GHI Admin Guide

14 May 2026

Table of Contents

Spectrum Scale (GPFS) - HPSS Interface Release 4.3.0.0.0, 14 May 2026

Copyright Notification

Copyright © 2008-2026 International Business Machines Corporation, The Regents of the University of California, Triad National Security, LLC, Lawrence Livermore National Security, LLC, National Technology & Engineering Solutions of Sandia, LLC, and UT-Battelle.

All rights reserved.

Portions of this work were produced by Lawrence Livermore National Security, LLC, Lawrence Livermore National Laboratory (LLNL) under Contract No. DE-AC52-07NA27344 with the U.S. Department of Energy (DOE); by the University of California, Lawrence Berkeley National Laboratory (LBNL) under Contract No. DE-AC02-05CH11231 with DOE; by Triad National Security, LLC, Los Alamos National Laboratory (LANL) under Contract No. 89233218CNA000001 with DOE; by National Technology & Engineering Solutions of Sandia, LLC (NTESS), Sandia National Laboratories (SNL) under Contract No. DE-NA0003525 with DOE; and by UT-Battelle, Oak Ridge National Laboratory (ORNL) under Contract No. DE-AC05-00OR22725 with DOE. The U.S. Government has certain reserved rights under its prime contracts with the Laboratories.

DISCLAIMER

Portions of this software were sponsored by an agency of the United States Government. Neither the United States, DOE, The Regents of the University of California, Triad National Security, LLC, Lawrence Livermore National Security, LLC, National Technology & Engineering Solutions of Sandia, LLC, UT-Battelle, nor any of their employees, makes any warranty, express or implied, or assumes any liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights.

Trademark Usage

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

IBM is a registered trademark of International Business Machines Corporation.

IBM, Db2, Db2 Universal Database, AIX, RISC/6000, pSeries, and xSeries are trademarks or registered trademarks of International Business Machines Corporation.

UNIX is a registered trademark of the Open Group.

Linux is a registered trademark of Linus Torvalds in the United States and other countries.

Kerberos is a trademark of the Massachusetts Institute of Technology.

DST is a trademark of Ampex Systems Corporation.

Other brands and product names appearing herein may be trademarks or registered trademarks of third parties.

Who Should Read This Book

The GHI Admin Guide is intended as a resource for GHI administrators. For those performing the initial configuration for a new GHI system, Chapter 1 provides a configuration roadmap. For both

new systems and those upgraded from a previous release, Chapter 1 provides a configuration, operational, and performance checklist which should be consulted before bringing the system into production. The remaining chapters contain the details for configuring, reconfiguring, monitoring, and managing a GHI system.

Conventions Used in This Book

Example commands that should be typed at a command line will be proceeded by a percent sign ("%") and be presented in a 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).

Chapter 1. Release 4.3

This chapter summarizes GHI changes for Release 4.3 into four categories: new features, retired features, deprecated features, and changed features.

1.1. New features in 4.3

New features in 4.3 are described below.

1.1.1. Dry run with ghi_stage

ghi_stage(7) now supports a dry run option (-r). The dry run option does not stage any data, it counts bytes and files and reports them. See the ghi_stage(7) man page for more details.

1.1.2. Callout with ghi_stage

ghi_stage(7) now supports a callout option (-C). The callout option allows ghi_stage to "call out" to a user-provided script for every file successfully staged. This can be used to automate workflows in response to stage completion, such as pinning files. See the man page for details.

1.1.3. Checksum on Migrate

A new GHI FS option was added, Checksum on Migrate (--com). Recently, an option was added for GPFS users to add a checksum to a file (see: Data integrity during migration), which would be validated by GHI on migration. This requires the user to set the checksum.

With Checksum on Migrate, GHI enables this for a full file system and will enable generation of checksums on migration if they don't already exist. This enables file data to be generated in the GHI IOM, which will validate the data as it moves from the IOM to HPSS and is further migrated down to tape. These checksums will not be visible to GPFS through xattrs.

1.1.4. Logging with Multiple FS

GHI 4.3 has logging changes to enable better logging with multiple filesystems. GHI processes which act on specific filesystems now have the FS name included in the syslog identifier.

See Configuring Logging with Multiple Filesystems for details and instructions on setup.

Note that some processes work across filesystems, and will log with an "all" identifier.

1.1.5. Improved GPFS Recall Storm Behavior

For sites which use GPFS snapshots, there is a new tool which prevents "GPFS recall storms" from hitting GHI when snapshots exist and files are being changed on the GPFS file system.

See the Mitigating Snapshot Recall Requests section for details.

1.1.6. Garbage Collection Dump Tool

There is a new tool which enables listing the garbage collection table (ghi_dumpgc(7)). This tool can be used to retrieve listings from the GHI garbage collection table, including filtering by certain criteria.

1.1.7. Undelete Tool

There is a new tool, ghi_undelete_file(7), which admins can use to restore a GHI file from HPSS. This tool re-links HPSS data with a new file in GPFS, which appears as if it were migrated. This can be used to recover GPFS files which were accidentally deleted, so long as they have not been deleted from HPSS.

1.2. Features retired in 4.3

The tool **ghiconvertnames** had been deprecated, and was removed.

1.3. Features deprecated in 4.3

None.

1.4. Changes to existing GHI features in 4.3

- The **GHI Installation Guide** and **GHI Management Guide** have been combined.
- **ghi_ls -e** now reports the file hash information of the file in HPSS, if any exists.
- The GHI backup process now runs image files through the GPFS image file verification tool to validate that it was generated correctly. This is also done as part of the ghi_backup_manager(7) validation process and the restore process.

Chapter 2. Preparing for GHI install

GHI is made to work with the IBM cluster file system (IBM Storage Scale). This software may be referred to as either Spectrum Scale or GPFS within this document.

Prior to installing GHI, ensure you understand the customer's requirements, which will help you properly size and configure the GHI system. This 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* section for information about managing the system.

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.

2.1. 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.

2.2. Operating system

2.2.1. 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
```

2.2.2. 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
```

2.3. Configuring Logging with Multiple Filesystems

GHI processes which act on specific filesystems have the filesystem name included in their syslog identifier.

To configure separate logging with multiple filesystems, use rsyslog to key off the filesystem name identifier when splitting messages.

An example log for a filesystem named mygpfs looks like this:

Dec 10 12:36:28 my_ghi_system ghi_ed(mygpfs)[146114]: DEBUG ed_main.c[SignalHandler:986:00000000000000000000000000000000] mygpfs: Purging creds

To configure logging, update your rysyslog configuration like:

```
if $programname contains 'mygpfs' then {

  $template HPSSTemplate,"%TIMESTAMP%.%TIMESTAMP:::date-subseconds%
%syslogtag%%msg:::sp-if-no-1st-sp%%msg:::drop-last-lf%\n"

  # Put everything to the hydra2.log
  *.*        /var/hpss/log/hydra2.log; HPSSTemplate

  #
  # It is often useful to see alarms and events in the default syslog
  # message file. This can be commented out if this is undesirable.
  # The template here can also be modified as needed.
  #
  *.=alert;*.=crit;*.=err;*.=warning;*.=notice /var/log/messages; HPSSTemplate

  #
  # Do not forward this message on further.
  # This avoids logging certain messages to /var/log/messages twice.
  #
  *.*        stop
}
```

This will ensure that your messages for mygpfs get directed to the appropriate log file. Note that a number of GHI processes (e.g. ghi_cm, ghi_pm, ghi_md) work across filesystems. They will log with the "all" identifier.

Configure each GHI filesystem with its own rsyslog rule directing logs to its own log file.

2.4. 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(7), and ghi_ls(7). Install the memcached and libmemcached-devel RPMs from the RHEL software distribution on each machine you want memcached to run to improve operations.

2.4.1. 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
```

2.4.2. Configure memcached

2.4.2.1. Memcached service setup

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
```

2.4.2.2. GHI memcached configuration

On each GHI node, configure GHI to use memcached. Locate the GHI memcached config template (`/var/hpss/ghi/config/templates/memcached.conf.template`). Copy it to the GHI node in `/var/hpss/ghi/etc/memcached.conf`. Update the template file, where each --SERVER line should refer to a GHI node where memcached is running along with the configured port number.

2.4.2.3. Verifying memcached configuration in GHI

If logs like "memcached_pool() returns NULL" are seen on the IOMs, then memcached is not configured correctly and is not in use. This can impact the performance of operations such as backup and restore processing. Check the memcached service, and the GHI memcached configuration file. After addressing any issues, restart the IOM.

2.5. GHI-ISHTAR

Before installing GHI-ISHTAR, verify that prerequisites hpss-lib, hpss-lib-devel, and hpss-clnt are installed. ISHTAR is bundled with GHI as ghi-ishtar-<ghi-version>.0-0.

```
% 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
```

2.6. 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.

Chapter 3. Spectrum Scale

3.1. Install Spectrum Scale

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

3.2. 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
```

3.3. 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.

3.4. 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.
```

3.4.1. 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
```

3.4.2. 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. 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.
```

6. 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.

7. 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**

8. 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
```

3.4.3. 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.

Chapter 4. Db2

4.1. 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 arbitrary, 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.

4.1.1. 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 Admin 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
```

4.2. 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.

4.2.1. 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
```

4.2.2. 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"
```

4.2.3. 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
```

4.3. 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.

4.4. 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.
```

Chapter 5. 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.

5.1. 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.

5.2. 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
   ```

Chapter 6. GHI installation and configuration

6.1. 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/man7
/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
```

6.1.1. 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
```

6.2. GHI users and groups

All authentication and authorization are done using the GHI cluster principal. Each GHI cluster should use a unique principal. 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.

6.3. Configure GHI

GHI is configured using command-line tools. All the GHI commands discussed in this section are fully documented in the man pages section.

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.

6.4. 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(7)

command. The ghicrcluster(7) 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(7) returns "`Done.`", restart Spectrum Scale and GHI:

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

If ghicrcluster(7) fails during the configuration, retry the configuration step with the "**-r**" option after the errors from the failure are resolved (**ghicrcluster -r** [**-v**]).

6.5. Create the Spectrum Scale file systems GHI will manage

Use the command ghiaddfs(7) 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(7).

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.

6.6. 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 GHI IOM configuration.

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 *Configure GHI-ISHTAR*.

Example:

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

See ghiaddiom(7) for more details about its use.

6.7. 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(7) <*file system*> **--iotps**.

6.8. 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(7) 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(7) is invoked from the policy engine and requires a "**-p**" option to punch holes in the file system.

6.9. 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 initial 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.

Chapter 7. Backup and restore

7.1. Backups

To back up a Spectrum Scale file system, use the GHI ghi_backup(7) 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 described here.

Most sites create a crontab entry using either of the following forms of ghi_backup(7) to run a daily backup.

Example command:

