GHI Installation Guide

Copyright notification

DISCLAIMER

Trademark usage

About this book

The Spectrum Scale (formerly GPFS) to HPSS Interface (GHI) installation guide is for use at system installation time. It outlines the steps required to install and configure a GHI system.

Conventions used in this document

Example commands that should be typed at a command line will be preceeded by a percent sign

(%) and be presented in a monospace font:

```
% sample command
```

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

```
# This is a comment
```

Angle brackets ("<>") denote a required argument for a command:

```
% sample command <argument>
```

Square brackets ("[]") denote an optional argument for a command:

```
% sample command [optional argument]
```

Vertical bars ("|") denote different choices within an argument:

```
% sample command <argument1 | argument2>
```

Commands written within the sentence will be bolded:

> Sample sentence with a **command** in it.

A byte is an eight-bit data octet.
A kilobyte, KB, is 1024 bytes (2^{10} bytes).
A megabyte, MB, is 1,048,576 bytes (2^{20} bytes).
A gigabyte, GB, is 1,073,741,824 bytes(2^{30} bytes).
A terabyte, TB, is 1,099,511,627,776 bytes (2^{40} bytes).
A petabyte, PB, is 1,125,899,906,842,624 bytes (2^{50} bytes).
An exabyte, EB, is 1,152,921,504,606,846,976 bytes (2^{60} bytes).

1. Preparing for GHI install

Prior to installing GHI, ensure you understand the customer's requirements, which will help you properly size and configure the GHI system. This installation guide does not document site planning processes; that is part of the proposal or system engineering phase of the project. In addition, refer to the *GHI Management Guide* for planning considerations.

IBM recommends installing GHI on a Spectrum Scale cluster that has no other Hierarchical Storage Management (HSM) application running, for example, Tivoli Storage Manager (TSM). If another

HSM-managed file system is required, it must run on a separate cluster and be remotely mounted on the GHI-managed cluster. GHI is dependent on timely Data Management Application Programming Interface (DMAPI) events from Spectrum Scale; therefore, there should not be two applications competing for events.

For systems installed with High Availability (HA) Core Server, it is critical to ensure that the required GHI components are installed on the backup or stand-by Core Server. These components include Db2 accounts creation and configuration, Db2 server configuration and Independent Standalone HPSS TAR (ISHTAR).

GHI installation requires root or root-equivalent privileges, except where noted otherwise.

1.1. Prerequisites

Before installing GHI, review the GHI release notes on the HPSS Admin wiki for prerequisites, special notes, and possible known issues for the version you plan to install. The release notes define the software version of each prerequisite software:

- HPSS Core Server and Movers
- Operating system
  - memcached
  - libmemcache
  - rpcbind
  - xmlrpc-c
  - xmlrpc-c-client
  - xinetd
  - logrotate
- Python – not covered in this document
- IBM_db (Python support for Db2) – not covered in this document
- Python tqdm (for progress bars)
- Spectrum Scale
- Db2 client
- HPSS client
  - hpss-lib
  - hpss-lib-devel
- GHI-ISHTAR
- GHI

Newer versions of tar (RHEL8+), enable the --sparse option by default. Extractions with the --sparse option do not trigger DMAPI read events in GPFS. This can cause tar extraction to silently fail.

Do not allow the use of tar --sparse when using GPFS with GHI.

As a workaround, use tar --hole-detection=raw.

1.2. Operating system

1.2.1. Set ulimits

Change the default soft and hard core size from "0" to "unlimited". This will allow GHI to create a core dump file for debugging purposes. The default inode scan bucket size is 1000. Increase the max open file descriptors limit to 65536 in `/etc/security/limits.d/19-hpss.conf` on all systems that will run GHI. Reboot each node to validate each change is correct and persistent.

Example:

```
% vi /etc/security/limits.d/19-hpss.conf
#*               soft    core            0
#*               hard    rss             10000
#@student        hard    nproc           20
#@faculty        soft    nproc           20
#@faculty        hard    nproc           50
#ftp             hard    nproc           0
#@student        -       maxlogins       4
*                soft    core            unlimited  <- (add)
*                hard    core            unlimited  <- (add)
*                soft    nofile          65536  <- (add)
*                hard    nofile          65536  <- (add)
```

Validate each change by running:

```
$ ulimit -a
```

1.2.2. Configuration of rsyslog

IBM recommends that you suppress repeat messages and turn rate limiting off.

Sites must evaluate policies and configuration needs for their own systems and determine what works best. Below is an example:

1. In `/etc/rsyslog.conf` update or add the following lines:

```
$SystemLogRateLimitInterval 0
$SystemLogRateLimitBurst 0
$IMUXSockRateLimitInterval 0
$IMJournalRateLimitInterval 0
$IMJournalRateLimitBurst 0
```

2. In `/etc/systemd/journald.conf` update or add the following lines:

```
RateLimitInterval=0
RateLimitBurst=0
Storage=volatile
Compress=no
MaxRetentionSec=5s
```

3. In `/etc/rsyslog.d/hpss.conf` update or add the following line:

```
$RepeatedMsgReduction off
```

4. Restart the services for changes to take effect

```
systemctl restart systemd-journald
systemctl restart rsyslog
```

1.3. Memcached

Memcached is an in-memory key-value store for small chunks of arbitrary data. Memcached allows applications to take memory from parts of the system where it has more than it needs and make it accessible to areas where applications have less than they need.

GHI uses memcached to reduce the load on the HPSS metadata. Memcached improves the performance of GHI image backup verification, **ghiverifyfs**, and **ghi_ls**. Install the memcached and libmemcached-devel RPMs from the RHEL software distribution on each machine you want memcached to run to improve operations.

1.3.1. Install memcached and libmemcached

1. Read the release notes to check prerequisites for the appropriate version to use.

```
% yum list available | grep memcached
% yum install memcached
% yum install libmemcached
```

2. Verify the packages and versions have been properly installed.

```
% rpm -qa | grep memcached
```

1.3.2. Configure memcached

Setup for RHEL 7 nodes

1. Run the following commands to enable, start, and status memcached.

```
% systemctl enable memcached.service
% systemctl start memcached.service
% systemctl status memcached.service
```

2. Verify that memcached configuration files have been created:

```
% cd /usr/lib/systemd/system/
% ls | grep memcached
```

3. If `memcached.services` does not exist, follow these steps:
   a. Create `/usr/lib/systemd/system/memcached.service`.
   b. Add the lines below to the file.

```
[Unit]
Description=Memcached
Before=httpd.service
After=network.target

[Service]
Type=simple
EnvironmentFile=-/etc/sysconfig/memcached
Restart=always
ExecStart=/usr/bin/memcached -u $USER -p $PORT -m $CACHESIZE -c $MAXCONN
$OPTIONS

[Install]
WantedBy=multi-user.target
```

   c. Create the file `/etc/sysconfig/memcached` with contents:

```
PORT="11211"
USER="memcached"
MAXCONN="1024"
CACHESIZE="64"
OPTIONS=""
```

d. Check status and, if necessary, disable, enable, reload, and restart `memcached.service`:

```
% systemctl list-unit-files | grep memcached
% systemctl enable memcached.service
% systemctl daemon-reload
% systemctl restart memcached.service
% systemctl status memcached.service
```

1.4. GHI-ISHTAR

Before installing GHI-ISHTAR, verify that prerequisites hpss-lib, hpss-lib-devel, and hpss-clnt are installed. As of GHI 3.3, ISHTAR is bundled with GHI. The RPM name has changed to be ghi-ishtar-<ghi-version>.0-0, rather than having a specific ISHTAR version number.

```
% rpm -qa "hpss*"
```

1. Install prerequisites if they are missing.

   ```
   % rpm -ivh hpss-lib-<version>*
   % rpm -ivh hpss-lib-devel-<version>*
   % rpm -ivh hpss-clnt-<version>*
   ```

   After HPSS RPMs are installed, a message will appear letting the user know where the package directory is located. This directory path will be needed for the next step.

   ```
   root@elayne /hpss_src/hpss753 > rpm -ivh hpss-clnt-7.5.3.0-0.el7.ppc64le.rpm
   Preparing...                          ################################# [100%]
   Updating / installing...
     1:hpss-clnt-7.5.3.0-0.el7          ################################# [100%]
   Files for package hpss-clnt installed under /hpss_src/hpss-7.5.3.0-0.el7
   ```

2. Create /opt/hpss link to the directory where HPSS Client files are installed.

   ```
   % ln -s /hpss_src/hpss-<version>* /opt/hpss
   ```

   Example:

   ```
   % ln -s /hpss_src/hpss-7.5.3.0-0.el7 /opt/hpss
   ```

1.5. Install GHI-ISHTAR

GHI-ISHTAR must be installed on all GHI nodes. GHI-ISHTAR is used by the IOM and by session node tools such as ghiverifyaggr.

GHI-ISHTAR is bundled with GHI. Only the provided version should be used with GHI.

```
$ rpm -ivh ghi-ishtar*.rpm
```

Files for package `ghi-ishtar` are installed under `/var/hpss/hsi`.

HPSS libraries must be installed on each GHI node before GHI-ISHTAR can be installed.

2. Spectrum Scale

2.1. Install Spectrum Scale

Contact your IBM Spectrum Scale customer support representative to obtain the Spectrum Scale software, and install it according to instructions.

2.2. Configure Spectrum Scale

1. After Spectrum Scale is installed, make sure **ssh** or **rsh** is working between nodes in the cluster. If using **ssh**, be certain to complete additional configuration steps to allow for passwordless command execution (steps are covered in the Spectrum Scale documentation).
2. Enable threshold processing. Check to see if the requested configuration attributes are set:

   ```
   % mmlsconfig
   % mmchconfig enablelowspaceevents=yes
   ```

3. Tune GPFS thread limits for performance. Check to see if the requested configuration attributes are set. The maximum DMAPI workers is 64. The general thread count should be configured in consultation with support.

