates success, any other code indicates some type of failure. Status 64 (BADPARAM) is returned on syntax error or other issue not noted by other exit codes.

For complete list of HSI exit codes see *http://www.mgleicher.us/GEL/hsi/hsi-exit-codes.html*.

C: NERSC HPSS Tutorial

Tutorial slides for first-time NERSC HPSS system users are available on the NERSC web site at *https://www.nersc.gov/users/training/events/data-transfer-and-archiving/*.

9 References

[1] Gleicher Enterprises, "Welcome to the HSI Home Page," *http://www.mgleicher.us/GEL/hsi/*.

[2] Gleicher Enterprises, "HSIRC File," *http://www.mgleicher.us/GEL/hsi/hsirc_file.html*.

[3] Andrews, Matthew, *New HPSS PFTP Authentication Mechanism*, October 2007.

[4] National Energy Research Scientific Computing Center (NERSC), "Using Sleepers to Manage Batch Access to HPSS," *https://www.nersc.gov/users/data-and-networking/hpss/storing-and-retrieving-data/sleepers/*.

[5] Gleicher Enterprises, "Using HSI with multiple HPSS systems," *http://www.mgleicher.us/GEL/hsi/hsi_reference_manual_2/using_hsi_with_multiple_hps.html*.

[6] Hurlbert, Wayne, email to FSG Group, November 29, 2007.

[7] Hurlbert, Wayne, email to Patrick Decowski, March 28, 2008.

[8] Gleicher Enterprises, "HSI Man Page," *http://www.mgleicher.us/GEL/hsi/hsi_man_page.html*.