```
% ghi_backup firefly image
```

See the Mitigating Snapshot Recall Requests section of the `GHI Management Guide` if planning to use snapshots with GHI either via **mmcrsnapshot** or **ghi_backup** with the --keep-snapshot option.

7.2. Restore

Refer to Backup and recovery for restore details.

Chapter 8. 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.

8.1. 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.

8.2. Converting from 3.3.0 to 4.1.0

8.2.1. 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(7). The default location is "/ghi".

On the HPSS core server:

1. Add new user to HPSS core with hpssuser tool. Refer to the *HPSS Admin 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 Admin 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

Chapter 9. GHI basics

9.1. Introduction

GHI is made to work with the IBM cluster file system (IBM Storage Scale). This software may be referred to as either Spectrum Scale or GPFS within this document.

The Spectrum Scale/HPSS Interface feature of HPSS (GHI) is software to connect Spectrum Scale and HPSS together under the Spectrum Scale Information Lifecycle Management (ILM) policy framework. This integration of Spectrum Scale with HPSS creates a hierarchical Spectrum Scale file system having virtually unlimited storage capacity and provides the option to use the hierarchical capabilities of Spectrum Scale and HPSS to provide disaster recovery protection for the Spectrum Scale file systems. GHI is an optional feature of HPSS and is offered under the HPSS license agreement. GHI users are expected to acquire or have acquired Spectrum Scale under a separate Spectrum Scale license agreement.

Both Spectrum Scale and HPSS scalability and performance are designed to meet the needs of data-intensive applications such as engineering design, digital media, data mining, financial analysis, seismic data processing, and scientific research. Typically, users tend to have many files in a file system, and these may be any mixture of sizes from very small to very large. Both Spectrum Scale and HPSS are highly scalable and are capable of ingesting thousands of files per second at rates limited by the hardware — usually the storage hardware and the transfer media. GHI is a scalable extension of HPSS.

A primary goal of GHI is to offer an integrated Hierarchical Storage Management (HSM) and backup solution for Spectrum Scale. GHI uses and extends Spectrum Scale ILM capabilities, providing a cost-efficient integrated storage solution that is scalable to hundreds of petabytes and billions of files. GHI enables Spectrum Scale file data transfers between Spectrum Scale high-performance storage, usually high-speed disk and HPSS cost-efficient storage (high capacity disk and tape). This movement between Spectrum Scale and HPSS occurs automatically under control of Spectrum Scale ILM policy rules and the DMAPI (Data Management API) framework, thus providing a complete and scalable HSM and backup solution that exploits HPSS parallel file system capabilities.

Spectrum Scale and HPSS are both network-centered cluster solutions offering horizontal scalability by adding cluster components. The GHI feature extends this architecture. Thus, the archive is not a single processor as in conventional storage systems. GHI provides servers that can be distributed across a high-performance network to provide scalability and parallelism.

GHI uses the Parallel I/O (PIO) interface provided as part of the HPSS Client Application Program Interface (Client API) to support parallel access to storage devices for fast access to very large files stored in HPSS. The GHI I/O Manager (IOM) organizes and manages the data transfer. An IOM will spawn threads to accomplish the actual collection and transfer the data: one per stripe, based on the HPSS stripe width of the COS configuration.

GHI uses ISHTAR (Independent Standalone HPSS TAR) to aggregate many small files into one large

file before migrating them into HPSS. Temporary storage on a Spectrum Scale file system is used to build the index files for the aggregate. The temporary files are removed once the aggregate file is written to HPSS. ISHTAR uses a multi-threaded buffering scheme to write files directly into HPSS, thereby achieving a high rate of performance.

GHI is written in ANSI C. It uses Remote Procedure Calls (RPC), Kerberos or UNIX for server authentication, and Db2 as the basis for its portable, distributed architecture for maintaining Spectrum Scale backups. The Db2 Server for HPSS is used to store the backup tables for GHI. Several Db2 tables are configured in the GHI database for each GHI managed Spectrum Scale file system. These tables support GHI backups, GHI garbage collection, and GHI mapping information. The GHI Session nodes are configured as Db2 clients to access the backup tables during GHI backup and restore operations.

9.2. GHI configuration

This section defines the high-level steps necessary to configure, start, and verify the correct operation of a new GHI system, whether that system is created from scratch or created by upgrading from any previous version of GHI. To create or modify the GHI configuration, we recommend that the administrator first be familiar with the information described in GHI Installation and *HPSS Admin Guide*.

Before performing the procedures described below, be certain that the appropriate system preparation steps have been performed. See the *GHI Installation Guide*. For a system created from scratch, be certain that the GHI installation and infrastructure configuration have been completed. To upgrade from a previous system, see the release notes for the upgraded version for specifics on the upgrade procedures.

9.2.1. Starting GHI for the first time

The GHI system is ready to be configured once the GHI software is installed on the node and the GHI infrastructure components are configured. Make note of the following limitations:

- File UIDs and GIDs greater than 2097151 are not supported.
- The maximum path length of files being migrated to HPSS is 1024 characters.
- GHI has a byte limitation for the path length and multi-byte characters will use additional space within that limitation (1043 characters or 1043 bytes including the snapshot path).
- GHI does not support the use of the newline character (\n) in a path
- For an image backup, the maximum path length is 1044 characters.
- Individual files in an aggregate must be less than 68 GB.

To start GHI, you must first start all infrastructure components for HPSS and Spectrum Scale as follows:

1. Make sure HPSS, Spectrum Scale, and all GHI nodes are powered on and the networks connecting them are working.
2. On the HPSS Core server, as root, start Db2, the SSM, and the startup daemon with:

```
% /opt/hpss/bin/rc.hpss start
```

If there is a remote PVR in the HPSS cluster, start the startup daemon there also.

3. Log in to the HPSS graphic user interface (**hpssgui.pl**) or HPSS administrator tool (**hpssadm.pl**) and start the rest of the HPSS servers and movers.
4. On the Spectrum Scale Cluster Manager node, start GPFS on all the servers that GHI is installed on:

```
% /usr/lpp/mmfs/bin/mmstartup -N ghinode1,ghinode2,ghimode3, ···ghinodeN
```

5. On the Spectrum Scale cluster manager node, start the GHI cluster:

```
% /opt/ghi/bin/ghistartup -g
```

During the initial GHI configuration, Spectrum Scale callbacks are established to automatically start when Spectrum Scale starts, so it may already be running when this command is run.

9.2.2. Configuring GHI for a new system roadmap

The following steps summarize the configuration of a GHI file system and policies needed to get GHI running. This list assumes that GHI has been installed following the steps in *GHI Installation*.

1. Configure the GHI file system. The settings to be changed can be listed with ghilsfs(7) and changed with ghichfs(7). Each setting is defined in detail later in this document, but there are settings that need to be changed before the file system is used in production.

    1. First, the Classes of Service (COSs) that will be used must be set. There are two for the file system: backup and aggregate index. The backup COS is where the backup data is stored. The aggregate index COS stores the aggregate index files. Since those indexes should never be purged from HPSS disk, their COSs should be constructed to meet that criterion.
    2. Next, the minimum and maximum files per aggregate. These settings will depend on the cutoff size of aggregate versus nonaggregate files.
    3. The final settings are for the monitor files. The SD and IOM monitor files need to be enabled. The frequency should be set to 60 seconds to start and can be modified as needed.

    Once these settings are set, restart the ED, SD, and IOM for them to take effect using ghishutdown(7) and ghistartup(7). These settings are described in more detail later in this document.

2. There are at least three policies that need to be modified for GHI to work: **migrate.policy**, **backup_migration.policy**, and **threshold.policy**.

    The **migrate.policy** and **backup_migration.policy** will define how data is moved from Spectrum Scale to HPSS and will have the same rules in them. The **backup_migration.policy** will have an additional argument to release some restrictions on aggregates to create more complete backups. The fewer rules in these policies, the better performance the policy scan will have. The rules are written based on how the data is stored in the Spectrum Scale file system. In most cases, rules are written for each project stored in Spectrum Scale.

The **threshold.policy** defines how the automatic purge of the Spectrum Scale file system takes place. There are three main rules in this policy: the path where the files are purged from, the thresholds, and the file size. The path will be the file system mount point. There are two thresholds, high mark and low mark. The high mark is the percentage of space filled in the file system where purge starts and the low mark is the percentage of space filled in the file system where the purge stops. Finally, the file size is configured. The policy should be configured so that the largest files are purged first so that the most space can be freed with the least amount of data movement.

9.2.3. Stopping GHI

The following steps show how to stop GHI.

1. Disable any automated scripts that are using GHI. Normally this includes automated migration, backups, or purges, but it could include more, depending on the site.
2. Dismount the spectrum scale file system. you should always dismount the file system before stopping ghi. If you leave it mounted, then any dmapi event that is triggered will be aborted by ghi.

   ```
   % /usr/lpp/mmfs/bin/mmumount <file system> -a
   ```

3. Shut down GHI. Spectrum Scale will still be running at this point. If you want to stop Spectrum Scale and GHI at the same time, you can use the **-G** instead of the **-g**.

   ```
   % /opt/ghi/bin/ghishutdown -g
   ```

Chapter 10. GHI concepts

10.1. Terminology

Spectrum Scale and HPSS have different meanings for the same term. GHI uses the HPSS terminology; here are the definitions of those terms:

Backup
Backup refers to backing up a Spectrum Scale file system into HPSS. The information needed to restore a Spectrum Scale file system including the restoration of the Spectrum Scale cluster file system configuration will be backed up into HPSS as well.

Cluster
A loosely-coupled collection of independent system nodes organized into a network for sharing resources and communicating with each other.

Garbage Collection
A GHI and Db2 process for removing GHI files from HPSS that are no longer referenced by Spectrum Scale or a valid backup.

Migration
Migration refers to the movement of file data from Spectrum Scale to HPSS while maintaining the file data in Spectrum Scale. There are two scenarios where migrations are performed. The first scenario is when the Spectrum Scale policy engine is run to transfer file copies from Spectrum Scale to HPSS. The second scenario is during a backup, which is when the most recent version of all files that have not been copied during an HSM migration, triggered by a policy, are copied to HPSS. The Spectrum Scale term is "pre-migration".

Purge
Purge refers to freeing up data segments in the Spectrum Scale file to free up Spectrum Scale resources. A Spectrum Scale policy is used to trigger a threshold request. The data blocks for the selected files are freed, leaving a stub in Spectrum Scale. The Spectrum Scale term is "punching a hole".

Recall
Recall refers to the movement of file data from HPSS to Spectrum Scale using ghi_stage(7) or from an ILM policy. This is not a DMAPI event. Recall events can be synchronous or asynchronous. The Spectrum Scale term is "pre-stage".

Restore
Restore refers to the capability to restore either a Spectrum Scale file system or a Spectrum Scale cluster from a selected backup.

Stage	Stage refers to the movement of file data from HPSS to Spectrum Scale. This process is invoked by accessing a file that is not dual-resident, and the data only resides in HPSS. This is a synchronous event. This process generates a DMAPI I/O event to stage the file back. This is sometimes referred to as "stage on-demand".
Pin	Pin or pinning refers to flagging a file so it cannot be purged from the Spectrum Scale file system.
GHI User Data	The files written by the users in Spectrum Scale.
Full-Access File System	File systems will normally be full-access; a GHI backup may be taken and migrated files may be deleted from Spectrum Scale to cause GHI to do garbage collection within HPSS.
Read-only File System	A read-only file system is one that is created and associated with a full-access file system and populated by restoring a GHI backup of the full-access file system. Restored files on a read-only file system may be recalled or staged from HPSS, purged from GHI, but they may not be modified or deleted. New files may be created, modified, or deleted but they may not be migrated into HPSS. In short, file-related actions on a read-only file system will not result in any addition of data to HPSS or deletion of data from HPSS. Read-only file systems were implemented to allow the validation of GHI backups. They may also be used to retrieve files from a backup without affecting the full-access file system or the status of backups.

10.2. Hierarchical storage management

GHI supports the following ILM mechanisms to transfer data between Spectrum Scale and HPSS, as well as to manage available space in the Spectrum Scale file system:

- Data migration
- Data recall
- File system limits
- Garbage collection

GHI also uses the Spectrum Scale DMAPI interface to stage data back from HPSS on-demand when a user attempts to access (such as opening) a Spectrum Scale file.

10.2.1. Data migration - transferring data into HPSS

Files are transferred (that is, migrated) from Spectrum Scale into HPSS based on rules sent to the Spectrum Scale policy engine. A migration policy provides a set of rules to identify which Spectrum Scale files are candidates for transfer to HPSS. The policy engine can generate two lists of files to be migrated. One list contains the files to be aggregated together as they are transferred into HPSS. The other list contains non-aggregate files which are individually transferred to HPSS. After the

candidate lists are generated, the policy engine invokes the GHI script ghi_migrate(7).

ghi_migrate: One or more instances of the script is invoked by the policy engine to coordinate with the GHI Scheduler to migrate the files to HPSS. For aggregation, files are placed in groups of an "aggregate bulk size" so that each **ghi_migrate** receives a request for a single aggregate. For non-aggregates, a single **ghi_migrate** instance receives a list of files to be processed based on the same "aggregate bulk size", but in practice, the number of files is usually only a small fraction of "aggregate bulk size". The policy engine calls **ghi_migrate**; do not run this manually.

GHI provides the following migration template in the `/var/hpss/ghi/policy` directory as an example of how to generate a list of files to be migrated:

migrate.policy: ILM rules are used to split files into two migration categories: aggregates and non-aggregates. Non-aggregates are larger files that are migrated into HPSS as a single file. Aggregates are smaller files that are combined with other small files into a single large file and then migrated into HPSS. ISHTAR is used to combine the files into larger aggregate files and migrate them into HPSS. When ISHTAR migrates files into HPSS, it creates two files in HPSS: the aggregate file and the index file. The aggregate file contains the data of all the files combined into it. The index file is the list of all the files in the corresponding aggregate and where they are in the aggregate. The location is called the ordinal. The index file is not required for migration or recall of the files from the aggregate, but it speeds things up when it is available.

This file should be modified by the GHI administrator to migrate the data from Spectrum Scale to HPSS in a tape-smart way. Here is an example of a tape-smart policy for aggregates:

```
RULE EXTERNAL POOL 'hsm_aggr' EXEC '/opt/ghi/bin/ghi_migrate OPTS' '-f ##'