   ```
   % mmlsconfig
   % mmchconfig dmapiWorkerThreads=64
   % mmchconfig maxGeneralThreads=1280
   ```

4. Configure NSD (Network Shared Disk) multipath. If using multipath, follow the steps below to create NSDs:

a. Create a `/etc/multipath/bindings` file. The file needs to match on all nodes using the NSD.

b. Create an **nsddevices** script for NSD discovery:

```
% cp /usr/lpp/mmfs/samples/nsddevices.sample /var/mmfs/etc/nsddevices
```

c. Edit `/var/mmfs/etc/nsddevices` to look like the example below:

```
osName=$(/bin/uname -s)

if [[ $osName = Linux ]]
then
CONTROLLER_REGEX='mpath[a-z]+'
for dev in $( /bin/ls /dev/mapper | egrep $CONTROLLER_REGEX )
do
# dmm vs. generic is used by Spectrum Scale to prioritize internal order
# of searching through available disks, then later Spectrum Scale
# discards other disk device names that it finds that match as the
# same NSD device by a different path. For this reason,
# dmm vs. generic is an important distinction if you are not
# explicitly producing the entire and exclusive set of disks
# that Spectrum Scale should use, as output from this
# script (nsddevices) and exiting this script with a "return 0".
echo mapper/$dev dmm
echo mapper/$dev generic
done
fi

# To bypass the Spectrum Scale disk
# discovery (/usr/lpp/mmfs/bin/mmdevdiscover),
return 0
# To continue with the Spectrum Scale disk discovery steps,
return 1
```

d. Ensure this script is executable:

```
% chmod +x /var/mmfs/etc/nsddevices
```

e. Execute `/var/mmfs/etc/nsddevices`.

Example:

```
% /var/mmfs/etc/nsddevices
mapper/mpatha dmm
mapper/mpathb dmm
mapper/mpathc dmm
```

2.3. Create SSH trust

After Spectrum Scale is installed, create SSH trust between all nodes in each direction in each node and between each node. Be certain to complete additional configuration steps to allow for passwordless command execution.

2.4. Create a new Spectrum Scale cluster

Run the command to create a Spectrum Scale cluster on only the main node. Upon successful completion of the **mmcrcluster** command, the `/var/mmfs/gen/mmsdrfs` and the `/var/mmfs/gen/mmfsNodeData` files are created on each node in the cluster.

1. Run **mmcrcluster**. Example:

```
% mmcrcluster -n /var/hpss/ghi/gpfs_config/node.conf -p ghi_server1 \
-r /usr/bin/ssh -R /usr/bin/scp
```

2. Check that the `mmsdrfs` and `mmfsNodeData` files are created and the output shows success and completion:

```
% cat /var/mmfs/gen/mmsdrfs
% cat /var/mmfs/gen/mmfsNodeData
```

Output:

```
mmcrcluster: Performing preliminary node verification ...
mmcrcluster: Processing quorum and other critical nodes ...
mmcrcluster: Finalizing the cluster data structures ...
mmcrcluster: Command successfully completed
mmcrcluster: Warning: Not all nodes have proper GPFS license designations.
mmcrcluster: Propagating the cluster configuration data to all affected nodes.
This is an asynchronous process.
```

2.4.1. Configure license

The **mmchlicense** command designates appropriate Spectrum Scale licenses. Run **mmchlicense** to accept and configure licenses.

```
% mmchlicense server --accept -N all
```

Output (the following nodes will be designated as possessing server licenses):

```
ghi_server2.clearlake.ibm.com
ghi_server1.clearlake.ibm.com
```

2.4.2. Create NSDs on the main GHI node only

1. On the primary GHI node, create NSD configuration file(s) for each disk:

```
% cd /var/hpss/ghi/gpfs_config
% touch nsd.StanzaFile nsd.StanzaFile2 ... nsd.StanzaFileX
% vi nsd.StanzaFile
```

2. Add the following lines:

```
%nsd:
device=/dev/sdb
nsd=nsd1
servers=ghi_server1
usage=dataAndMetadata
% vi nsd.StanzaFile2
```

3. Add the following lines:

```
%nsd:
device=/dev/sdc
nsd=nsd2
servers=ghi_server1
usage=dataAndMetadata
```

Create a block for each resource. Include all GHI nodes that see the disk, separated by a comma. For example, if two servers share a disk resource "servers=" value, the line will contain both hostnames like this: `servers=` <*node1 shortname*>, <*node2 shortname*>.

4. Create NSD stanzas file that uses the multipath aliases. For systems using multipath, skip this step if you are not using multipath. Edit `/var/hpss/ghi/gpfs_config/nsd.StanzaFile` and insert the lines below:

```
%nsd: device=/dev/mapper/mpatha
nsd=nsd1
servers=ghi_server1,ghi_server2
usage=dataAndMetadata
```

5. Enable DMAPI on the Spectrum Scale file system:

```
% mmchfs <file system> -z yes
```

6. Run the **mmcrnsd** command to create NSD servers. The option **-F** specifies the file containing the NSD stanzas for the disks to be created. The option **-v no** specifies that the disks are to be created irrespective of their previous state.

```
% mmcrnsd -F /var/hpss/ghi/gpfs_config/nsd.StanzaFile -v no

mmcrnsd: Processing disk sdb
mmcrnsd: Propagating the cluster configuration data to all affected nodes.
This is an asynchronous process.

% mmcrnsd -F /var/hpss/ghi/gpfs_config/nsd.StanzaFile2 -v no

mmcrnsd: Processing disk sdc
mmcrnsd: Propagating the cluster configuration data to all affected nodes.
This is an asynchronous process.
```

7. Ensure all the Spectrum Scale nodes are active and then create the Spectrum Scale file system. Wait until the **mmgetstate** output shows that all nodes are active before issuing the **mmcrfs** command.

```
% mmgetstate -a

Node number  Node name        GPFS state

1      ghi_server1            active
2      ghi_server2            active
```

If the node state remains down, run **mmstartup -a** to start Spectrum Scale.
If the node state remains down after **mmstartup**, check the Spectrum Scale logs.
If the node state is arbitrating, check the Spectrum Scale logs.
If the node needs to be recycled, run **mmshutdown -a**, and rerun **mmstartup**.

The Spectrum Scale log location is `/var/mmfs/gen/mmfslog`.

The GHI file system name and mount point must be unique among all GHI clusters connected to the HPSS system. The GHI file system name matches the Spectrum Scale file system name and the GHI file system mount point matches the Spectrum Scale mount point.

8. Run **mmcrfs** to create the file system(s) with options to enable automount (**-A yes**), activate quotas automatically (**-Q yes**), enable DMAPI (**-z yes**), set blocksize (**-B 256K**), and specify that the disk not belong to an existing file system (**-v no**). For purposes of this documentation, the **mmcrfs** commands below have each been split into two separate lines; each indented line

should be considered as part of the command line above it, with no breaks.

```
% mmcrfs /ghi_server1_fs1 /dev/ghi_server1_fs1
   -F var/hpss/ghi/gpfs_config/nsd.StanzaFile -A yes -Q yes -z yes -B 256K -v no

% mmcrfs /ghi_server1_fs2 /dev/ghi_server1_fs2
   -F var/hpss/ghi/gpfs_config/nsd.StanzaFile2 -A yes -Q yes -z yes -B 256K -v no
```

If the user plans on having a Spectrum Scale file system without a GHI file system for image restores, the "temp space" Spectrum Scale file system should have DMAPI set to "no" (**-z no**).

Sample output:

```
The following disks of ghi_server1_fs2 will be formatted on node
ghi_server2.clearlake.ibm.com:
    nsd2: size 153600 MB
Formatting file system ...
Disks up to size 1.51 TB can be added to storage pool system.
Creating Inode File
Creating Allocation Maps
Creating Log Files
Clearing Inode Allocation Map
Clearing Block Allocation Map
Formatting Allocation Map for storage pool system
Completed creation of file system /dev/ghi_server1_fs2.
mmcrfs: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

Use **mmlsfs** to list the file system attributes. For example, if you want to check if DMAPI is enabled on all Spectrum Scale file systems, run: **mmlsfs all | grep DMAPI**

9. Display the configuration data for a Spectrum Scale cluster for each node. Log in to the main node:

```
% root@ghi_server1 /var/mmfs > mmlsconfig

Configuration data for cluster ghi_server1.clearlake.ibm.com:

clusterName ghi_server1.clearlake.ibm.com
clusterId 16335425671093415616
autoload no
dmapiFileHandleSize 32
minReleaseLevel 5.0.2.0
ccrEnabled yes
cipherList AUTHONLY
adminMode central

File systems in cluster ghi_server1.clearlake.ibm.com:

/dev/ghi_server1_fs1
/dev/ghi_server1_fs2
```

Log in to all secondary nodes to check:

```
% root@ghi_server2 /root > mmlsconfig

Configuration data for cluster ghi_server1.clearlake.ibm.com:

clusterName ghi_server1.clearlake.ibm.com
clusterId 16335425671093415616
autoload no
dmapiFileHandleSize 32
minReleaseLevel 5.0.2.0
ccrEnabled yes
cipherList AUTHONLY
adminMode central

File systems in cluster ghi_server1.clearlake.ibm.com:

/dev/ghi_server1_fs1
/dev/ghi_server1_fs2
```

2.4.3. NSD multipath

1. Configure NSD multipath. If using multipath, follow the steps below to create NSDs:
   a. Create a `/etc/multipath/bindings` file. The file needs to match on all nodes using the NSD.
   b. Create an **nsddevices** script for NSD discovery.