RULE 'toHsm_aggr' MIGRATE FROM POOL 'system'
SIZE (FILE_SIZE)
WEIGHT (DIRECTORY_HASH)
TO POOL 'hsm_aggr' SHOW ('-s' FILE_SIZE)
WHERE FILE_SIZE > ## AND PATH_NAME LIKE '%<project_path>%'
```

What this policy does is tell Spectrum Scale to sort the files selected by the migration policy by their directory hash instead of the default inode. Then, when ISHTAR migrates the files as aggregates, files under the same pathname are grouped together. This helps when recalling data from HPSS tapes by minimizing the number of tape mounts needed to stage the data back when files within a directory are being staged up together. Contact HPSS support for suggestions on how to configure this policy.

Any attempt to migrate files from a GHI read-only file system will be rejected. The rejection code is -1 (Operation not permitted).

10.2.2. Data integrity during migration

GHI supports file-level checksumming for files that are migrated individually to HPSS (non-aggregate files).

The simplest approach is to enable checksumming for all eligible files by enabling the **Checksum**

on Migrate option on the GHI file system. This checksum will be generated by GHI when the file is migrated to HPSS. HPSS can then validate the file data when the data is migrated from disk to tape.

The best data integrity is achieved by having the user or application using GPFS calculate its own MD5 checksums and set the extended attributes described below prior to migration. This allows the user checksum to be validated by GHI, and then further validated by HPSS when the data is migrated from disk to tape. This additionally protects against data corruption that could occur between when the user wrote the data and when GHI processed it, but requires client-side integration through the extended attributes.

The file hash attributes can be set in one of two ways:

1. GHI provides a tool, ghi_filehash(7), to set the file checksum in **GPFS**, which GHI then has access to for validation.
2. Admins or users can set the extended attributes themselves on the files in GPFS. The two attributes that must be set are:
   a. **user.ghi.hashtype** set to "md5"
   b. **user.ghi.hash** set to the MD5 checksum

The attributes can be seen by the user using standard tools for investigating extended attributes such as getfattr. An example of correctly set attributes is below:

```
user.ghi.hash="ed4e6832867d7fd384d8debeb98f9e2a"
user.ghi.hashtype="md5"
```

Checksums for files in HPSS can be viewed using ghi_ls(7). The checksums listed there are what has been provided to HPSS for validation. Note that this is not the same as the user-facing file hash stored in GPFS, which is used to validate GHI migrations into HPSS. Those can be listed using ghi_filehash(7).

10.2.3. Data recall - transferring data from HPSS

Files are transferred from HPSS to Spectrum Scale in two ways: recall or stage. When a non-aggregate is transferred from HPSS to Spectrum Scale, it is recalled as a single file, just like in migration. However, when a single file is part of an aggregate, ISHTAR will search the index to determine where the file is and transfer only that single file from the aggregate back to Spectrum Scale. The other files in the aggregate will remain in HPSS until they are requested.

Recall operations

Recalling files from HPSS occurs as a background or a scheduled task. Either Spectrum Scale policy rules or a list of files and directories can be defined to retrieve the file data in advance of a user request to access those files. Policy rules in Spectrum Scale identify a list of candidate files that are eligible for recalling from HPSS. Next, the Spectrum Scale policy engine invokes the GHI script ghi_recall(7). The script parses through the list and generates sets of requests based on files belonging to the same aggregate and files residing on the same tape. This will optimize the retrieval of the data.

Files that reside on the same tape will be recalled together to minimize the number of tape mounts. Files that reside in the same aggregate will be recalled together using a single ISHTAR request. There is a recall template file, **recall.policy**, in the `/var/hpss/ghi/policy` directory that provides an example of how to generate a list of files to be recalled.

The ghi_stage(7) command is the alternative to recalling files using a policy. ghi_stage(7) can take as input either file lists or a specific files and directories to recall.

Stage operations

Stage files back from HPSS synchronously when the files are accessed by a user or a program. When files reside in HPSS, regions are placed on the files. When a user accesses the file data, DMAPI events are generated. The stage operation for a Spectrum Scale file is performed differently depending on where the data resides. If a file resides in both Spectrum Scale and HPSS, a WRITE or TRUNCATE event is generated when the user updates the file. This does not cause the file to be staged because it still resides in both places. It does, however, cause GHI to clear out the DMAPI regions since the file contains new data and needs to be migrated again, and the original backup of the file in HPSS will become a candidate for garbage collection.

If a file only exists in HPSS, a READ event is generated when the user opens the file in Spectrum Scale. This causes the file to be staged from HPSS and become dual resident (that is, it resides in HPSS and Spectrum Scale). If the file is modified after the stage is complete, a WRITE or TRUNCATE event will be generated and processed as stated above. GHI will handle all these actions, and the user need not even be aware whether the file being read resides in Spectrum Scale or is archived in HPSS.

Staging can be turned on or off for each file system using ghichfs(7) and passing in the **--sod** parameter. An error number can also be configured for each file system when staging is turned off. This is configured using ghichfs(7) and passing in the **--dse** option.

10.2.4. Managing available space

There are high-water and low-water marks that can be configured to indicate if a Spectrum Scale file system is out of space or is low on space. When the system triggers a high (NO_SPACE) or low (LOW_SPACE) event, the Spectrum Scale policy engine generates a list of files to be purged from the file system. Candidates to be purged are based upon file age, file size, etc. The Spectrum Scale ILM policy allows the candidates to be weighted so you can specify which files to consider first. The list of files generated will be enough to free up resources until the low-water mark is reached. There is a threshold template, **threshold.policy**, in the `/var/hpss/ghi/policy` directory that needs to be customized to list files to be purged.

To configure the system to react to these events, add the threshold policy and update the Spectrum Scale configuration to enable threshold processing. When activated and one of the DMAPI events is triggered, the **tsmigrate** process will run **mmstartpolicy**, which starts **mmapplypolicy** on one of the nodes in the cluster and the threshold policy will be used to determine which files' content to purge from the file system. Purging the data from GPFS internally while retaining the namespace information and attributes required to stage the data back is also known as "punching a hole" in the file.

10.3. Mitigating Snapshot Recall Requests

When files in GPFS snapshots are deleted, a recall of the file may be triggered as part of the delete process. In large systems with snapshots, this can trigger a large number of unnecessary concurrent recall requests. This scenario is referred to as a "recall storm". In a system like GHI where the files are being recalled from tape, this can resort in inefficient use of tape hardware.

If the site is using GPFS snapshots with GHI (for instance, using **ghi_backup** --keep-snapshot, or using mmcrsnapshot) the admin should consider enabling "recall storm mitigation". This feature enables GPFS to handle delete requests with snapshots more efficiently when backed by software like GHI.

For sites which do not use GPFS snapshots in tandem with GHI, the feature offers no benefits.

For more information on managing this feature, consult the **ghi_mitigate_recall_storm** man page.

10.3.1. Garbage collection and file deletion

When files are deleted from Spectrum Scale, a DMAPI event is triggered and GHI processes the event accordingly. GHI garbage collection is the removal of unreferenced GHI files from HPSS. Unreferenced files are GHI files in HPSS that no longer exist in Spectrum Scale and are not referenced by a backup of the Spectrum Scale file system. Files are not referenced when:

- They have not been migrated to HPSS.
- They are not part of a valid backup (that is, a successful backup).

Spectrum Scale will notify GHI when a GHI-managed file is deleted or updated. GHI will place an entry for each notification into the Db2 garbage collection (GC) table. When the GHI backup manager is executed to delete a backup, entries will be pulled from the GC table and processed as follows:

- If the file is not part of a backup, the file will be deleted from HPSS.
- If the file is part of a backup that is invalidated due to a restore of an earlier backup, the file will be deleted from HPSS.
- If the file is part of another backup, GHI will retain the notification in the GC table until all referencing backups are deleted.

When a file is deleted from Spectrum Scale and the file is not part of a backup:

- For a non-aggregate, GHI will delete the file from HPSS.
- For an aggregate, GHI will mark the file as invalid in the aggregate. If all the files in the aggregate have been deleted, the aggregate file and the index file will be deleted.

The following diagrams demonstrate what happens to a file, still in Spectrum Scale namespace and HPSS namespace, that has been deleted after it was migrated to HPSS.

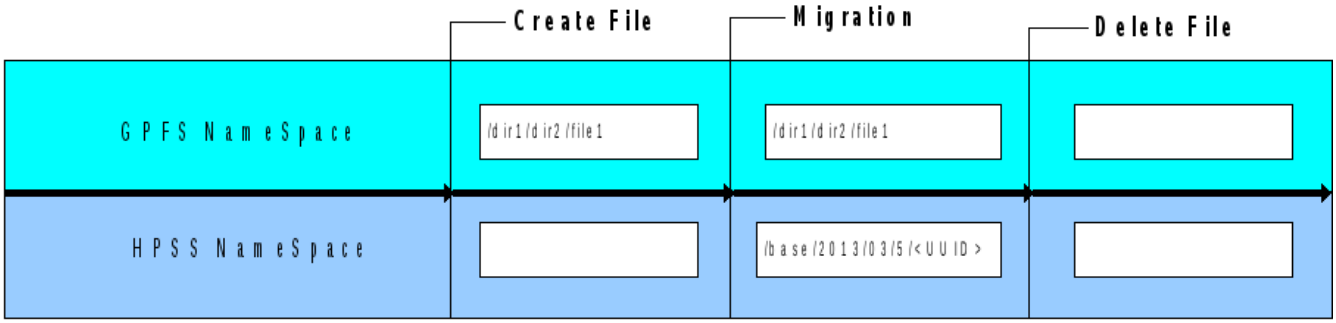

Figure 1. Namespace Mapping Table

When a file is deleted from Spectrum Scale and the file is part of a backup:

- An entry will be added to the GHI garbage collection table.
- The file will not be deleted from HPSS until all the backups associated with the file are deleted.

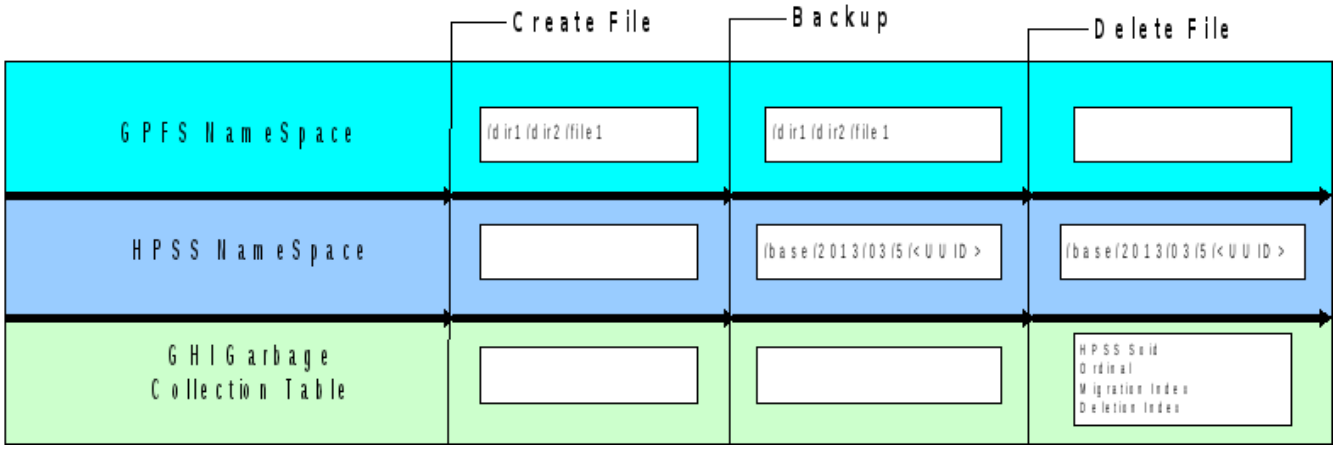

Figure 2. Namespace Mapping Table with Garbage Collection

When a file is added to the garbage collection, it is added as the SOID, ordinal, migration index, and deletion index. The SOID and ordinal identify which file in HPSS is to be deleted. The migration and delete index determine when the file is deleted. When a GHI backup is deleted, GHI compares the backup index of the backup being deleted. This is compared to the migration and delete index of the files in the garbage collection to determine if it is ready to be deleted. For example, there are five backups with indices 1, 2, 3, 4, and 5. A file was written after backup 1, and deleted before backup 5, so backups 2, 3, and 4 have this file. The file will not be deleted from HPSS until backups 2, 3, 4, and 5 are deleted. Backup 5 is in this list because it is the next pending backup. When the file was deleted, that backup could have been in progress when the delete happened, so it is included as a safety check. Once all these backups are deleted, the file is removed from HPSS.

10.3.2. Garbage Collection when HPSS or Db2 are down

Garbage collection is an operation that needs to occur quickly to respond to the DMAPI delete event. Failure to respond quickly can result in GPFS appearing to the end user to be stuck.

GHI periodically checks on the health of HPSS and Db2. If either is down, it will enter a "COMM FAIL" state. In this state, garbage collection events will be cached locally in GPFS, and replayed when connectivity is restored.

It can take several minutes for GHI to detect that Db2 is down. During this time, delete events will

pause. When they are timed out, they will return a failure from db2. At that point, GHI will enter the "COMM FAIL" state and cache any further delete events.

Note that if GHI goes down while Db2 is down, it will not be able to come back up. GHI requires Db2 to retrieve information related to the state of the current backup index, among other metadata.

10.3.3. Restoring deleted files prior to garbage collection

Use the `ghi_undelete_file` tool to recover files from HPSS that have been deleted from GPFS. The tool provides a safe means to recover the file and remove it from the garbage collection table. See the man page for details.

10.4. How Spectrum Scale files are stored in HPSS

The Spectrum Scale namespace is not mirrored in HPSS, meaning the file name is different in HPSS than it is in Spectrum Scale. All files associated with a Spectrum Scale file system are in their own directory in HPSS. Each file that is transferred into HPSS is given a unique file name based on the UUIDs. Aggregate files have an additional file name ending with `.idx`.

With the mapping tool, an administrator can locate or map a file using the Spectrum Scale file name, the HPSS file name, or the inode/igen that is assigned to the file in Spectrum Scale. Mapping provides more details on mapping.

10.4.1. Extended attributes

The Spectrum Scale extended attributes contain the following information used to map a Spectrum Scale file system object to an HPSS object:

HPSS Identifier

A unique identifier to locate where the Spectrum Scale file contents are archived in HPSS.

Hash

This is the bit file ID hash to the HPSS object. This was added to support HPSS 7.5.

Aggregate Flag

A flag to indicate whether the file is in an aggregate or not.

Ordinal

The index into the aggregate index file for the member. Applies to aggregates only.

Snapshot Identifier

This identifier associates the Spectrum Scale object with the backup in which the object was last backed up into HPSS.

Version Number

This is used to determine the format of the extended attributes. It is either 7.4 or 7.5. The 7.4 version indicates the SOID format matches the HPSS SOID prior to HPSS 7.5. The 7.5 version indicates the SOID matches the HPSS SOID format that began in 7.5.1. The format of the extended

attributes could vary between HPSS releases, so the version number allows you to query a file to see what version was used when the file was last backed up into HPSS.

The version number of an HPSS-managed file can be viewed with `ghi_ls -v`. The command `ghi_ls -x` can be used to force an update of the version number in cases where an old version is not automatically converted up to the latest during processing.

A separate DMAPI attribute (_GHI_PIN) contains a timestamp that indicates when the file was pinned.

10.4.2. Location of Spectrum Scale Files in HPSS

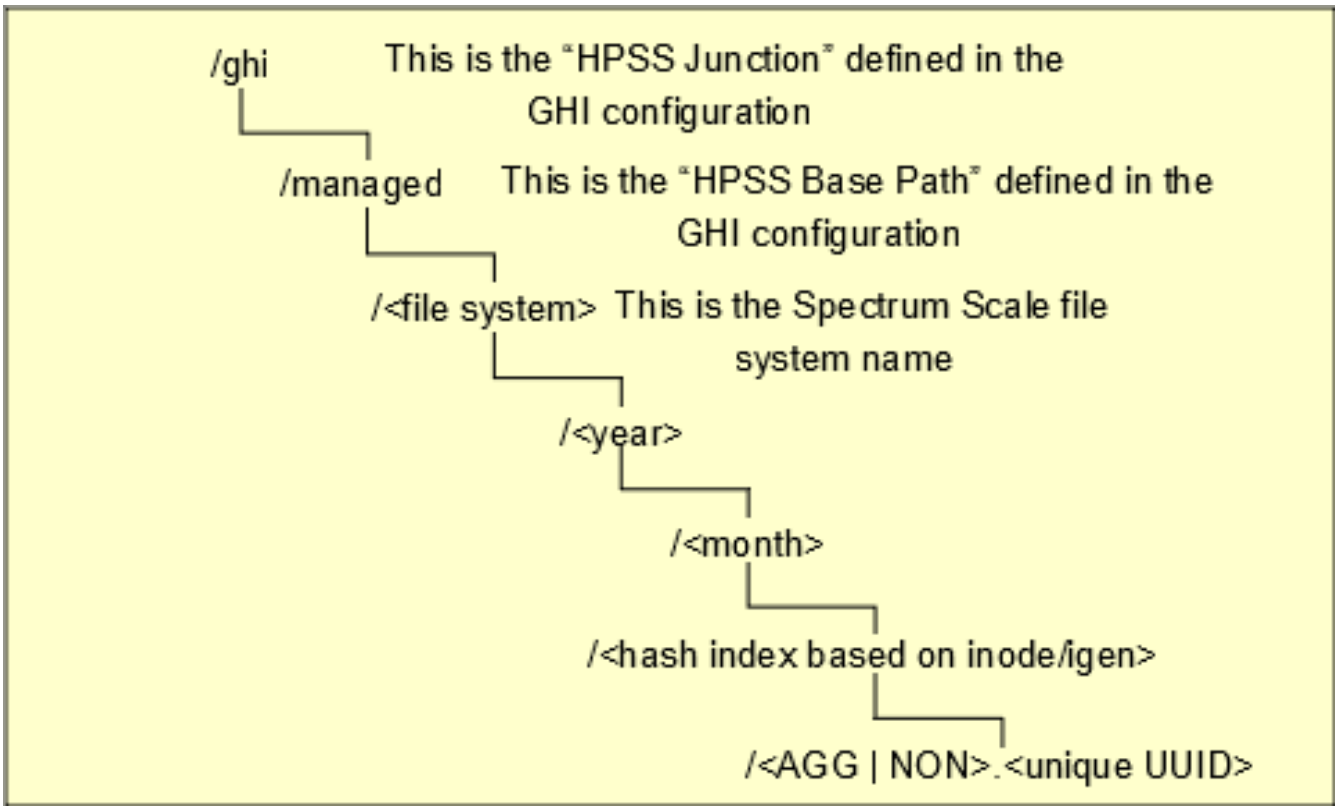

Figure 3. Location of Spectrum Scale files in HPSS

Hashing directories are created to store the Spectrum Scale files in HPSS. The directories are generated based on the following information:

- **/ghi** : Default root directory in HPSS for storing the Spectrum Scale files. This value can be reconfigured by modifying the GHI configuration with the **ghichfs** command. However, this value must not be changed unless the existing directory is empty. This directory should have Drwxrwxr-x, i.e 0775 permissions.
- **file system** : Spectrum Scale file system name.
- **year** : The year the file was migrated into HPSS (4 digits).
- **month** : The month the file was migrated into HPSS (2 digits).