```
% cp /usr/lpp/mmfs/samples/nsddevices.sample /var/mmfs/etc/nsddevices
```

c. Edit `/var/mmfs/etc/nsddevices` to look like the example below:

```
osName=$(/bin/uname -s)

if [[ $osName = Linux ]]
then
CONTROLLER_REGEX='mpath[a-z]+'
for dev in $( /bin/ls /dev/mapper | egrep $CONTROLLER_REGEX )
do
# dmm vs. generic is used by Spectrum Scale to prioritize internal
# order of
# searching through available disks, then later Spectrum Scale
# discards other disk device names that it finds that match as the same
# NSD device by a different path. For this reason, dmm vs. generic is an
# important distinction if you are not explicitly producing the entire
# and exclusive set of disks that Spectrum Scale should use,
#  as output from
# this script (nsddevices) and exiting this script with a "return 0".
echo mapper/$dev dmm
echo mapper/$dev generic
done
fi

if [[ $osName = AIX ]]
then:
# Add function to discover disks in the AIX environment.
fi

# To bypass the Spectrum Scale disk discovery
# (/usr/lpp/mmfs/bin/mmdevdiscover),
return 0
# To continue with the Spectrum Scale disk discovery steps,
return 1
```

d. Ensure the script is executable. Example:

```
% chmod +x /var/mmfs/etc/nsddevices
```

e. Execute `/var/mmfs/etc/nsddevices`. Example:

```
# /var/mmfs/etc/nsddevices
mapper/mpatha dmm
mapper/mpathb dmm
mapper/mpathc dmm
```

f. Create NSD stanzas file that uses the multipath aliases. Edit `/var/hpss/ghi/gpfs_config/nsd.StanzaFile` and insert the lines:

```
%nsd: device=/dev/mapper/mpatha
nsd=nsd1
servers=ghi_server1,ghi_server2
usage=dataAndMetadata
```

g. Continue with creating NSDs.

2. Enable DMAPI on the Spectrum Scale file system:

```
% mmchfs <file system> -z yes
```

3. Db2

3.1. Users and groups

GHI needs two users (hpss, hpssdb) and two groups (hpss, hpsssrvr) on all GHI nodes that will have the HPSS client installed. Each GHI cluster requires an additional user per cluster on GHI nodes within the cluster that will have the HPSS client installed. The names of these cluster users are arbitary, but this document will use ghicluster1, ghicluster2...ghiclusterX in future examples. The ghiclusterX users should belong to the hpsssrvr group. The user and group ID numbers created on the GHI nodes must match the corresponding user and group ID numbers on the HPSS Core Server. User IDs hpss and hpssdb should exist after the HPSS Core Server has been installed and configured. The user ID for the GHI cluster user(s) will need to be created on the HPSS Core Server using **hpssuser**.

1. Use the system command **id** to verify the required users and groups exist:

```
% id <user>
% id -g <user>
```

User	**Primary Group**	**Home Directory:**
hpss	hpss	/var/hpss
hpssdb	hpssdb	/db2data/db2_hpssdb
ghicluster1	hpsssrvr	/var/ghicluster1

2. If any of the above users or groups do not exist, use the **useradd** system command to add them. The following shows the usage of the **useradd** command and an example adding hpssdb as a user and group:

```
% useradd -d <home directory> -g <group> -p password <user>

% useradd -d /db2data/db2_hpssdb -g 300 -p hpssdb hpssdb
```

3. Ensure the Core Server and GHI nodes have matching entries for users hpss, hpssdb, and GHI cluster user(s) in the `/etc/passwd` and `/etc/group` files:

```
% cat /etc/passwd | grep hpss
hpss:x:300:300:HPSS User:/var/hpss:/bin/bash
hpssdba:x:301:301::/db2data/db2_hpssdb:/bin/bash
ghicluster1:x:1001:302::/var/ghicluster1:/bin/bash

% cat /etc/group | grep hpss
hpss:x:300:hpss,hpssdba
hpssdba:x:301:root
hpsssrvr:x:302:ghicluster1
```

The ghicluster1 user in `/etc/passwd` is in the primary group of hpsssrvr. Also notice that in `/etc/group` ghicluster1 is a member of the hpsssrvr group. Make sure all Core Server and GHI nodes have the correct configuration and passwords.

3.1.1. Add GHI cluster user with the hpssuser tool

1. On the HPSS Core Server, use **hpssuser** to add a GHI cluster user. See the *HPSS Management Guide* for more details.

   Each GHI cluster user should have a unique name and user Id. The output below uses ghiclusterX for the name and 1001 for the user ID as an example.

```
The output below is for adding a user with UNIX authentication. The command
is the same when adding a user with Kerberos authentication. The output changes
slightly.

$ /opt/hpss/bin/hpssuser -add ghiclusterX -acct -keytab <keytab path>
User ID#: 1001
Primary group name: hpsssrvr
Enter password for ghiclusterX: [ghiclusterX]
Re-enter password to verify: [ghiclusterX]
Full name: ghiclusterX
Login shell: /bin/bash
Unix (local/system) home directory: /var/ghiclusterX
[ adding unix user ]
[ added unix user ]
[ adding unix keytab entry to '/var/hpss/etc/hpss.unix.keytab' ]
[ added unix keytab entry to '/var/hpss/etc/hpss.unix.keytab' ]
```

2. Check that GHI cluster user has been added to `/var/hpss/etc/passwd` and to `/var/hpss/etc/group` under the group hpsssrvr. This step is valid only if you are using HPSS local password and group files; otherwise, skip this step.

```
The output below uses ghiclusterX as the name and 1001 as the user ID as an
example.

% cat /var/hpss/etc/passwd | grep ghiclusterX
ghiclusterX:x:1001:301:ghiclusterX:/var/ghiclusterX:/bin/bash

% cat /var/hpss/etc/group | grep ghiclusterX
hpsssrvr:*:301:hpssmvr,hpsssd,hpssftp,hpssssm,hpsspvr,hpssgk,hpssmps,
hpssrait,hpsscore,hpsspvl,hpssfs,hpssls,ghiclusterX
```

3. Copy HPSS Core `/var/hpss/etc/` to each GHI node with **scp**. On the core:

```
% cd /var/hpss/etc
% tar -cvzf /tmp/etcnew.tar.gz ./
% scp /tmp/etcnew.tar.gz root@<GHI NODE>:/var/hpss
```

4. Move and rename the old `/var/hpss/etc` directory, and create a new `/var/hpss/etc` directory. On each GHI node:

```
% cd /var/hpss/
% mv etc etc.ori
% mkdir /var/hpss/etc
% cp /var/hpss/etcnew.tar.gz /var/hpss/etc
% cd /var/hpss/etc
% tar -xzvf etcnew.tar.gz
```

5. On the GHI node modify the `/var/hpss/etc/env.conf` file to have the environment variable `HPSS_PRINCIPAL_DMG` set to the GHI cluster user principal name.
6. On the GHI node modify the `/var/hpss/etc/env.conf` file to have the environment variable `HPSS_PRINCIPAL_SSM` set to the hpssssm user.
7. This step is valid only if you are using HPSS local password and group files; otherwise, skip this step. Remove nonrequired GHI cluster users from GHI nodes. For example, there are 2 GHI clusters, first cluster is associated with ghicluster1 user, second cluster is associated with ghicluster2 user.

```
On cluster 1 GHI nodes:
% hpssuser -del ghicluster2 -acct

On cluster 2 GHI nodes:
% hpssuser -del ghicluster1 -acct
```

8. Link `/var/hpss/hpssdb` to the hpssdb user's home directory. On each GHI node:

```
$ ln -s /db2data/db2_hpssdb /var/hpss/hpssdb
```

3.2. Set up GHI tablespace on the HPSS Core Server

GHI should be configured to use the same Db2 storage group that is used in HPSS.

GHI tablespaces should be configured on the HPSS Core Server only while the HPSS system is down. The actual configuration for Db2 should be determined during the system engineering planning phase of the deployment. The GHI Db2 mapping table has the potential to become very large and care should be taken in configuring Db2 to handle it.

Repeat this section to set up the GHI tablespace on the HA Backup Core Server for proper failover operations.

3.2.1. Database using single partition

This configuration is performed only on the HPSS Core Server while Db2 is running and HPSS servers are down.

1. Shut down all servers via the HPSS GUI.
2. Find the number of partitions. As hpssdb user, the following shows there is only one partition:

```
% cat $HOME/sqllib/db2nodes.cfg
0 <core_server>.clearlake.ibm.com 0
```

3. Source the database profile:

```
% source ~hpssdb/sqllib/db2profile
```

4. Create the database. The second of the following two examples is the default for a one partition and two storage paths file system. For systems that do not use the default, edit path partition names and storage path file systems to match your system configuration. The following examples show path names and partition expressions usage:

```
% db2 "CREATE DATABASE HGHI ON \
\'/db2data/p0000/stg0001', \
\'/db2data/p0000/stg0002' \
DBPATH on '/db2data/db2_hpssdb'"
```

```
% db2 "CREATE DATABASE HGHI ON \
\'/db2data/p \$4N /stg0001\', \
\'/db2data/p \$4N /stg0002\' \
DBPATH ON '/db2data/db2_hpssdb'"
```

5. Modify the callback script to source the database profile (DB2PROF):

```
% vim /opt/ghi/bin/hpssEventNotify
```

3.2.2. Create database partition group

1. Connect to the HGHI database:

```
% db2 CONNECT TO HGHI
```

2. For a single partition run the command:

```
% db2 "CREATE DATABASE PARTITION GROUP HPSS_GHI ON DBPARTITIONNUM (0)"
```

3. Check that a partition is created:

```
$ db2 list db partition groups
```

Example output:

```
 DATABASE PARTITION GROUP
 ----------------------------
 HPSS_GHI
 IBMCATGROUP
 IBMDEFAULTGROUP

 3 record(s) selected.
```

4. Create the bufferpool used for GHI DB tablespace:

```
% db2 "CREATE BUFFERPOOL SMALLTABLES \
   DATABASE PARTITION GROUP HPSS_GHI SIZE 1000 AUTOMATIC \
   PAGESIZE 4K"
```

5. Create the bufferpool used for GHI mapping tablespace:

```
% db2 "CREATE BUFFERPOOL bp32k \
   DATABASE PARTITION GROUP HPSS_GHI SIZE 1000 AUTOMATIC \
   PAGESIZE 32K"
```

6. Create Db2 tablespaces.

   a. Create Db2 tablespace for GHIDB:

```
% db2 "CREATE LARGE TABLESPACE GHIDB \
 IN DATABASE PARTITION GROUP HPSS_GHI \
 PAGESIZE 4K \
 MANAGED BY AUTOMATIC STORAGE \
 AUTORESIZE YES \
 INITIALSIZE 32M \
 MAXSIZE NONE \
 EXTENTSIZE 128 \
 PREFETCHSIZE AUTOMATIC \
 BUFFERPOOL "SMALLTABLES" \
 OVERHEAD 7.500000  \
 TRANSFERRATE 0.060000 \
 NO FILE SYSTEM CACHING \
 DROPPED TABLE RECOVERY ON \
 DATA TAG NONE"
```

   b. Create Db2 tablespace for GHIMAPPING:

```
% db2 "CREATE LARGE TABLESPACE GHIMAPPING
 IN DATABASE PARTITION GROUP HPSS_GHI \
 PAGESIZE 32K \
 MANAGED BY AUTOMATIC STORAGE \
 AUTORESIZE YES \
 EXTENTSIZE 128 \
 PREFETCHSIZE AUTOMATIC \
 BUFFERPOOL BP32K \
 DATA TAG NONE \
 OVERHEAD 7.500000 \
 TRANSFERRATE 0.060000 \
 MAXSIZE NONE  \
 NO FILE SYSTEM CACHING \
 DROPPED TABLE RECOVERY ON"
```

3.2.3. Configure logging on the HPSS Core Server

1. Grant user hpss access to the database:

```
% db2 "grant connect on database to user hpss"
% db2 "grant createtab on database to user hpss"
% db2 "grant dbadm on database to user hpss"
```

2. Configure the primary logs, secondary logs, log archives, log file size, and number of logs similar to the standard of the HPSS databases:

```
% mkdir /db2data/p0000/db2_log/hghi
% db2 "update db cfg for hghi using NEWLOGPATH <primary_log_path> hghi"
% db2 "update db cfg for hghi using NEWLOGPATH '/db2data/p0000/db2_log/hghi'"
% mkdir /db2data/p0000/db2_logmirror/hghi
% db2 "update db cfg for hghi using MIRRORLOGPATH <secondary_log_path> hghi"
% db2 "update db cfg for hghi using MIRRORLOGPATH
'/db2data/db2_logmirror/hghi'"
% db2 "update db cfg for hghi using AUTO_MAINT off"
% db2 "update db cfg for hghi using AUTO_RUNSTATS off"
% db2 "update db cfg for hghi using AUTO_TBL_MAINT off"
% mkdir /db2data/p0000/db2_logarchive1/hghi
% db2 "update db cfg for hghi using LOGARCHMETH1 \
DISK:/ <primary_log_archive_path>/hghi/"
% db2 "update db cfg for hghi using LOGARCHMETH1 \
DISK:/db2data/p0000/db2_logarchive1/hghi/"
% mkdir /db2data/p0000/db2_logarchive2/hghi
% db2 "update db cfg for hghi using LOGARCHMETH2 \
DISK:/<secondary_log_archive_path>/hghi/"
% db2 "update db cfg for hghi using LOGARCHMETH2 \
DISK:/db2data/p0000/db2_logarchive2/hghi/"

% db2 "update db cfg for hghi using LOGFILSIZ 25000"
% db2 "update db cfg for hghi using LOGPRIMARY 10"
% db2 "update db cfg for hghi using LOGSECOND -1"
```

Table 1. LOGBUFSZ

Machine Memory	**LOGBUFSZ <Table Value>**
Less than 16 GB RAM	4096
Between 16 GB and 64 GB RAM	8192
Greater than 64 GB RAM	16384

```
% db2 "update db cfg for hghi using LOGBUFSZ <table value>"

% db2 "update db cfg for hghi using DFT_QUERYOPT 2"
```

3. Disconnect from the database:

```
% db2 disconnect all
```

3.3. Install Db2 client on all GHI nodes

Install the Db2 client on each Spectrum Scale quorum node (all nodes which include “quorum” in the “Designation” column from the **mmlscluster** command). Follow the IBM Db2 *Command Reference* document to install the server.

3.4. Add Db2 permanent licenses on all GHI nodes

Add a permanent license on each Spectrum Scale quorum node that has the Db2 client installed.

1. Add license:

```
% cd /opt/ibm/db2/<version>/adm
% ./db2licm -a <path name to Db2 generic license file>/db2aese_c.lic
```

The generic Db2 license file (*/db2/license/db2ese_c.lic) can be found on the Db2 Installation CD or image. It can also be obtained by contacting HPSS support.

Refer to the IBM Db2 *Command Reference* document for more information on how to use the **db2licm** utility to manage the Db2 license. Create the Db2 database connection on the GHI session nodes which should already have the Db2 client installed per the prerequisites.

2. Create an instance as root:

```
% /opt/ibm/db2/<version>/instance/db2icrt -a CLIENT -s client -u hpssdb hpssdb
```

3. Source db2profile system-wide to establish database environment. As root, add lines to aliases.sh.

```
$ su - root
$ vim /etc/profile.d/aliases.sh
. ~hpssdb/sqllib/db2profile
```

4. Set DB2COMM. As hpssdb:

```
% su - hpssdb
% db2set DB2COMM=tcpip
```

5. Verify that DB2COMM is set to "TCPIP":

```
% db2set -all

[i] DB2COMM=TCPIP
[g] DB2SYSTEM=ghi_server1.clearlake.ibm.com
```

6. Verify the local services in `/etc/services` file for Db2 support. As root, copy the Db2 service entries from the Core Server `/etc/services` file. The number of entries will differ based on configuration. Example output:

```
# Local services
db2c_hpssdb     59999/tcp
DB2_hpssdba     60000/tcp
DB2_hpssdba_1   60001/tcp
DB2_hpssdba_2   60002/tcp
DB2_hpssdba_END 60003/tcp
```

7. Catalog the database profile:

```
% db2 catalog tcpip node $NODE remote $HPSS_CORE server $PORT

% db2 catalog tcpip node ghi_server2 remote <HPSS_Core_server> server 59999
DB20000I  The CATALOG TCPIP NODE command completed successfully.
DB21056W  Directory changes may not be effective until the directory cache is
refreshed.
```

Where:

$NODE
: unique name; using the short host name of the current machine is recommended.

$HPSS_CORE
: hostname of the HPSS Core server.

$PORT
: port number acquired from the Core Server `/etc/services` file.

Steps to check hpssdb port on Core Server:

a. Source the database profile:

```
% . ~hpssdb/sqllib/db2profile
```

b. Run the command:

```
% db2 get dbm cfg | grep SVCENAME
```

c. Look at the value for the SVCENAME:

```
TCP/IP Service name     (SVCENAME) = db2_hpssdb
```

d. Cat the `/etc/services` file and **grep** for the SVCENAME from above:

```
% cat /etc/services | grep db2_hpssdb
```

e. Use the port number found from the **grep** of the `/etc/services` file for $PORT.

8. Catalog the database hghi:

```
% db2 catalog db hghi as hghi at node $NODE

% db2 catalog db hghi as hghi at node ghi_server2
Db20000I  The CATALOG DATABASE command completed successfully.
Db21056W  Directory changes may not be effective until the directory cache is
refreshed.
```

Verify that the Db2 client can connect to the Db2 server on the HPSS core machine:

```
% /opt/ghi/bin/ghi_db_test --connect
```

9. Catalog each hpss subsystem database (for ghi_verify.py):

```
% db2 catalog db hsubsys1 as hsubsys1 at node $NODE

% db2 catalog db hsubsys1 as hsubsys1 at node ghi_server2
Db20000I  The CATALOG DATABASE command completed successfully.
Db21056W  Directory changes may not be effective until the directory cache is
refreshed.
```

4. HPSS

The HPSS Core Server must also be able to connect to the network configured for the Spectrum Scale configuration. For example, if the Spectrum Scale cluster is configured exclusively on a data network, HPSS must be able to connect to that data network, even if the Spectrum Scale nodes also have an additional network to connect to the HPSS Core Server.

4.1. Verify HPSS RPMs on all GHI nodes

Verify that the following RPMs are installed on all the GHI nodes:

```
% rpm -qa | grep hpss
 hpss-clnt-<version>*
 hpss-lib-<version>*
 hpss-lib-devel-<version>*
```

These should exist when GHI-ISHTAR was previously installed.

4.2. Configure the HPSS client

1. Set up `/var/hpss/etc` on GHI client machines.
   a. Verify that `/var/hpss/etc/*` was copied from the HPSS Core Server to each GHI node.
   b. Add HPSS_API_HOSTNAME=<long hostname> to `/var/hpss/etc/env.conf`.
   c. Add HPSS_PTHREAD_STACK=524288 to `/var/hpss/etc/env.conf`.
2. Set up authentication. Copy the HPSS PAM module (`/etc/pam.d/hpss`) from the HPSS Core Server to `/etc/pam.d/hpss` on all GHI nodes.
3. Set up links:

   ```
   % /opt/ibm/db2 > ln -s /opt/ibm/db2/<version> /opt/ibm/db2/default
   % /opt/ghi/db2 > ln -s /opt/ibm/db2/<version> /opt/ghi/db2/default
   ```

4. If using Kerberos authentication, copy `/etc/krb5.conf` from the HPSS Core Server to all GHI nodes.
5. Specify the HPSS_NET_FAMILY. Ensure that the HPSS client configuration has the correct HPSS_NET_FAMILY in `/var/hpss/etc/env.conf`. The default value is "ipv4_only". Examples:

   ```
   ipv6_only
   ipv4_only
   ipv6
   ```

5. GHI installation and configuration

5.1. Install GHI

1. Install the following RPMs on all non-IOM nodes:

```
% rpm -ivh ghi-lib-<version>.el#.hpss.<hpss version>.<architecture>.rpm
% rpm -ivh ghi-<version>.el#.hpss.<hpss version>.<architecture>.rpm
% rpm -ivh ghi-ishtar-<version>.el#.hpss.<hpss version>.<architecture>.rpm
```

GHI files will be installed under the directory `/hpss_src/ghi-<version>`. Note that the ghi-iom RPM should not be installed on the same machine as the ghi RPM. See IOM install for information on installing the ghi-iom RPM.

2. Create a link at `/opt/ghi` to `/hpss_src/ghi-<version>`. GHI requires this link to exist to function properly.

```
% ln -s /hpss_src/ghi-<version> /opt/ghi
```

3. Verify that the following directories exist:

```
/opt/ghi
/opt/ghi/bin
/opt/ghi/lib
/usr/share/man/cat7
/var/hpss/ghi
/var/hpss/ghi/policy
/var/hpss/ghi/config
/var/hpss/ghi/config/templates
/var/hpss/hsi/bin
```

4. Create a `/var/hpss/ghi/etc` directory:

```
% mkdir /var/hpss/ghi/etc
```

5.1.1. Configure GHI-ISHTAR

1. Copy the `htar.ksh` wrapper script to `/var/hpss/hsi/bin`:

```
% cd /var/hpss/hsi/bin
% cp htar.ksh.template htar.ksh
% edit htar.ksh  (Variables to modify are described with example below)
% /bin/chmod 755 htar.ksh
```

GHI-ISHTAR needs to be installed and configured on all GHI nodes, including each IOM. See section *Create IOMs for each GHI-managed file system* for information on installing IOMs.

2. Modify the `htar.ksh` script to provide correct values for the following variables:

TMPDIR

Location of the temporary files. The amount of space required is based on the size of an aggregate, plus temporary files created for the data files. Example:

```
export TMPDIR=/<Spectrum Scale_mount_point>/scratch/.ghi
```

DEFAULT_REALM

Realm name for the location of the HPSS Core Server. This name must exactly match what is set for "site name" in `/var/hpss/etc/site.conf` from the Core Server, including case sensitivity. Example:

```
if [ "$DEFAULT_REALM" = "" ]; then
DEFAULT_REALM=core_server.clearlake.ibm.com
fi
```

HPSS_AUTH_METHOD

Set this variable for the desired authentication type ("unix" for UNIX, or "kerberos" for Kerberos). This variable will determine the keytab file you will use. Example:

```
export HPSS_AUTH_METHOD=unix
```

HPSS_PRINCIPAL

HPSS principal to use for HTAR. Note that this varies depending on auth method used so this setting must come AFTER HPSS_AUTH_METHOD is set. This HPSS principal should be the principal configured for this GHI cluster. Example:

```
Change <GHI_CLUSTER_USER> to the user configured for this cluster.

if [ "$HPSS_PRINCIPAL" = "" ]; then
  if [ "$HPSS_AUTH_METHOD" = "kerberos" ]; then
      export HPSS_PRINCIPAL=`whoami`@$DEFAULT_REALM
  else
     if [ "`whoami`" = "root" ]; then
        export HPSS_PRINCIPAL=<GHI_CLUSTER_USER>
     else
        export HPSS_PRINCIPAL=`whoami`
     fi
```

HPSS_KEYTAB_PATH

Location of keytab. Set this variable when using UNIX authentication (ex. `/var/hpss/etc/hpss.unix.keytab`). Example:

```
export HPSS_KEYTAB_PATH=
/var/hpss/etc/hpss.unix.keytab
```

HPSS_HOSTNAME

Interface to be used for the data path. Example:

```
export HPSS_HOSTNAME=ghi_server1
```

5.2. GHI users and groups

All authentication and authorization are done using the GHI cluster principal. Each GHI cluster should use a unique pricipal. In the following instruction the principals ghiclusterX will be used to refer to these GHI cluster principals. Where "X" is represents which cluster the principal is for. Meaning ghicluster1 is the principal for the first GHI cluster. The numeric IDs must match those on the HPSS Core Server, which may be obtained from the `/etc/passwd` file on your HPSS Core Server.

1. Verify the ghiclusterX user ID exists on each GHI node within the GHI cluster.
2. Verify group ID hpsssrvr is set for ghiclusterX.

If the user ghiclusterX or group hpsssrvr do not exist, create them.

5.3. Configure GHI

GHI is configured using command-line tools. All the GHI commands discussed in this section are fully documented in the *GHI Management Guide*.

These are the steps to configure GHI:

1. Create GHI cluster from the Spectrum Scale configuration.
2. Add the Spectrum Scale file system for GHI to manage.
3. Add IOMs for each GHI-managed Spectrum Scale file system.

5.4. Create the GHI cluster

Define the overall cluster configuration, including the nodes which will be known to GHI (not necessarily all nodes known to Spectrum Scale). This is accomplished using the **ghicrcluster** command. The **ghicrcluster** command must run on the session node that is designated as "cluster manager node". Use the **mmlsmgr** command to determine which node is the cluster manager.

```
root@ghi_server1 /var/hpss/hsi/bin > mmlsmgr
file system      manager node
---------------- ------------------
ghi_server1_fs2        192.168.221.199 (ghi_server1)
ghi_server1_fs1        192.168.221.200 (ghi_server2)

Cluster manager node: 192.168.221.199 (ghi_server1)
```

In addition, Spectrum Scale must be running when defining the cluster configuration to GHI.

All nodes that are designated as quorum with the **mmlscluster** command must be listed after the cluster manager. This will allow GHI to assign them as a manager node in the case of a failover.

```
% ghicrcluster [-v] <GHI_node1> <GHI_node2> <GHI_node3> ...
% ghicrcluster [-v] -N <nodelist_file>
% ghicrcluster -r [-v]
```

Where:

<***GHI_node#***> | <***nodelist_file***>

The node list of machines from **mmlscluster** which will have the designation of "manager" in the command **mmlscluster**.

The below command is an example:

```
% ghicrcluster -v firefly falcon
```

After **ghicrcluster** returns "`Done.`", restart Spectrum Scale and GHI:

```
% ghishutdown -G
% ghistartup -G
```

If **ghicrcluster** fails during the configuration, retry the configuration step with the "**-r**" option after the errors from the failure are resolved (**ghicrcluster -r** [**-v**]).

5.5. Create the Spectrum Scale file systems GHI will manage

Use the command **ghiaddfs** for each file system to be created, which may be issued from any node in the cluster. File systems to be defined must *not* be mounted in Spectrum Scale when the **ghiaddfs** command is issued. The **ghiaddfs** command will supply default values for the file system which can be updated or changed with the command **ghichfs**.

For each file system, the name and mount point are supplied by the user. The ports to be used by

the associated SD and ED may also be user-supplied or left to their default values.

```
% ghiaddfs [-v] <FS_Name> [-c "# <comment>"] <Mount_Point> [<SD_Port> <ED_Port>]
```

Where:

<FS_Name>

The same as the Spectrum Scale configuration name.

<Mount_Point>

The same as the Spectrum Scale configuration mount point.

<SD_Port>

The default scheduler daemon port is 80x0, where "x" is the order in which file systems were configured; for example, 8010 for the first configured file system. GHI will assign a port if one is not specified.

<ED_Port>

The default event daemon port is 80x1, where "x" is the order in which file systems were configured; for example, 8011 for the first configured file system. GHI will assign a port if one is not specified.

The below command is an example:

```
% ghiaddfs firef1y /firef1y
```

The GHI file system name and mount point must be unique among all GHI clusters connected to the HPSS system.

5.6. Create IOMs for each GHI-managed file system

If you have multiple GHI nodes, you should create one IOM per GHI file system on each node. Each file system will use the same IOM port number across all nodes.

The default port selected for an IOM is 80x2, where "x" is the order in which the file system was configured (8012 for the first configured file system, 8022 for the second configured file system, and so on). For more details about ports, see the *GHI Management Guide*.

Install the ghi-iom, ghi-lib, and ghi-ishtar RPMs on each node that will be used as an IOM and link the GHI install location with the /opt/ghi directory.

```
% rpm -ivh ghi-lib-<version>.el#.hpss.<hpss version>.<architecture>.rpm \
  ghi-iom-<version>.el#.hpss.<hpss version>.<architecture>.rpm \
  ghi-ishtar-<version>.el#.hpss.<hpss version>.<architecture>.rpm

% ln -s /hpss_src/ghi-<version> /opt/ghi
```

In addition, each node used as an IOM will need to copy the `htar.ksh` wrapper script to `/var/hpss/hsi/bin` as outlined in the *Configure GHI-ISHTAR* section.

Refer to the *GHI Management Guide* for more details about **ghiaddiom**.

```
% ghiaddiom [-RTv] <file_system> [-c "# <comment>"] <IOM_Node>[:<Port>]
           <Active_On_Session_Node> <Est_XFER_Rate> <Chunksize>
```

Where:

<file_system>

Name of the file system added with the **ghiaddfs** command.

<IOM Node>[:<Port>]

Name of the node the IOM will run on and optionally the port number.

<Active_On_Session_Node>

Active state of the IOM on the manager session node.

<Est_XFER_Rate>

Estimated data transfer rate.

<Chunksize>

Maximum number of bytes to transfer per non-aggregate HPSS I/O request.

Example:

```
% ghiaddiom -v firefly firefly:8012 TRUE 1GB 1TB
```

5.7. Modify the xinetd.conf file for the number of IOMs

```
% vi /etc/xinetd.conf*
```

Change "`cps = 50 10`" to "`cps =` *<IOM thread pool size × number of IOMs>* `10`".

The IOM thread pool size can be obtained from **ghilsfs** *<file system>* **--iotps**.

5.8. Information Lifecycle Management (ILM) policies

GHI makes use of Spectrum Scale ILM policies. A policy is a plain-text file that describes files and directories to be included or excluded from processing. IBM provides templates which you may use as a starting point to configure custom policies. These templates can be found in the `/var/hpss/ghi/policy` directory. Below is a list of policy templates.

`migrate.policy`

This file can be placed in any directory in the system. The policy should have separate rules for aggregates and non-aggregates. The script **ghi_migrate** is invoked from the policy engine and requires a "**-a**" option to process aggregates.

`reset_incomplete_migration.policy`

Use this policy to reset files for which a migration was started but never completed. Such files will show as "`[incompletely-migrated]`" when listed with **ghi_ls -h**. They are "migrated enough" such that Spectrum Scale will not select them to be re-migrated, and the migration-reset process will result in the files being set back to "un-migrated" so that Spectrum Scale will select them in the next applicable migration policy run. This file can be placed in any directory in the system.

`recall.policy`

The recall policy does not use a bulk size. The policy generates a list. That list is parsed into aggregates and non-aggregates. The `recall.policy` file can be placed in any directory in the system.

`tape_smart_migration.policy`

This is an example used to migrate files in a tape-smart manner. Files are migrated by HPSS file families and by path name. This policy can be used in combination with the **--split-filelists-by-weight** option for **mmapplypolicy** to generate file lists that contain elements with the same WEIGHT value.

`backup_migration.policy`

The migration policy will run a full Spectrum Scale scan and will attempt to migrate any files that are not stored currently in HPSS. The policy file should be updated to reflect the migration rules used for this file system. The policy should be able to select every file that has not been migrated to HPSS and exclude any file which should not be migrated. Verify that the backup migration policy matches what is being backed up in the `backup_metadata.policy` file to ensure that files which have not been migrated are included in the metadata backup.

`backup_metadata.policy`

This policy is used by the Spectrum Scale SOBAR **mmimgbackup** command. Spectrum Scale file system namespace and file metadata are sent to GHI and HPSS.

Do not change the backup_metadata policy without contacting HPSS support.

`backup_error.policy`

The backup error policy contains the rules that are used to validate the capture of a file system's metadata.

Do not change the backup_error policy without contacting HPSS support.

`threshold.policy`

The Spectrum Scale ILM threshold policy provides the capability for GHI to space manage the Spectrum Scale file system. New and modified Spectrum Scale files are copied to HPSS on a periodic basis. When the Spectrum Scale file system reaches a predefined space threshold, the Spectrum Scale ILM threshold policy is executed to identify file candidates whose data can be removed from the file system. This file must be copied from the `/var/hpss/ghi/policy` directory to `/var/hpss/ghi/policy/`<*file system*> and modified to be specific to the file system. The script **ghi_migrate** is invoked from the policy engine and requires a "**-p**" option to punch holes in the file system.

5.9. Configure ghi_dump_fs logging configuration file(s)

GHI provides a tool **ghi_dump_fs** that allows you to dump the contents of a GHI managed file system for inspection or disaster recovery purposes. The tool produces two output files, `ghi_dump_fs.log` and `ghi_dump_fs.error.log`, and makes use of the **logrotate** tool to control log rotation and archiving.

Before running the **ghi_dump_fs** tool, a logrotate configuration file should be created for each file system you intend to run the tool on. A script **ghi_dump_fs_config_logging** is available to create the configuration file(s). The tool creates a configuration file, `ghi_dump_fs_<file_system>`, from a template file copied from the `/var/hpss/ghi/config/templates` directory to the logrotate configuration file directory. The tool uses `/etc/logrotate.d` as the logrotate configuration file directory, however, this location can be changed by specifying the `-l` option.

For each file system, run the following command:

```
% ghi_dump_fs_config_logging <file system>
```

This command will create the logrotate configuration file `ghi_dump_fs_<file system>` in the `/etc/logrotate.d` directory. By default the configuration file will keep five archived copies of the two output files created by the **ghi_dump_fs** tool. Optional arguments can be used to change the output directory where you intend to store the output logs created with the **ghi_dump_fs** tool, change the number of archived copies to retain, and configure a maximum age for the archived copies. Refer to the help option `-h` for additional information.

With the log configuration file(s) in place, when the **ghi_dump_fs** tool is executed one of the intial steps of the tool is to force the log rotation using the appropriate configuration file for the file system the tool is being run on.

6. Backup and restore

6.1. Backups

To back up a Spectrum Scale file system, use the GHI **ghi_backup** command line interface. The backup interface uses the Spectrum Scale **mmimgbackup** command, which uses the ILM policy management engine.

GHI backups use the Spectrum Scale snapshot feature to take a point-in-time image of the file system. When running a backup:

1. A snapshot of the Spectrum Scale namespace is saved after the backup migration policy and any other running migration policies have completed.
2. The state of each of the files is saved.

Each file system to be backed up uses its own copy of each of the following backup policy templates that reside in the `/var/hpss/ghi/policy` directory:

`backup_migration.policy`

The backup migration policy contains the migration rules for the Spectrum Scale file system to be backed up. The rules can migrate files as aggregates or non-aggregates. The rules must select all the files to be backed up.

`backup_metadata.policy`

The backup metadata policy contains the rules that previous GHI versions need to capture a file system's metadata. The new image backup feature does not require a metadata policy for metadata backup. The metadata is contained in the image generated by Spectrum Scale as part of the backup process.

`backup_error.policy`

The backup error policy contains the rules that are used to validate the capture of the file system's metadata.

IBM recommends running a daily backup. Checking the backup logs daily to correct any errors is a good practice to ensure successful backups. The GHI backup option is "image". Full non-image backup is deprecated. Details of backup are provided in the *GHI Management Guide*.

Most sites create a crontab entry using either of the following forms of **ghi_backup** to run a daily backup. For purposes of this documentation, each command below has been split into two separate lines; each indented line should be considered as part of the command line above it, with no breaks.

```
ghi_backup <filesys> <type> [--keep-snapshot] [-E <Fsets> [-U
  <mmapplypolicy_user_args>] [-I <mmimgbackup_user_args>]]

ghi_backup <filesys> <type> [--keep-snapshot] [-E -F <List of Fsets> [-U
  <mmapplypolicy_user_args>] [-I <mmimgbackup_user_args>]]
```

Where:

<type>

Perform the indicated type of backup. Presently, the only value accepted for this field is "image", which means an image backup using the Spectrum Scale SOBAR capability.

--keep-snapshot

Keep the Spectrum Scale snapshot that is taken during backup.

<Fsets>

Filesets which need not be linked in; either a space-separated list of filesets, or (using **-F**) a space-separated list of files containing a list of filesets.

<mmapplypolicy_user_args>

Arguments to be passed into **mmapplypolicy** only.

Example command:

```
% ghi_backup firefly image
```

6.2. Restore

Refer to "Backup and recovery" in the *GHI Management Guide* for restore details.

7. GHI conversions

Make sure that for any version you upgrade to, you read through all the instructions from the version you are upgrading from up until the version you are installing.

7.1. Converting to 4.2

As a part of installing GHI 4.2, make sure that you review the service files for your IOMs. You should update them to add the

LimitNOFILE=500000

directive under the "[Service]" section. For example:

```
[Service]
Restart=always
ExecStart=/opt/ghi/bin/ghi_iom ghi_fs02 8022
LimitNOFILE=500000
```

The open file limit should be sufficient but may need to be tuned.

7.2. Converting from 3.3.0 to 4.1.0

7.2.1. Configuring GHI for a different cluster user

As part of GHI 4.1 each GHI cluster can interface with HPSS as a unique user. In GHI 3.3 and below it was assumed that the user HPSS_PRINCIPAL_DMG ("hpssdmg") was used for all clusters.

The following steps show how to change a preexisting GHI cluster from one user to a newly created user.

On a GHI node in the cluster:

1. Get the names of the file systems configured on this cluster

```
% ghilsfs
```

2. Stop GHI on the cluster

<HPSS_BASE_PATH> is the "HPSS Base Path" output by ghilsfs. The default location is "/ghi".

On the HPSS core server:

1. Add new user to HPSS core with hpssuser tool. Refer to the *HPSS Installation Guide* for specific instructions.
2. Use the hpss_recurse tool with the chmod_chown_callback.py callback on our ghi file systems stored in HPSS. The file systems are stored in HPSS at the path /<HPSS_BASE_PATH>/<ghi file system name>
   a. For every ghi file system run the following:

```
% hpss_recurse -b /<HPSS_BASE_PATH>/<ghi file system name> -c
/opt/hpss/tools/lib/py-packages/chmod_chown_callback.py -C "-o <new user uid> -g
<new user gid>" -n <number of threads to use>
```

hpss_recurse with the -c argument and chmod_chown_callback.py callback will change the ownership of every file and directory in /<HPSS_BASE_PATH>/<ghi file system name> to the user <new user uid> and group <new user gid>. The files will have the permissions to only allow owner read and write (Frw-------, i.e 0600) and directories will have owner read, write, execute permissions (Drwx------, i.e 0700). For more details on the hpss_recurse tool see the man page.

On each GHI node in the cluster:

1. Add new user, The user and group ID numbers created on the GHI nodes must match the corresponding user and group ID numbers on the HPSS Core Server. Refer to the *HPSS Management Guide* for specific instructions.

2. Modify `/var/hpss/etc/env.conf` file to have the environment variable `HPSS_PRINCIPAL_DMG` set to the new user principal name created on the GHI cluster nodes.
3. Modify `/var/hpss/etc/env.conf` file to have the environment variable `HPSS_PRINCIPAL_SSM` set to hpssssm.
4. Modify `/var/hpss/hsi/bin/htar.ksh` file to use new user as the HPSS_PRINCIPAL.

```
Change <GHI_CLUSTER_USER> to the user configured for this cluster.

if [ "$HPSS_PRINCIPAL" = "" ]; then
  if [ "$HPSS_AUTH_METHOD" = "kerberos" ]; then
      export HPSS_PRINCIPAL=`whoami`@$DEFAULT_REALM
  else
     if [ "`whoami`" = "root" ]; then
        export HPSS_PRINCIPAL=<GHI_CLUSTER_USER>
     else
        export HPSS_PRINCIPAL=`whoami`
     fi
```

On the HPSS core server change the ownership and permissions of the HPSS base path for ghi.

1. hpsschown <GHI_CLUSTER_USER> <HPSSSRVR_GID> <HPSS_BASE_PATH>
2. hpsschmod 775 <HPSS_BASE_PATH>

It does not matter which cluster user owns the HPSS base path just that all of the cluster users belong to the hpsssrvr group.

On the HPSS core server if the hpssdmg user is no longer required, delete it:

1. Remove the HPSS_PRINCIPAL_DMG user from env.conf
2. Remove ACLs for HPSS_PRINCIPAL_DMG user from core server Account Validation Interface

```
% /opt/hpss/bin/hpss_server_acl

hsa> acl -t CORE
1) PVL Mount Notification Interface (v1) 007ff347-e533-1c
2) Realtime Monitor Interface (v1) 80c9a256-2f13-11
3) Client Interface (v2) 32ba9692-4667-11
4) Account Validation Interface (v1) 647f22a8-a1e9-11
Select an interface
Choose an item by number (RET to cancel):
> 4
hsa> show

perms - type - ID (name) - realm ID (realm)
=========================================
r--c--- - user - 302 (hpssftp) - 10000 (<core_server>.clearlake.ibm.com)
r--c--- - user - 306 (hpssfs) - 10000 (<core_server>.clearlake.ibm.com)
rw-c-dt - user - 307 (hpssmps) - 10000 (<core_server>.clearlake.ibm.com)
rw-c-d- - user - 312 (hpssssm) - 10000 (<core_server>.clearlake.ibm.com)
rw-c--- - user - 1001 (hpssdmg) - 10000 (<core_server>.clearlake.ibm.com)
------t - any_other

hsa> del user hpssdmg rwc
hsa> show

perms - type - ID (name) - realm ID (realm)
=========================================
r--c--- - user - 302 (hpssftp) - 10000 (<core_server>.clearlake.ibm.com)
r--c--- - user - 306 (hpssfs) - 10000 (<core_server>.clearlake.ibm.com)
rw-c-dt - user - 307 (hpssmps) - 10000 (<core_server>.clearlake.ibm.com)
rw-c-d- - user - 312 (hpssssm) - 10000 (<core_server>.clearlake.ibm.com)
rw-c--- - user - 1001 (hpssdmg) - 10000 (<core_server>.clearlake.ibm.com)
------t - any_other

hsa> quit
```

3. Remove the HPSS_PRINCIPAL_DMG with hpssuser

```
% hpssuser -del hpssdmg -acct
```

On a GHI node in the cluster:

1. Startup GHI
2. Verify that new files are created with our newly configured user and that GHI can access preexisting files.

On the GHI nodes in the cluster:

1. If the hpssdmg user is no longer required, delete it

7.3. Converting from 3.1.0 to 3.2.0

The conversion process will use RPMs for GHI 3.2.0.

1. Modify the callback script to source the database profile on all GHI nodes. (DB2PROF)

```
% vim /opt/ghi/bin/hpssEventNotify
```

7.4. Converting from 3.0.1 to 3.1.0

The conversion process will use RPMs for GHI 3.1.0.

1. Shut down GHI.