- **hash directory** : For non-aggregate files, the inode and igen are used to determine the directory. For aggregate files, the file's UUID is used to determine the directory. The index and data file for

the aggregate are placed in the same directory.

- **unique UUID** : Unique ID for each file.

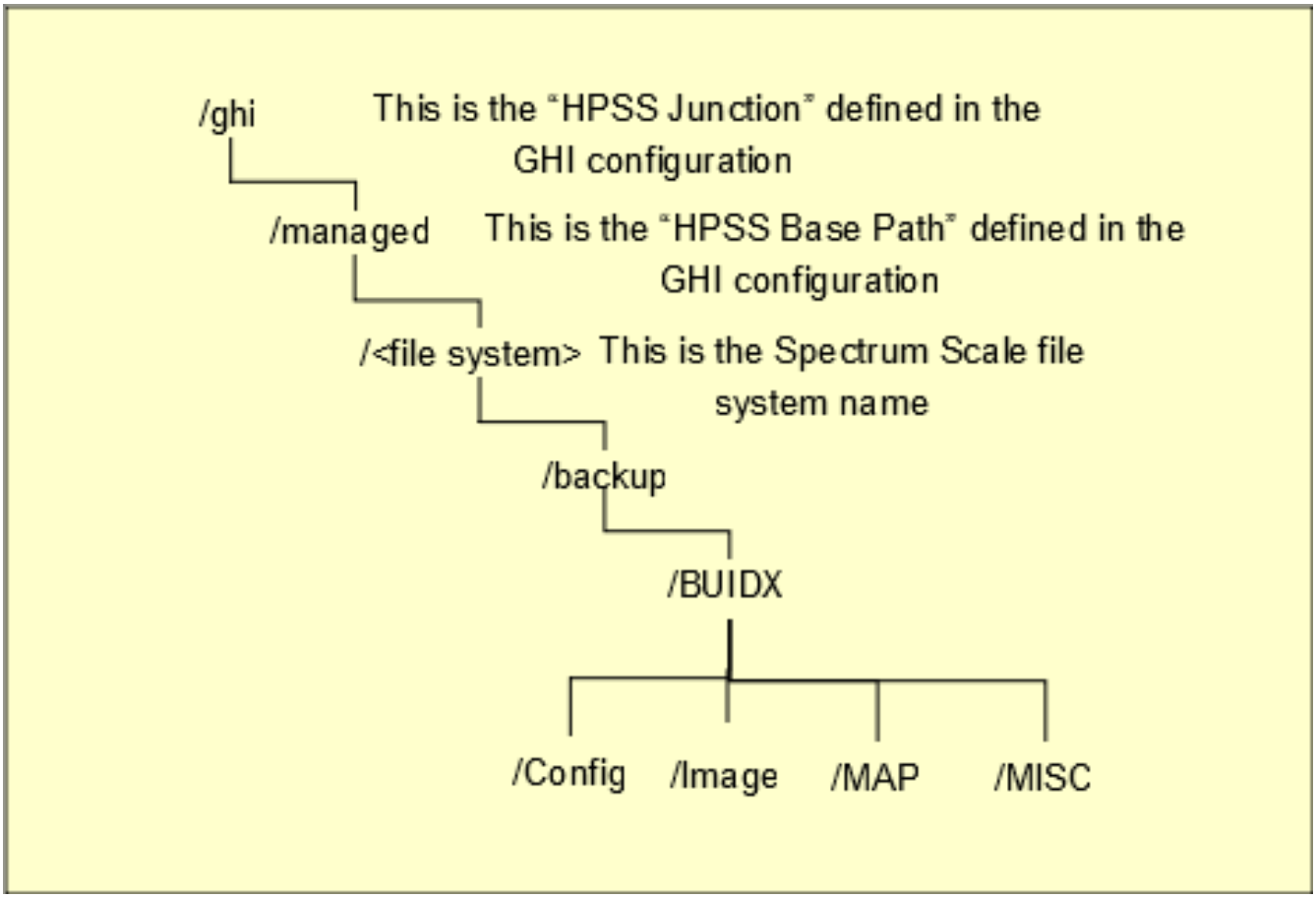

Figure 4. Location of image backup files in HPSS

Image backup files are stored in HPSS based on the backup index of the backup.

- **/ghi** : Default root directory in HPSS for storing the Spectrum Scale backup files. This value can be reconfigured by running **ghichfs**. However, this value must not be changed unless the existing directory is empty. This directory should have Drwxrwxr-x, i.e 0775 permissions.
- **File system** : Spectrum Scale file system name.
- **BU Index** : Db2 index associated with the backup.
- **Config, Image, MAP, and MISC directories** : Based on the type of files.
  - **Config** : Contains the Spectrum Scale cluster and file system configuration information.
  - **Image** : Contains the Spectrum Scale image backup files.
  - **MAP** : Contains the GHI mapping information for this backup.
  - **MISC** : contains the Spectrum Scale quota files and the version file.

10.4.3. HPSS characteristics

Classes of Service (COS)

Each file in HPSS is assigned a Class of Service (COS). The COS defines a set of parameters associated with operations and performance characteristics of a file. Refer to the *HPSS Admin Guide* for

information on COS. Each GHI file, which appears to HPSS as a user file, belongs to a COS which is selected when the file is created. The **Force Selection** flag can be set in the COS definition in the HPSS GUI to prevent automatic selection. If the flag is set, that COS must be specified in the policy to be able to store files to it.

There are three classes of GHI files written to HPSS (Data files, Indexes, Backups). The Default COS value used for each are set by the *ghichfs* command, while any overrides are set in the policy files.

- **Data files (aggregate and non-aggregate).** By default, a data file uses the Class of Service Maximum File Size Hints information passed to HPSS when the file is created. A policy can be defined to override the default COS by specifying a "OPTS -c <COS>" or "OPTS -c <COS:auto>" in the policy for non-aggregates and aggregates respectively. For example, "OPTS -c 5" means that GHI will ignore the default COS and use COS 5 instead when migrating files.
- **Aggregate index files.** By default, these files are written to HPSS using the "Aggregate Index COS" in the GHI configuration. All aggregate index files for a file system will go to the same COS. The site can override the default COS by specifying "OPTS -c <COS for data file>:<COS for index file>" or "OPTS -c auto:<COS for index file>". For example, "OPTS -c auto:5" means that GHI will migrate aggregate files using the default COS and the aggregate index files will be migrated using COS 5.
- **Backup Files.** These files are written to HPSS using the Backup COS in the GHI configuration. All backup files for a file system will go to the same COS.

The administrator can also specify a COS for individual rules in a policy run. This allows a site administrator to further configure policies to direct files to a specific Class of Service.

HPSS file families

HPSS files can be grouped into families. HPSS supports grouping files per file family for tape volumes only. All files in each family are stored on a set of tapes assigned to the family. When files are migrated from disk to tape, they are stored on a tape with other files in the same family. If no tape volume associated with the family is available, a newly imported blank tape is reassigned from the default family. The family affiliation is preserved when tapes are repacked.

Each GHI file belongs to a file family which is selected when either the file is created (tape-only COS) or when the file is migrated from disk to tape. File Families can be specified for each rule in a GHI policy. There are two types of GHI files that are associated with file families:

- **Data files (aggregate and non-aggregate).** By default, data files are not associated with a file family. However, a policy can be defined to specify the file family by adding an "OPTS -f<file family>:auto" or "OPTS -f <file family>" in the policy for aggregates and non-aggregates respectively.
- **Aggregate index files.** By default, these files are not associated with a file family. However, a policy can be defined to specify the file family by adding a "OPTS -f auto:<file family>" in the policy. The "auto" tells the system to not use a file family for the data file. However, the policy can be written as "OPTS -f <file family>:<file family>" to associate both the data and index files with the same or different file families.

There is currently no way to associate backup files with a file family.

HPSS storage subsystems

Storage subsystems can be used to separate HPSS resources. Spectrum Scale files can be placed in their own resources based on the HPSS Storage subsystem. GHI currently supports one subsystem per Spectrum Scale file system.

Refer to the *HPSS Admin Guide* for in-depth information on file families, classes of service, and storage subsystems.

Chapter 11. GHI internals

11.1. Servers and processes

A typical GHI system configuration consists of a single primary session node, one or more designated secondary session nodes for fail-over, multiple I/O manager nodes, and multiple client nodes which do not run any GHI software. The secondary session nodes can act as I/O manager nodes until they are told to take over as the primary session node.

The primary and secondary session nodes *must* be designated on the Spectrum Scale nodes as quorum nodes. Spectrum Scale selects the cluster configuration manager from one of the quorum nodes. GHI session node processes run on the quorum node designated as the cluster configuration manager. This section goes into detail on the responsibilities of the GHI servers and processes.

The figure below shows the basic architecture of a GHI system and the relationship to HPSS.

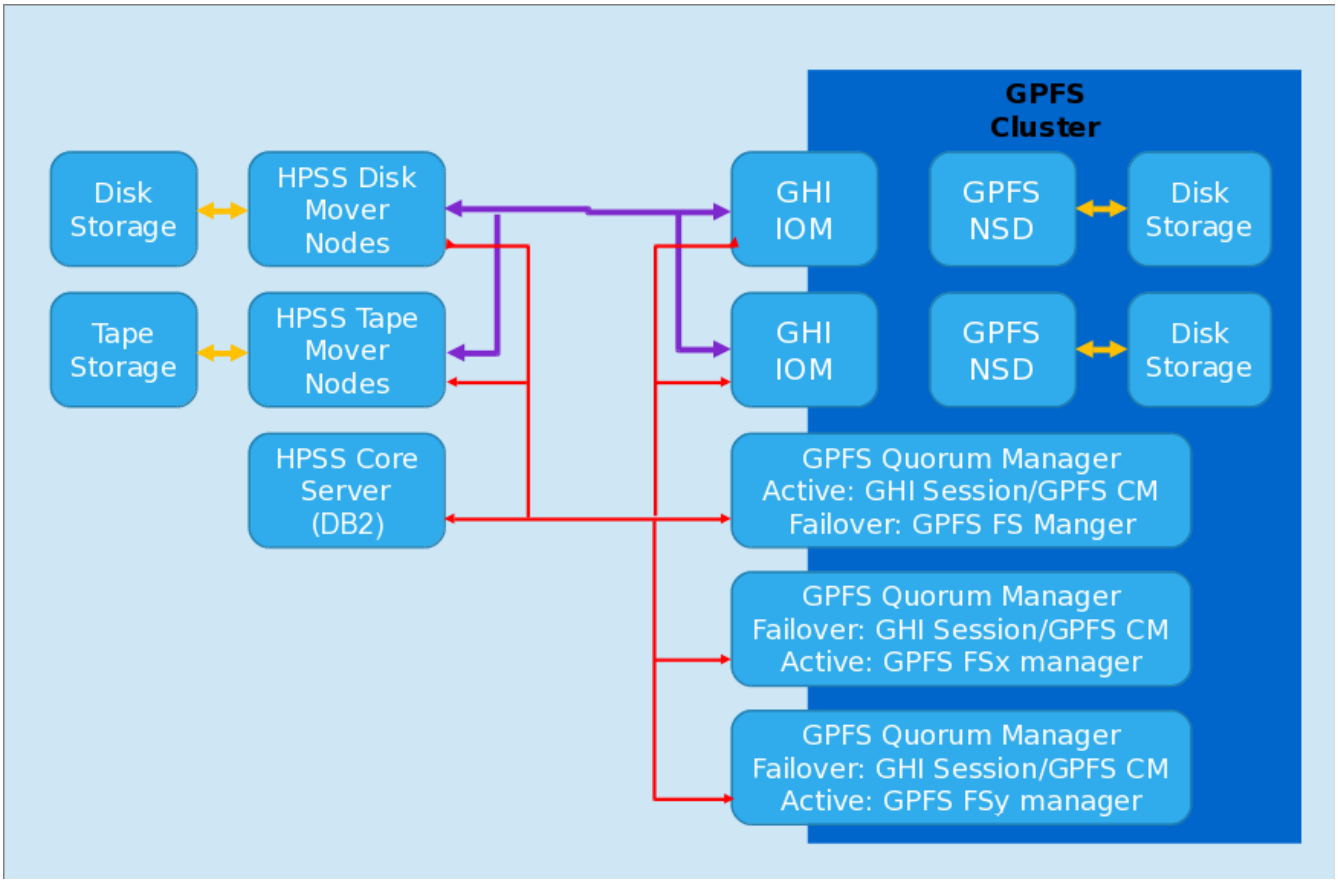

Figure 5. HPSS and GHI

Process Manager (PM)

The process manager is responsible for the execution of the other session node processes. The PM is started by Spectrum Scale as part of its heartbeat mechanism. It is started with the Spectrum Scale cluster configuration manager node start-up process by calling the GHI utility **hpssEventNotify**. When Spectrum Scale stops or loses quorum on the session node, the **hpssEventNotify** stops processes on the session node. In the event of a failover, it starts the processes on another configured quorum node. The GHI process manager runs on the GHI session node. It is responsible

for the following activities:

Starts and stops other GHI processes:

a. On startup, the process manager starts the mount daemon and the GHI configuration manager as child processes.
b. The process manager, at the request of the mount daemon, starts or stops an event daemon and a scheduler daemon when the file system is mounted or unmounted on the session node.
    - Ensures the mount daemon, configuration manager, and any event daemons and scheduler daemons are running and that they can perform work, and minimizes system hangs from occurring.
    - When one of the child processes dies, the process manager receives a SIGCHLD signal. This is an operating inter-process communication (IPC) signal. After receiving the signal, the process manager restarts that process.

Session node

A machine where a DMAPI session has been instantiated and the machine is registered to receive DMAPI events. The session node controls GHI activity. There can be only one session node active at a time. This node also functions as the Spectrum Scale cluster configuration manager (CM) node. If the CM moves, the session node will move with the CM. The session node processes are automatically started using Spectrum Scale callbacks when the CM starts. The following GHI processes run on the session node:

- GHI configuration manager (one instance per cluster).
- Event daemon (one instance per managed file system).
- Mount daemon (one instance per cluster).
- Process manager (one instance per cluster).
- Scheduler daemon (one instance per managed file system).
- I/O manager (if configured).

Configuration Manager (CM)

The CM is responsible for handling GHI configuration changes. GHI must be configured to run on every possible Spectrum Scale CM. GHI session node processes run on the CM.

Event Daemon (ED)

The event daemon runs on the session node and is responsible for capturing DMAPI events for GHI-managed files, including:

- Registers for DMAPI I/O (DESTROY, REMOVE, RENAME, READ, WRITE, and TRUNCATE) events.
- Receives read, write, and truncate events for files from the DMAPI session queue and submits the requests to the GHI scheduler daemon. If running on a read-only file system, WRITE and TRUNCATE events are aborted (instead of passing the request to the GHI scheduler daemon).
- Receives DESTROY events for files from the DMAPI session queue and performs garbage collection logic on the file.

- Receives responses from the GHI scheduler daemon and responds to the user request.
- On a read-only file system, receives RENAME and REMOVE events for files from the DMAPI session queue and determines whether the files are being managed by GHI (that is, they originated from a GHI backup of the associated full-access file system). If the files are managed by GHI, Spectrum Scale is instructed to abort the rename or remove operation. Thus, Spectrum Scale will not generate the usual subsequent DESTROY event for that file.

Mount Daemon (MD)

The mount daemon runs on the session node. It is responsible for the following activities:

- Captures MOUNT and UNMOUNT events for GHI-managed file systems and instructs the process manager to start or stop the associated event daemon and scheduler.
- Processes remote mounts for DMAPI-enabled file systems.

Scheduler Daemon (SD)

The scheduler daemon runs on the session node and starts when the file system mounts. It is responsible for distributing work to the IOMs, including:

- Accepts data transfer requests from the ED and ILM clients and schedules them in the queue based on their request category. There are six request categories: DMAPI, Backup, Migrate, Recall, Restore, and Admin.
  - 20% of the IOM threads are reserved for DMAPI requests.
  - The remaining threads are assigned requests by the SD based on a fair-share schedule sequence. The sequence is based on thread share values from the GHI configuration. Thread shares are defined for each request category. A sequence is generated from this where each request category will have its share of opportunities to run.

    For example:

    ```
    DMAPI share = 8
    Backup share = 5
    Migrate share = 3
    Recall share = 0
    Restore share = 2
    Admin share = 1
    ```

    The example above will generate the following scheduling sequence:

    ```
                              DMAPI
                              DMAPI
                              DMAPI
                       Backup DMAPI
                       Backup DMAPI
                Migrate Backup DMAPI
        Restore Migrate Backup DMAPI
    Admin Restore Migrate Backup DMAPI
    ```

First, three DMAPI are scheduled, then one backup, followed by one DMAPI, and so on. If a request is not available for the current category, the next categories are checked until a request is available to be processed. When the end of the sequence is reached, it is cycled back to start.

    - Thread shares can be changed dynamically in SD using **ghichfs**. The scheduling sequence is regenerated with the new shares and the sequence is run from the beginning.

- Communicates with the I/O managers to transfer data.
- Provides a mechanism to transfer results to the ED and ILM clients.
- Provides file system's full-access or read-only status.
- Provides load balancing of I/O to the IOMs.
- Detects an IOM failure and redistributes the work to other IOM machines.
- Processes purge requests for threshold processing.
- Filters out duplicate file requests. For example, a recall policy could be recalling files and someone then performs a ghi_stage(7) on one or more of these same files. The SD will filter out the duplicate requests.

I/O Manager (IOM)

A daemon that is responsible for data movement between Spectrum Scale and HPSS. The IOM starts via the **systemd** and remains in standby mode until the Spectrum Scale file system is mounted. The IOM can run on any Spectrum Scale node, although it is not recommended to run IOMs on the Network Shared Disk (NSD) node because the IOM will add an extra load that may decrease performance of the NSDs. The number of concurrent data transfers is limited to the configured thread pool size. The IOM is responsible for the following activities:

- Spawns ISHTAR to perform aggregate data transfers.
- Coordinates with HPSS to perform non-aggregate data transfers.