```
% ghishutdown -g
```

2. Verify there are no non-IOM GHI processes in the process list.

```
% ps -ef | grep ghi | grep -v ghi_iom
```

If any GHI processes remain in the list, use **kill -9** <*pid*> to shut them down.

3. Unmount the file system.

```
% mmumount <file system> -a
```

4. Remove old GHI RPMs.

```
% rpm -qa | grep ghi - List installed RPMs
% rpm -e <ghi_rpms installed from above>
% rpm -qa | grep ghi - Verify no GHI RPMs are installed
```

5. Install new GHI 3.1.0 RPMs.

RPMs deployed will vary. Contact HPSS support before installing.

```
% cd <path to GHI 3.1.0 RPMs>
% rpm -ivh ghi-3.1.0.0-0.<arch> ghi-lib-3.1.0.0-0.<arch> ghi-ishtar-5.1.2.0-
0.<arch>
```

6. Run **ghiupdate** twice: first with the **-vT** option to test, and then again with just the **-v** option.

```
% ghiupdate -vT --<OS arch> <all nodes listed in ghilsnodes>

% ghiupdate -v --<OS arch> <all nodes listed in ghilsnodes>
```

7. Make sure the directory where GHI is installed is linked to `/opt/ghi`.
8. Delete and recreate the mmcallbacks so they use the `/opt/ghi/bin` path. For example, if the mmcallbacks are:

```
% mmlscallback

hpssCBstartup
    command       = /opt/hpss/bin/hpssEventNotify
    event         = startup,clusterManagerTakeover,preShutdown
    parms         = %clusterName %eventName %clusterManager.ip %myNode.ip

hpssCBthreshold
    command       = /opt/hpss/bin/hpssCBthreshold
    event         = noDiskSpace,lowDiskSpace
    parms         = %eventName %fsName
```

   You should do the following to re-add the callbacks for the `/opt/ghi` directory:

   a. Delete the callbacks with:

```
% mmdelcallback hpssCBstartup

% mmdelcallback hpssCBthreshold
```

   b. Add them back for `/opt/ghi` with the following. For purposes of this documentation, the **mmaddcallback** commands below have been split into multiple lines; the indented lines should be considered as part of the starting command line, with no breaks.

```
% mmaddcallback hpssCBstartup --command /opt/ghi/bin/hpssEventNotify --event
   startup,clusterManagerTakeover,preShutdown --parms "%clusterName %eventName
   %clusterManager.ip %myNode.ip"

% mmaddcallback hpssCBthreshold --command /opt/ghi/bin/hpssCBthreshold --event
   noDiskSpace,lowDiskSpace --parms "%eventName %fsName"
```

9. Delete and re-add the IOMs so they are configured for the new location under the `/opt/ghi` directory. The following instructions can help achieve this:

   a. Start GHI.

   b. List all IOMs.

```
% ghilsiom <file system>
```

You can use this to see the current IOM configuration info.

c. Delete all IOMs.

```
% ghideliom <file system> <IOM node>:<port #>
```

Use this on each IOM node you want recreated on the file system.

d. Add all IOMs.

```
% ghiaddiom <file system> <IOM node>:<port #> <asn value> <etr value> <chunk
size>
```

Where:

<***asn value***>
: the value indicated earlier for the "Active Session Node" in the **ghilsiom** output.

<***etr value***>
: the value indicated earlier for the "Estimated Transfer Rate" in the **ghilsiom** output.

<***chunk size***>
: the value indicated earlier for the "Transfer Chunk Size" in the **ghilsiom** output.

Repeat the above for each IOM and port in **ghilsiom**.

10. Restart GHI.

```
% ghishutdown -g

% ghistartup -g
```

11. The new IOM should be usable and you can confirm the change by running **ghilsiom** <*file system*>.
12. You may wish to update your path to include `/opt/ghi/bin` so that the GHI executables can be found without specifying the path.
13. The files installed by the GHI RPM that specify GHI paths will use `/opt/ghi` instead of `/opt/hpss`. However, for any policy files containing GHI paths that already exist on the system that are not replaced by the GHI RPM, these will need to be updated so paths containing `/opt/hpss` are changed to `/opt/ghi`.
14. Old libraries must be removed.

As part of upgrading to versions of GHI 3.1.0 or later from GHI 3.0.1 or earlier, sites should remove the GHI files from the HPSS directory. These include the `ghi*` files under `/opt/hpss/bin` and `/opt/hpss/lib`, but also these files under `/opt/hpss/lib`:

```
HPSSModule.so
libhpssghi.so
libhpssghi_base.so
```

GHI will attempt to create a link in `libhpssghi_restore.so` in `/opt/hpss/lib`. Spectrum Scale continues to link with that location for handling image restores. The `libhpssghi_restore.so` file should be removed.

and these files under `/opt/hpss/bin`:

```
db2db.py
dmapishell
hpssAddCallbacks
hpssCBthreshold
hpssEventNotify
hpssdelete
hpsslist
hpssmigrate
hpssrecall
lsghi
lsgpfs
```

15. Policy files must be updated to reflect the new install location.

    Also, customers will have to update *all policy files* on all their nodes manually to reflect `/opt/ghi/bin` instead of `/opt/hpss/bin`. Until this is done, migrations, recalls, and stages will not work.

7.5. Converting from 3.0.0 to 3.0.1

The conversion process will use RPMs for GHI 3.0.1.

1. Shut down GHI.

```
% ghishutdown -g
```

2. Verify there are no non-IOM GHI processes in the process list.

```
% ps -ef | grep ghi | grep -v ghi_iom
```

If any GHI processes remain in the list, use **kill -9** <*pid*> to shut them down.

3. Unmount the file system.

```
% mmumount <file system> -a
```

4. Remove old GHI RPMs.

```
% rpm -qa | grep ghi - List installed RPMs
% rpm -e <ghi_rpms installed from above>
% rpm -qa | grep ghi - Verify no GHI RPMs are installed
```

5. Install new GHI 3.0.1 RPMs.

RPMs deployed will vary. Contact HPSS support before installing.

For purposes of this documentation, the **rpm** command below has been split into two separate lines; the indented line should be considered as part of the command line above it, with no breaks.

```
% cd <path to GHI 3.0.1 RPMs>
% rpm -ivh ghi-3.0.1.0-0.<arch>, ghi-lib-3.0.1.0-0.<arch>,
   ghi-ishtar-5.1.2.0-0.<arch>
```

6. Run **ghiupdate** twice: first with the **-vT** option to test, and then again with just the **-v** option.

```
% ghiupdate -vT --<OS arch> <all nodes listed in ghilsnodes>

% ghiupdate -v --<OS arch> <all nodes listed in ghilsnodes>
```

7. Convert the GHI garbage collection table on each file system. This step should be run for all file systems; the conversion of file system tables can be executed in parallel.

   a. For each file system, modify `/var/hpss/ghi/templates/ghimodifygc.ddl` by replacing the template info (<GC_FILESYSTEM>) with your file system name throughout the file (for example, if your file system name is "foo", the table name should be "GC_FOO").

```
% cat /var/hpss/ghi/templates/ghimodifygc.ddl

connect to hghi;
ALTER TABLE HPSS."GC_FOO_FS1"
 ADD COLUMN INODE
 BIGINT NOT NULL DEFAULT 0
 ADD COLUMN IGEN
 INTEGER NOT NULL DEFAULT 0;

reorg table HPSS."GC_CANAAN_FS1";
```

b. Run the modified `ghimodifygc.ddl`.

```
% db2 -svtf ghimodifygc.ddl
```

8. Start the new GHI.

```
% ghistartup -g
```

9. Verify that the ghi_pm, ghi_cm, and ghi_md processes are running.

```
% ps -ef | grep ghi_ | grep -v grep
```

10. Mount GHI-managed file systems.

```
% mmmount <file system> -a
```

11. Verify the SD, ED, and then the IOMs are active.

```
% tail /<file system>/scratch/mon/<mon_sd.out | mon_iom.out>
```

12. Convert the DMAPI PIN attributes into timestamp values *for all file systems*.

The pin conversion process converts DMAPI PIN attributes into timestamp values for all file systems. If Spectrum Scale is restored using a backup prior to this conversion, the unconverted DMAPI PIN attributes will be restored as well, and it will be necessary to rerun these steps to convert the PIN attribute values again.

a. List all pinned files using the following policy:

```
# Template for listing old style pinned files

RULE EXTERNAL POOL 'hsm' EXEC '/opt/hpss/bin/ghi_migrate'


RULE 'Pinned Files' LIST 'files_pinned'
                    SHOW ('-s' FILE_SIZE)
                    WHERE XATTR('dmapi._GHI_PIN') LIKE 'TRUE%'
                    AND path_name NOT LIKE '%/%scratch%'
                    AND path_name NOT LIKE '%/%.snapshot%'

RULE 'Default' SET POOL 'system'
```

b. Apply the above policy.

```
% ghiapplypolicy <file system> -P <above template file> -I defer
```

c. Generate a file list for all pinned files and convert them.

```
% cat <path/to/scratch/.ghi>/list.files_pinned | cut -d' ' -f7 >
/tmp/pinned_files
% ghi_pin -f /tmp/pinned_files

# Policy template example

RULE EXTERNAL POOL □hsm□ EXEC □/opt/hpss/bin/ghi_migrate□
RULE □UnPinned Files□ LIST □files_unpinned□
                    SHOW (□-s□ FILE_SIZE)
                    WHERE XATTR(□dmapi._GHI_PIN□) LIKE □FALSE%□
                    AND path_name NOT LIKE □%/%scratch%□
                    AND path_name NOT LIKE □%/%.snapshot%□
RULE □Default□ SET POOL □system□
```

d. Generate a file list for all unpinned files (files which were previously pinned, but were later unpinned using the **ghi_pin** tool) and convert them. For purposes of this documentation, the **cat** command below has been split into two separate lines; the indented line should be considered as part of the command line above it, with no breaks.

```
% cat <path/to/scratch/.ghi>/list.files_unpinned | cut -d' ' -f7 >
   /tmp/unpinned_files
% ghi_pin -u -f /tmp/unpinned_files
```

e. Validate that all attributes have been migrated by listing the pinned files and comparing row counts: they should match. If not, list the pinned files again and rerun the **ghi_pin** tool against the resulting list. For purposes of this documentation, the **wc** command below has been split into two separate lines; the indented line should be considered as part of the

command line above it, with no breaks.

```
% ghiapplypolicy -P /var/hpss/ghi/policy/pin_time_list.policy -I defer
% wc -l <path/to/scratch/.ghi>/list.files_pinned
   </path/to/scratch/.ghi>/list.files_pinned_time
1000 <path/to/scratch/.ghi>/list.files_pinned
1000 <path/to/scratch/.ghi>/list.files_pinned_time
```

It could take up to ten minutes for the ED to connect to the SD and fifteen minutes for the IOMs to connect to the ED. If after twenty minutes things are still not connecting, contact HPSS support.

7.6. Converting from 3.1 to 3.2

The conversion process will use RPMs for GHI 3.2.

1. Shut down GHI.

   ```
   % ghishutdown -g
   ```

2. Verify there are no non-IOM GHI processes in the process list.

   ```
   % ps -ef | grep ghi | grep -v ghi_iom
   ```

   

   If any GHI processes remain in the list, use **kill -9** <*pid*> to shut them down.

3. Unmount the file system.

   ```
   % mmumount <file system> -a
   ```

4. Remove old GHI RPMs.

   ```
   % rpm -qa | grep ghi - List installed RPMs
   % rpm -e <ghi_rpms installed from above>
   % rpm -qa | grep ghi - Verify no GHI RPMs are installed
   ```

5. Install new GHI 3.2 RPMs on all session nodes. Install new GHI 3.2 IOM RPMs on all IOM nodes.

   

   RPMs deployed will vary. Contact HPSS support before installing.

```
% cd <path to GHI 3.2 RPMs>
% rpm -ivh ghi-3.2.0.0-0.<arch> ghi-lib-3.2.0.0-0.<arch> ghi-ishtar-5.1.2.0-
0.<arch>
```

6. Run **ghiupdate** twice: first with the **-vT** option to test, and then again with just the **-v** option.

```
% ghiupdate -vT --<OS arch> <all nodes listed in ghilsnodes>

% ghiupdate -v --<OS arch> <all nodes listed in ghilsnodes>
```

7. Create indexes on the GHI mapping tables for each file system. These steps should be run for all file systems; the conversion of the system tables can be executed in parallel.
   a. For each file system, source the Db2 profile.

   ```
   % source <hpssdb user home>/sqllib/db2profile
   ```

   b. Create the HPSS path index.

   ```
   % db2 "CREATE INDEX HPSSIDX ON MAPPING_<FS>_BASE (HPSSPATH)"
   ```

   c. Create the Spectrum Scale path index.

   ```
   % db2 "CREATE INDEX GPFSIDX ON MAPPING_<FS>_BASE (GPFSPATH)"
   ```

   d. Create the Inode index.

   ```
   % db2 "CREATE INDEX INODEIDX ON MAPPING_<FS>_BASE (INODE)"
   ```

   e. Update statistics in the system catalog for the GHI mapping base table.

   ```
   % db2 "RUNSTATS ON TABLE MAPPING_<FS>_BASE AND SAMPLE DETAILED INDEXES ALL"
   ```

   f. Remove any cached SQL statements.

   ```
   % db2 "FLUSH PACKAGE CACHE DYNAMIC"
   ```

   g. Check if the mapping table needs to be reorganized.

   ```
   % db2 "REORGCHK CURRENT STATISTICS ON TABLE MAPPING_<FS>_BASE"
   ```

8. Create a new index on the GHI garbage collection table for each file system. This step should be run for all file systems; the conversion of the system tables can be executed in parallel.

   a. For each file system, source the Db2 profile.

   ```
   % source <hpssdb user home>/sqllib/db2profile
   ```

   b. Create the SOID Index for the GC table.

   ```
   % db2 "CREATE INDEX GC_<FS>_SOID_INDEX ON GC_<FS> (SOID)"
   ```

9. Start the new GHI.

   ```
   % ghistartup -g
   ```

10. Verify that the ghi_pm, ghi_cm, and ghi_md processes are running.

    ```
    % ps -ef | grep ghi_ | grep -v grep
    ```

11. Mount GHI-managed file systems.

    ```
    % mmmount <file system> -a


                        AND path_name NOT LIKE '%/%.snapshot%'
     RULE 'Default' SET POOL 'system'
    ```

    a. Generate a file list for all unpinned files (files which were previously pinned, but were later unpinned using the **ghi_pin** tool) and convert them. For purposes of this documentation, the **cat** command below has been split into two separate lines; the indented line should be considered as part of the command line above it, with no breaks.

    ```
    % cat <path/to/scratch/.ghi>/list.files_unpinned | cut -d' ' -f7 >
       /tmp/unpinned_files
    % ghi_pin -u -f /tmp/unpinned_files
    ```

    b. Validate that all attributes have been migrated by listing the pinned files and comparing row counts: they should match. If not, list the pinned files again and rerun the **ghi_pin** tool against the resulting list. For purposes of this documentation, the **wc** command below has been split into two separate lines; the indented line should be considered as part of the command line above it, with no breaks.

```
% ghiapplypolicy -P /var/hpss/ghi/policy/pin_time_list.policy -I defer
% wc -l <path/to/scratch/.ghi>/list.files_pinned
   </path/to/scratch/.ghi>/list.files_pinned_time
1000 <path/to/scratch/.ghi>/list.files_pinned
1000 <path/to/scratch/.ghi>/list.files_pinned_time
```

It could take up to ten minutes for the ED to connect to the SD and fifteen minutes for the IOMs to connect to the ED. If after twenty minutes things are still not connecting, contact HPSS support.

Appendix A: Glossary of terms and acronyms

ACL	Access Control List.
AIX	Advanced Interactive Executive. An operating system provided on many IBM machines.
API	Application Program Interface.
Archive	One or more interconnected storage systems of the same architecture.
Attribute	When referring to a managed object, an attribute is one discrete piece of information, or set of related information, within that object.
Class of Service	A set of storage system characteristics used to group files with similar logical characteristics and performance requirements together. A Class of Service is supported by an underlying hierarchy of storage classes.
CM	Configuration Manager
Co-managed	File data resides in both Spectrum Scale and HPSS.
Configuration	The process of initializing or modifying various parameters affecting the behavior of a GHI server or infrastructure service.
COS	Class of Service.

Core Server	An HPSS server which manages the namespace and storage for an HPSS system. The Core Server manages the namespace in which files are defined, the attributes of the files, and the storage media on which the files are stored. The Core Server is the central server of an HPSS system. Each storage subsystem uses exactly one Core Server.
Daemon	A UNIX program that runs continuously in the background.
Db2	A relational database system, a product of IBM Corporation, used by HPSS and GHI to store and manage HPSS and GHI metadata.
Directory	An HPSS object that can contain files, symbolic links, hard links, and other directories.
DMAPI	Data Management Application Programming Interface.
DOE	Department of Energy.
ED	Event Daemon.
Event	A log record message type used to log informational messages (for example: subsystem starting, subsystem terminating).
File	An object that can be written to, read from, or both, with attributes including access permissions and type, as defined by POSIX (P1003.1-1990). HPSS supports only regular files.
Fileset	A collection of related files that are organized into a single easily managed unit. A fileset is a disjoint directory tree that can be mounted in some other directory tree to make it accessible to users.
File system ID	A 32-bit number that uniquely identifies an aggregate.
FSID	File system unique identifier.
GB	Gigabyte (2^{30}).
GC	Garbage Collection.
GHI	Spectrum Scale/HPSS Interface.
ISHTAR	Specially modified GHI-specific version of the HTAR program.
HA	High Availability

Hierarchy	See "Storage Hierarchy".
HPSS	High Performance Storage System.
HSI	Hierarchical Storage Interface.
HSM	Hierarchtical Storage Management
ISHTAR	HPSS tar program – a utility to aggregate a set of files directly into HPSS without first writing to local storage, and to randomly retrieve individual member files via creation of a separate index file.
IBM	International Business Machines Corporation.
ID	Identifier.
ILM	Information lifecycle Management.
I/O	Input/Output.
IOM	I/O Manager.
IP	Internet Protocol.
KB	Kilobyte (2^{10}).
LAN	Local Area Network.
LANL	Los Alamos National Laboratory.
LLNL	Lawrence Livermore National Laboratory.
MB	Megabyte (2^{20}).
MD	Mount Daemon.
metadata	Control information about the data stored under HPSS, such as location, access times, permissions, and storage policies. Most HPSS metafile contents are stored in a Db2 relational database.
Migrate	To copy file data from a level in the file's hierarchy onto the next lower level in the hierarchy.

Mount	An operation in which a cartridge is either physically or logically made readable or writable, or both, on a drive. In the case of tape cartridges, a mount operation is a physical operation. In the case of a fixed disk unit, a mount is a logical operation.
Mount point	A place where a fileset is mounted in the XFS or HPSS namespaces, or both.
Mover	An HPSS server that provides control of storage devices and data transfers within HPSS.
Namespace	The set of name-object pairs managed by the HPSS Core Server.
PB	Petabyte (2^{50}).
PID	Process identifier
PM	Process Manager.
POSIX	Portable Operating System Interface (for computer environments).
RPM	RPM Package Manager
SD	Scheduler Daemon.
SNL	Sandia National Laboratories.
SOID	Storage Object Identifier
Spectrum Scale	New name of General Parallel File System (GPFS).
SQL	Structured Query Language
SSH	Secure Shell Protocol
Storage class	An HPSS object used to group storage media together to provide storage for HPSS data with specific characteristics. The characteristics are both physical and logical.

Storage hierarchy	An ordered collection of storage classes. The hierarchy consists of a fixed number of storage levels numbered from level 1 to the number of levels in the hierarchy, with the maximum level being limited to 5 by HPSS. Each level is associated with a specific storage class. Migration and stage commands result in data being copied between different storage levels in the hierarchy. Each Class of Service has an associated hierarchy.
Storage subsystem	A portion of the HPSS namespace that is managed by an independent Core Server and (optionally) the Migration/Purge Server.
TB	Terabyte (2^{40}).
TCP/IP	Transmission Control Protocol/Internet Protocol.