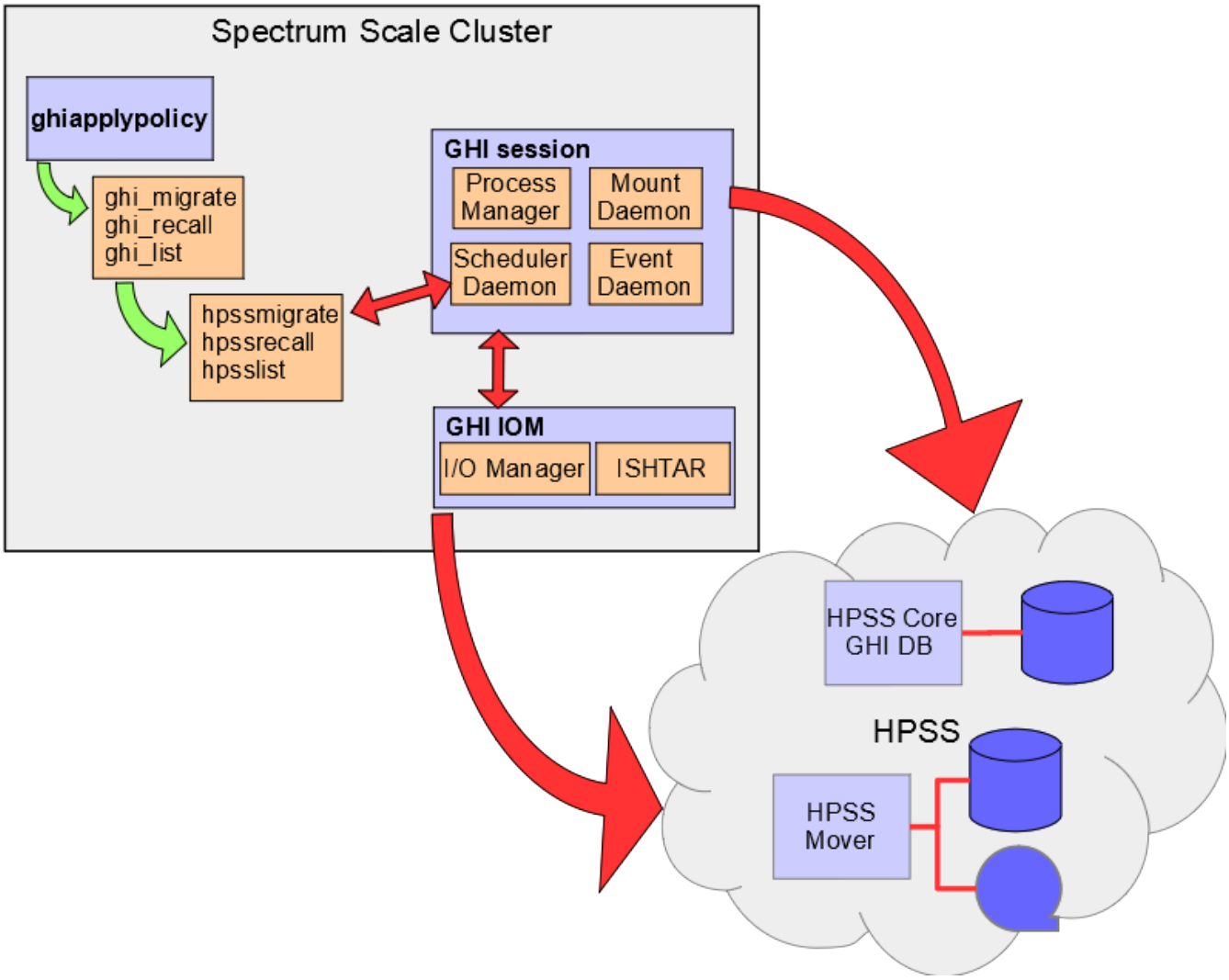

Figure 6. GHI session node

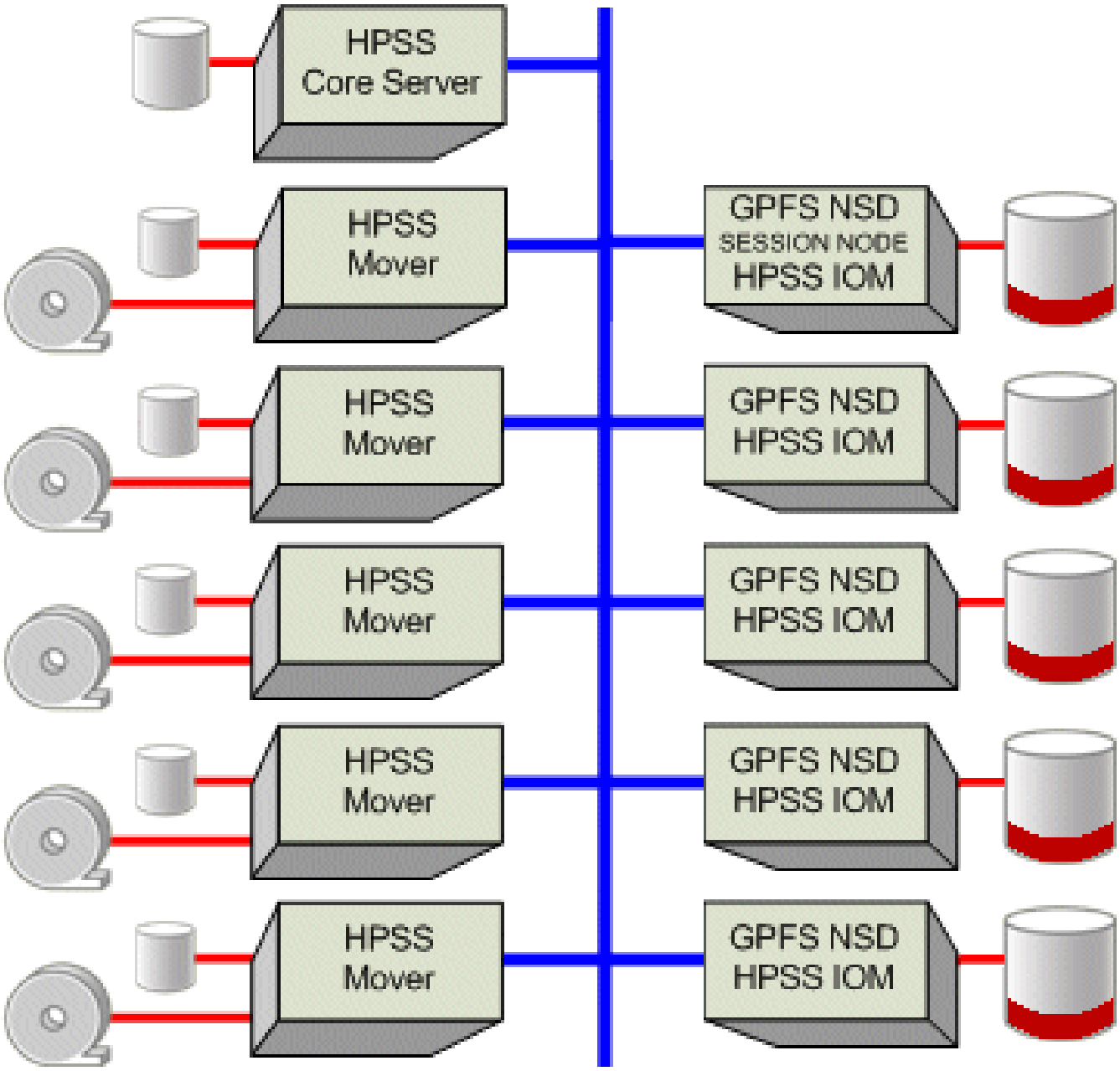

Figure 7. GHI I/O manager

ISHTAR

Software that aggregates files before migrating them into HPSS. ISHTAR clients are installed on each I/O manager node. The ISHTAR process is started via a script, `htar.ksh`, that resides in the `/var/hpss/hsi/bin/` directory. An ISHTAR process starts when an IOM requests processing of an aggregate. ISHTAR generates two files in HPSS: one file contains the data and the other file contains the index information. The GHI configuration setting "Max Aggregate Size" determines how many files are grouped into an aggregate. The Spectrum Scale ILM size option can also be used to construct aggregates. GHI-aggregated data relies upon ISHTAR index files to always be available to the system. If an index is inaccessible (missing, damaged, or delayed for an extended period), retrieving user file contents are impacted, including to the point of failure by the end-user to access their files. Storage of the index files must be constructed to protect this data from media failures or catastrophic damage. Index files should be considered equivalent to HPSS metadata and require the use of mirrored disk copies as well as multiple tape copies to properly protect the data. This includes using remote or off-site backups of this vital information as one would do for HPSS Db2 metadata.

11.2. Infrastructure

The GHI infrastructure components common among servers are discussed in this section.

Remote Procedure Call (RPC). Most GHI servers communicate requests and status (control information) via RPCs. GHI does not use RPCs to transfer user data. RPCs provide a communication interface resembling simple, local procedure calls.

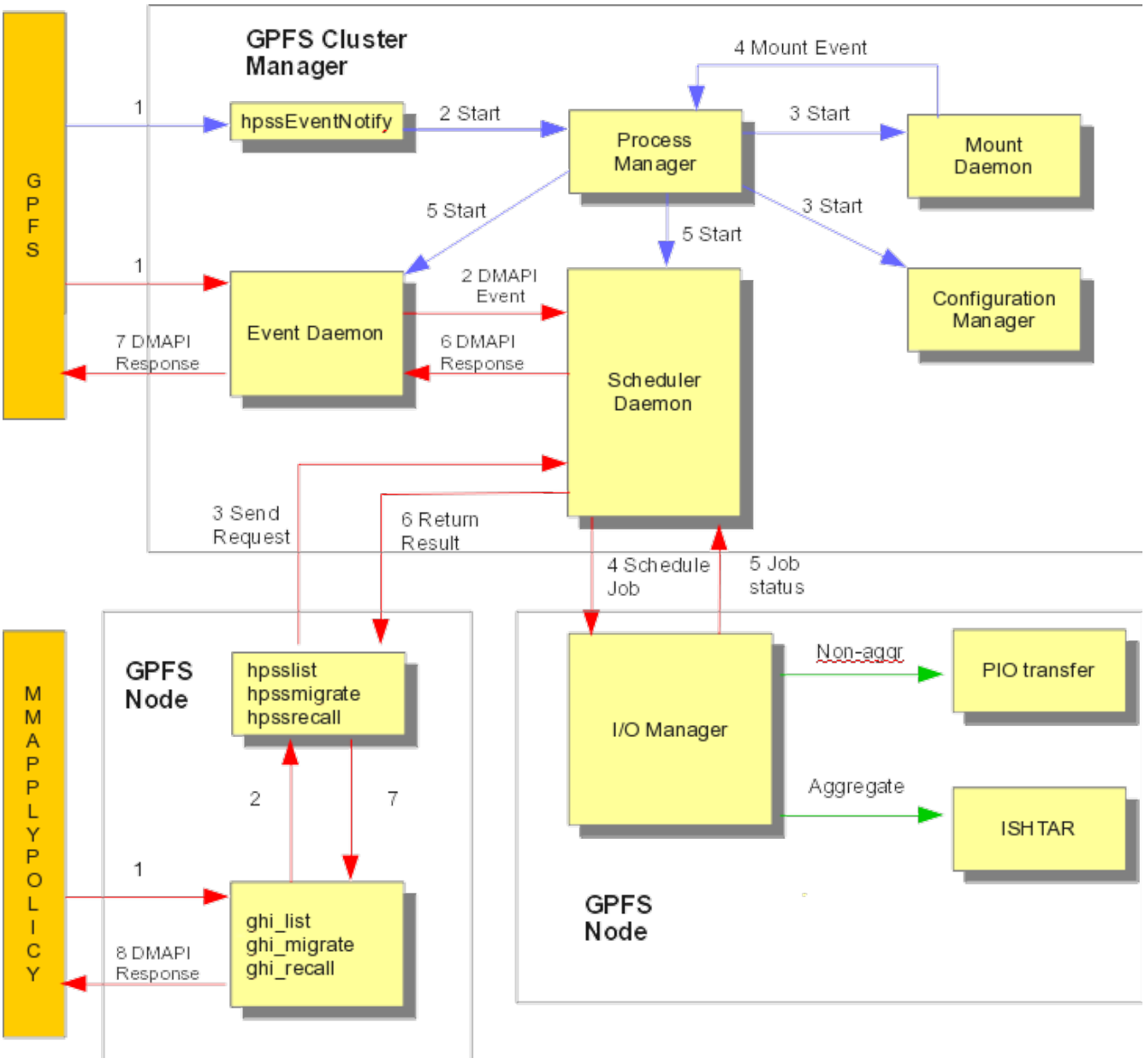

Figure 8. Intra-process communication

Threads

GHI uses a threads package for multitasking. The threads package enables GHI to run two or more processes simultaneously.

Security

GHI uses the HPSS security software to authenticate, to authorize access to HPSS objects, to enforce access control on HPSS objects, and to issue log records for security-related events.

- **Authentication**: Responsible for guaranteeing that the GHI cluster principal user, is the entity

that is claimed and that information received is from that entity.

- **Authorization**: Responsible for enabling an authenticated entity access to an allowed set of resources and objects. Authorization enables end-user access to HPSS directories and files.
- **Enforcement**: Responsible for guaranteeing that operations are restricted to the authorized set of operations.

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

Logging

GHI uses **rsyslog** for logging. Changing the GHI logging levels will alter which log messages are sent to **rsyslog**. The logging levels can be changed per component and per file system. See the **rsyslog** man page for more information.

User interfaces

GHI provides the user with a transfer interface, ghiapplypolicy(7). The interface is used to control the location where output files from the policy run are placed. It also controls the number of entries, that is, bulk rates, of each of the output files. **ghiapplypolicy** will use the GPFS policy engine to perform GHI operations on files which match the policy. **ghiapplypolicy** execution is controlled by the input policy and the options to the tool. See the man page for more details.

If a GHI operation like a migration needs to be stopped, the **ghiapplypolicy** process can be killed to force the operation to end early (in an error).

11.3. Network

Because of its distributed nature and high-performance requirements, a GHI system is highly dependent on the infrastructure network to communicate between GHI, HPSS, and Spectrum Scale servers. For control communications (that is, all communications except the actual transfer of data) among GHI servers, GHI requires TCP/IP services. Since control requests and replies are relatively small, a low-latency network is well suited to handle the control path.

The data path is logically separate from the control path but it may also be physically separate, although this is not required. For the data path, GHI supports the same TCP/IP networks as those supported for the control path. For large data transfers, the latency of the network may not impact the overall data throughput.

The HPSS Core Server must be able to connect to the network configured for Spectrum Scale.

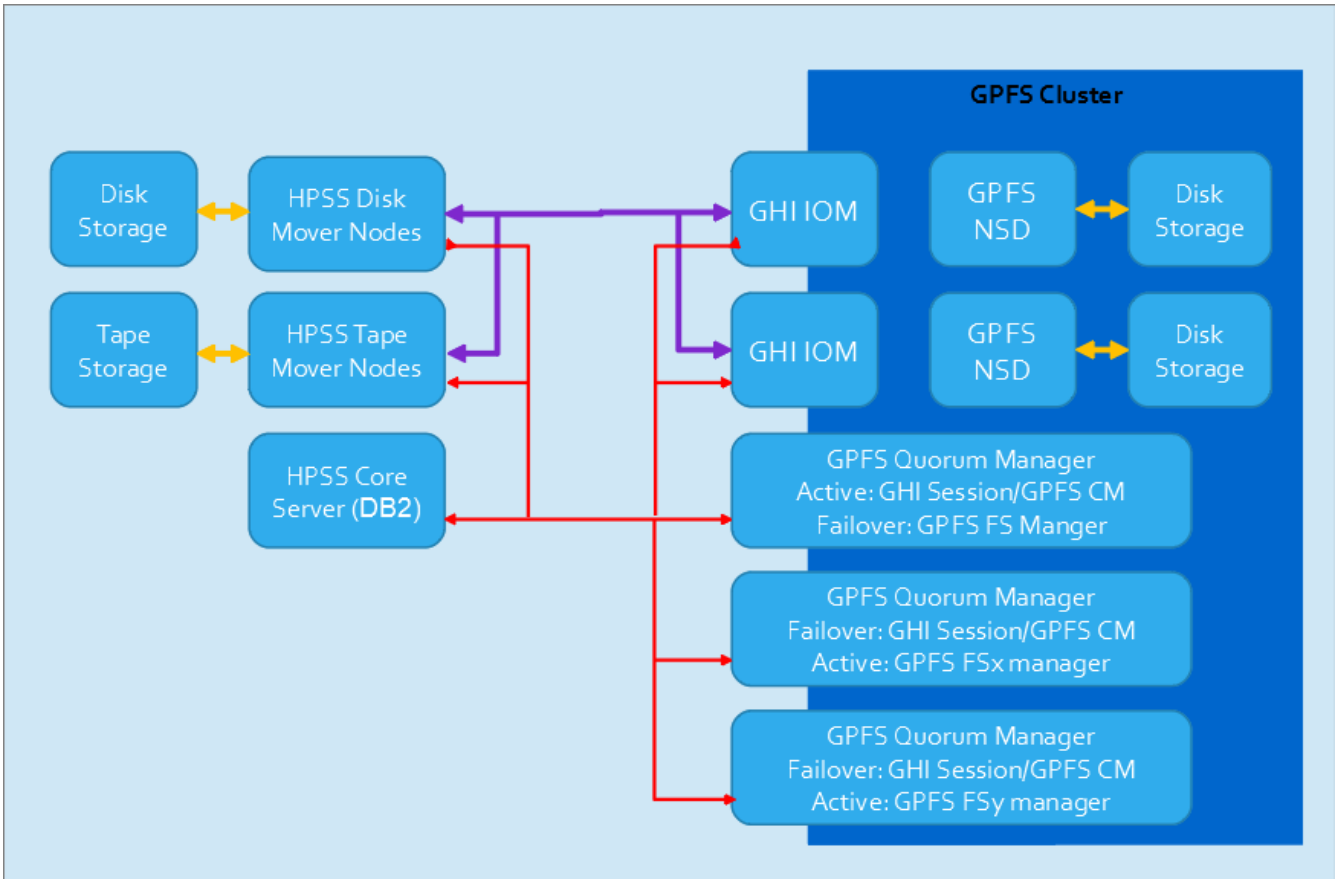

Figure 9. HPSS/Spectrum Scale network map

Notice how the blue network only connects to the Spectrum Scale nodes and does not connect to the HPSS nodes, which means GHI cannot migrate data to HPSS. There is a connection from the Spectrum Scale nodes to the HPSS nodes, the red network. This is because GHI communicates over the same network that Spectrum Scale is configured for. To communicate to HPSS, there must be a network route to the HPSS Core Server or move the HPSS Core Server to the blue network. The data movement now can take place over the configured data network, the red network, because it is set up in the HPSS configuration using the HPSS_API_HOSTNAME in the env.conf file.

GHI supports using IPV6 addresses. The HPSS_NET_FAMILY can be set to ipv4_only, ipv6, or ipv6_only. This value must be set in `/var/hpss/etc/env.conf`; otherwise, it will result in connection and data transfer errors.

11.4. ILM policy

GHI uses Spectrum Scale ILM policies to select files for migrations, purges, recalls, and backups. The administrator decides when to run each policy. GHI has a special command, ghiapplypolicy(7), that runs the policies on Spectrum Scale nodes that are configured for GHI. The **ghiapplypolicy** command is a wrapper for the Spectrum Scale **mmapplypolicy** command.

When Spectrum Scale processes a policy, the policy's rules are processed in the order in which they appear in the policy. Files which are rejected by a rule will not be considered under subsequent rules. Files which are selected by a rule are passed on for processing under the next rule. To avoid delays in processing the policy, the policy should be constructed so that rules that select the least number of files or reject the greatest number of files are listed first in the policy file. For example, if it is only desired to migrate un-migrated files within `/ghi/users/joe/` and `/ghi/users/john/`, the

policy could be written to first select all non-migrated files and then those in the two directories. Or, it could be written to first select only files in the two directories and then only those [from the two directories] which are not migrated. The second option will result in a faster policy run assuming these two directories contain only a small fraction of the total files on the file system.

The Spectrum Scale ILM migration policy provides the capability for GHI to copy (migrate) files from Spectrum Scale to HPSS. The migration policy identifies files that are new or recently modified that need to be copied to the HPSS repository. GHI processes the lists of files that have been identified and copies them from Spectrum Scale to HPSS. Larger files are usually copied straight to HPSS, while the smaller files are usually aggregated via ISHTAR into larger HPSS objects.

The Spectrum Scale ILM purge policy provides the capability for GHI to space-manage the Spectrum Scale file system. New and updated Spectrum Scale files are copied to HPSS on a periodic basis. When the Spectrum Scale file system reaches a pre-defined space threshold, the ILM purge policy is invoked to identify file candidates whose data can be removed from the file system. The ILM purge policy identifies the older, larger files as candidates. GHI "punches a hole" in the files that have been identified to free Spectrum Scale disk resources. The inode and metadata for these files are left in the Spectrum Scale file system, so from the user's point of view, nothing about these files has changed.

The Spectrum Scale ILM recall policy provides the capability for GHI to stage files in bulk from HPSS back to Spectrum Scale. The site administrator needs to create an ILM policy rule to stage a given set of files back from HPSS. These requests are optimized because files located on the same tape are recalled together to minimize tape mounts.

The Spectrum Scale ILM backup policy provides the capability for GHI to make a point-in-time backup of the Spectrum Scale file system. The ILM backup policy generates lists of files that need to be migrated, the file system namespace information, and a list of the Spectrum Scale files to be used to gather file attributes.

Administrators need to customize the policies to meet their needs. If a site has a large amount of disk file write activity, the administrator may want to have more free space and therefore run the purge policy frequently. However, if a site has a large amount of file read activity, the administrator may want to leave files on disk longer which means they would run the purge policy less frequently.

The result of a policy execution generates multiple exception and okay files. The exception files (with file extension `.exc`) contain all the files that failed during the policy run. The okay files (with file extension `.ok`) contain all the files that were successfully transferred. General options used by the policy files are:

-d

The "-d" option keeps both the exception files and the okay files from being deleted when the policy run is complete.

-D

The "dirty flag" option keeps the exception files, the okay files, and the generated policy files from being deleted.

If files are retained following the policy run, it is up to the administrator to clean those files out.

It is important to monitor the ILM policies to ensure they complete without errors. Some errors in the migration process can cause problems, such as the file system filling up or backups not completing.

Migrate policy – migrate.policy

The migrate policy defines how and which files are selected for migration into HPSS. GHI uses the directory `/<file system>/scratch/.ghi` to store temporary files during a migration, therefore this directory should be omitted from the migrate policy. Additionally, you should exclude files that are already in HPSS and Spectrum Scale (co-managed), or files with zero length. The migrate policy template file supplied with GHI includes these three rules by default.

The "Max Aggregate Files" sets the maximum number of files in an aggregate for migration. For example, if the Max Aggregate Files is set to 100 and 500 files are candidates to be migrated, then five files each containing a file list of 100 files will be generated from the policy engine. If all 500 files are selected for aggregation, then five 100-file ISHTAR aggregated objects will be created in HPSS. If some number of these 500 files end up being migrated into HPSS as individual (non-aggregated) objects, then fewer than five aggregates would be created in HPSS.

This figure shows what occurs when ghiapplypolicy(7) is called to start a migration.

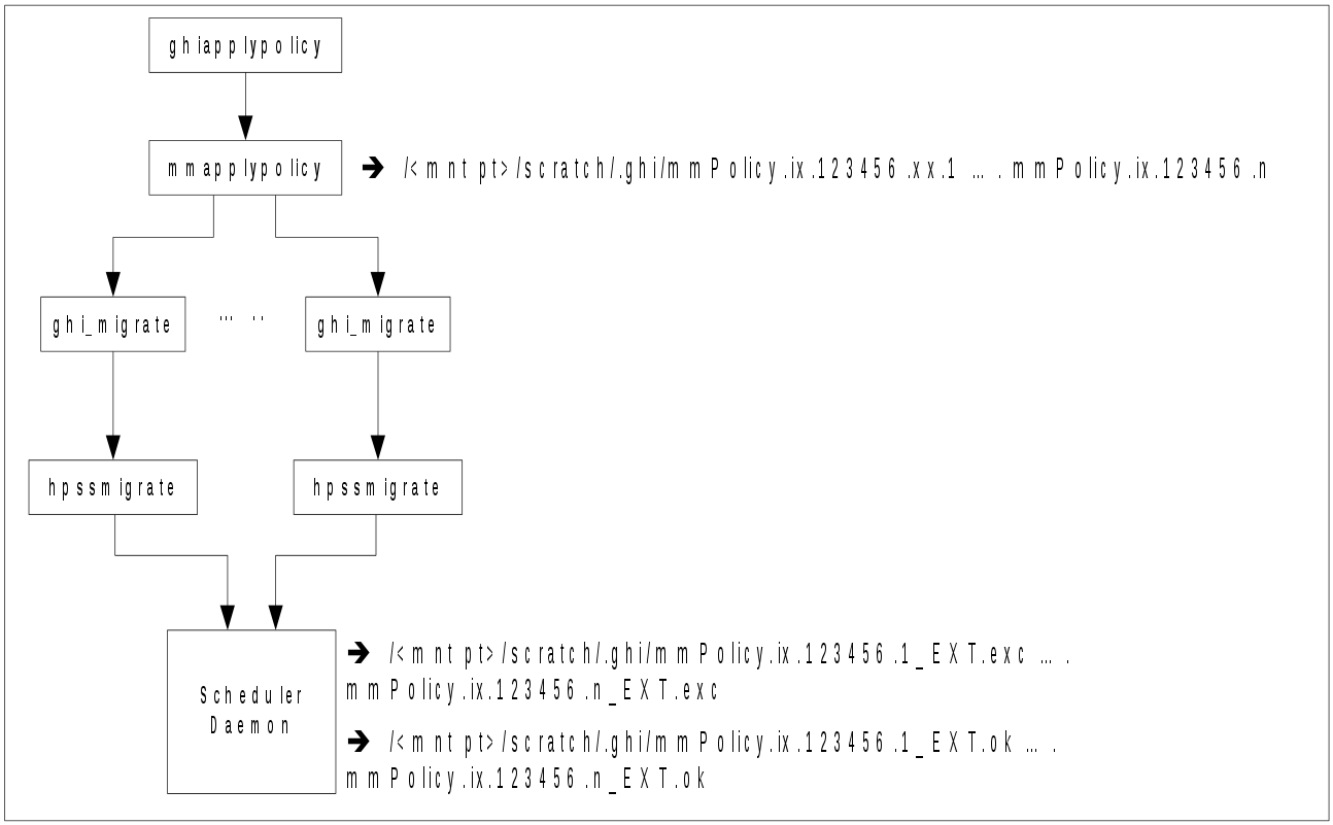

Figure 10. ghi_migrate

Recall policy – recall.policy

The recall policy defines which files are selected for recall from HPSS to Spectrum Scale. GHI uses the directory `/<file system>/scratch/.ghi` to store temporary files during a recall, therefore this directory should be omitted from the recall policy. Additionally, you should exclude files that are

not co-managed. The recall policy template supplied with GHI includes these two rules by default.

The policy generates one list for recalls. The list is parsed into aggregate and non-aggregate files. Aggregates are sorted based on where aggregates are in HPSS. The results of the sort are sent to the scheduler daemon on a per-aggregate basis, which allows GHI to make a single request to ISHTAR to retrieve all files from that aggregate. Non-aggregate files are sorted by the tape that contains the files and the file position on the tape.

Backup policies

There are two parts to the backup: the migration of data and the backup of namespace and attribute files. The migration of data uses the migration policy. Backing up the namespace and attribute files is performed via the Spectrum Scale **mmimgbackup** command. Any attempt to take a GHI backup of a Spectrum Scale read-only file system will fail.

Threshold policy – threshold.policy

The threshold policy is configured directly with Spectrum Scale and will start automatically to purge or punch holes in the Spectrum Scale file system to free space. There are four components in the policy that need to be configured. The first two are the high-water and low-water marks. When the Spectrum Scale file system is filled to the high-water mark, then the purge is started and will continue until the file system reaches the low-water mark and stops. The next component is the path in which the files will be purged. Finally, the file sizes that are going to be purged. Here is an example of a rule in the **threshold.policy**:

```
RULE 'toHsm1' MIGRATE FROM POOL 'system'
                  THRESHOLD(85,55)
                  WEIGHT(CURRENT_TIMESTAMP - ACCESS_TIME)
                  TO POOL `hsm_punch'
                  WHERE path_name LIKE '/mount/point/%'
                  WHERE FILE_SIZE > 262144
```

The high-water mark is 85% full, and the low is 55%. The path is defined by the path_name. Finally, any file greater than FILE_SIZE could become a candidate for purging. Any file smaller will not be purged.

Chapter 12. GHI configuration directories

System memory and disk space requirements for nodes where the GHI processes run depend on the configuration of the servers, the nodes that each server will run on, and the amount of concurrent access they are configured to handle. At least 8 GB of memory is required for the Spectrum Scale cluster nodes running GHI processes. When Spectrum Scale is running an ILM policy scan, it consumes a considerable amount of memory. Paging space should be sized with the same amount of space as the memory.

12.1. Data

When GHI is installed on the Spectrum Scale cluster, several directories are installed, each requiring approximately 15 MB of disk space. GHI uses space on the Spectrum Scale file system for temporary files, ILM policy output, and ISHTAR, in `/<mountpoint>/scratch/.ghi`.

/opt/hpss

Contains HPSS binaries, source code, include files, libraries, and utilities.

/opt/ghi

Contains GHI binaries. This directory should be a link to the actual directory where GHI is installed.

/var/hpss

The default location of HPSS and GHI configuration files and other files needed by the servers. It is required that this file system be at least 1 GB in size. Within `/var/hpss`, the following subdirectories are:

/var/hpss/cred

The default directory where credential files are placed. These files are typically very small.

/var/hpss/etc

The default directory where some additional HPSS configuration files are placed. These files are typically very small.

/var/hpss/ghi

The default directory where GHI files are located. There are five sub-directories: config, etc, log, policy, and tmp. The HPSS environment variable HPSS_GHI_ETC_PATH contains the base path to the GHI `/var/hpss/ghi/etc/` directory.

/var/hpss/ghi/tmp

The default directory where the process manager creates a lock file for each GHI process it starts. GHI may also write diagnostic log files and performance files here. The lock files are very small; however, the log files may get into the megabyte range, and the performance files can easily grow into gigabytes or even larger. Performance monitoring, which results in the creation of performance files, is normally disabled because of its large file size. Output from SIGHUP to dump RPC server state will go here in a <PID>.diag file.

/var/hpss/adm/core

The default directory where GHI servers put core files when GHI processes terminate abnormally. Core files may be large so it is required there be at least 2 GB reserved for this purpose on the session node and at least 1 GB on IOM nodes.

The administrator needs to remove unwanted core files to prevent `/var/hpss` from filling up.

/var/hpss/hpssdb

The location where the GHI database instance is stored. The database is used to access the Db2 server on the HPSS Core Server. The minimum file system size required is 20-30 MB for the runtime Db2 client, but should be larger to accommodate Db2 logs. Consult with HPSS support to size your GHI database infrastructure.

12.2. Metadata

The GHI backup, garbage collection, and mapping metadata is stored in the HGHI Db2 database of the HPSS Core Server.

The backup table has one row, 48 bytes in length, generated each time a GHI backup is performed. Rows in the table are added each time a backup is initiated and are never deleted. Rows are modified when a backup is deleted and when files are restored. The columns for the backup table contain the backup index, timestamp of the backup, snapshot ID, and the backup status. The backup index is unique for each backup and is how the administrator selects a backup. The snapshot ID is a unique ID created when the Spectrum Scale **mmcrsnapshot** command is run as part of a GHI backup. The status of a backup can be in one of the following eight states:

Pending	Next backup index to be used.
In progress	Backup is in progress.
Failed	The backup did not complete successfully and cannot be used for a restore.
Complete	Completed successfully and can be used if a restore is needed.
Deleted	Backup has been deleted.
Deleting	The backup is being deleted, or it was in the process of being deleted when it was interrupted and therefore needs to be restarted.
Invalidated	A backup is not valid. This happens if you restore a backup that is not the latest. For example, BU5 is invalidated if BU4 is restored.
Restore	A backup is being restored. Once the restore completes, the backup will become complete again.

The garbage collection table has one row generated for each deleted file in a backup. Rows are added each time a GHI-managed file is deleted from Spectrum Scale or altered. Rows are deleted when backups are deleted and the row meets the deletion criteria. When a row is deleted, the file it represents in HPSS is deleted. The columns in the garbage collection table contain the SOID (storage object identifier), ordinal, migration index, delete index, ordinal, state, and count. The SOID is how the file in HPSS is represented in Db2. The ordinal is where the file is in the aggregate. The

migration index is the index of the file that migrates.

The mapping table has one row, 2176 bytes in length, per file in the Spectrum Scale file system for each backup loaded into the Db2 table. For example, a file is written and three backups are taken, then the file is deleted, and three more backups are taken, now there are six backups loaded into the mapping table. There are three rows for this file, one for each backup it is contained in. Rows are added and deleted with the GHI mapping utilities.

Chapter 13. GHI features

GHI includes many features to help the administrator manage the system. A few of those features are highlighted in this chapter.

13.1. Mapping

The purpose of the GHI mapping is to create maps of the GHI backups that will tell the administrator which Spectrum Scale files point to which HPSS files and to which inode and igen.

The mapping can also be used to locate and read a file from a previous backup. For restoring individual files to GPFS, use the process described in Restoring deleted files prior to garbage collection.

One key feature of this tool is once the map is loaded and Db2 is accessible, the map can be used with or without access to GHI or HPSS. With this map, the user can locate the file using the Spectrum Scale file name, the HPSS file name, or the inode/igen assigned to the file in Spectrum Scale. GHI backups from GHI 2.4.0.1 and earlier versions cannot have their backups mapped. See the man pages for more information and examples on these tools.

Each new filesystem that is added to GHI will have a mapping table created automatically. Mapping tables can be created manually using the ghiCreateMapping(7) tool.

GHI automatically populates the mapping table when a backup is taken. This is done in the background, so there will be a gap between when a backup is taken and when its contents are ready to be used with the mapping tools.

Sometimes there may be a need to adjust the loaded mapping tables. Use ghiLoadMapping(7) to load selected backups. Either single backups or all backups can be specified. When all backups is specified, the tool will attempt to load all valid backups, skipping any which are already loaded.

Once all the backups are loaded, the administrator can search for files using the HPSS file name, Spectrum Scale file name, or Spectrum Scale inode number using ghiSearchMapping(7).

As a part of deleting a GHI backup with ghi_backup_manager(7), the mapping data for the deleted backup is automatically deleted from the mapping table. However, in some cases, it may be necessary to unload mapping table data manually. To do that, use ghiUnloadMapping(7).

The GHI mapping tables will take up space in the GHI database based upon the number of backups which are loaded at once, which should factor into planning metadata sizing for GHI. You can reduce the amount of space the mapping table uses by decreasing the number of valid GHI backups, or by unloading unneeded mapping data.

It can be useful to know what backups are currently loaded into the mapping table for validation or analysis. The currently loaded backups can be listed via ghiListMapping(7).

13.2. HTAR repack

This feature allows the administrator to identify sparse aggregates and combines them into new aggregates to free up space. This works by resetting the attributes of files that have already been migrated to HPSS, and re-migrating them as more complete aggregates during the next migration or backup. This means a migration will need to be run after the ghirepackfs(7) command is complete. After the **ghirepackfs** command finishes, the User Defined Attributes (UDA) will be updated on the old aggregate file in HPSS to be deleted when the backups that reference those files are deleted.

There are two ways to perform ghirepackfs(7). The first is based on a percentage of deleted files to active files in an aggregate to repack. When files in Spectrum Scale that are members of a GHI-HTAR aggregate are deleted and processed through the garbage collection table, the UDAs on the HPSS files are updated to reflect the number of deleted files in that aggregate:

```
scrub> udalist
//ghi/davis-1/2017/08/1306/AGG.d4b316cc-812a-11e7-b49b-00215ec467d4

Found 3 attributes:

                     KEY                                VALUE
 ------------------------------------------------------------------
             /hpss/ghi/Midx                              7

       /hpss/ghi/aggr/total                              5000

     /hpss/ghi/aggr/deleted                              187

scrub>
```

The administrator can provide this percentage to the ghirepackfs(7) command for GHI to repack these aggregates. To use this method, the administrator must first create a GHI *Mapping* table, load all the current backups into it, and unload all the deleted backups from it. The old aggregate files are then cleaned up when the backups that reference them are deleted. To run the tool, select a starting point ratio and test it to see how many files are selected. In this example, the ratio is 50%:

```
% /opt/ghi/bin/ghirepackfs <file system> -Tv -r 50
```

The second way is for the administrator to create a list of HPSS aggregate files to repack and pass that to the **ghirepackfs** command. This method does not require the mapping tables to be in place or up to date. Once the files are selected to be repacked by either method, GHI stages the data to be repacked back to Spectrum Scale and on the next migration or backup, the data is re-migrated into complete aggregates. The tool can also repack specific aggregates by using the force option:

```
% /opt/ghi/bin/ghirepackfs <file system> -v -f /tmp/filelist
```

Where `/tmp/filelist` is a list of the ISHTAR files names. This tool does not accept Spectrum Scale file names. The GHI backups do not have to be loaded into the mapping table for the **-f** option to work.

IBM recommends you use this utility on a quarterly basis, after a month's worth of GHI backups have been deleted, or if there is a directory that has several aggregates.

13.3. File system verification

There is a tool to check the file system for three types of inconsistencies: Spectrum Scale metadata, orphan files, and UDA mismatch files. Spectrum Scale metadata inconsistencies are Spectrum Scale files that do not have a corresponding HPSS file. Orphan files are HPSS files that do not have a corresponding Spectrum Scale file, do not have an entry in the GC table and are not referenced in a valid backup. UDA mismatch checks the aggregate file's metadata for incorrect counts of destroy/delete events. Before the administrator can run the orphan or UDA mismatch arguments, the administrator must first create a GHI mapping table, load all the current backups into it, and unload all the deleted backups from it. The Spectrum Scale metadata inconsistencies argument does not require the mapping table.

To check for Spectrum Scale files with incorrect metadata, run this command after a completed migration:

```
% /opt/ghi/bin/ghiverifyfs -f
```

To check for orphan files in HPSS, first create a GHI mapping table, load all current valid (that is, successful) backups into it, unload all the deleted backups.

Prior to running ghi_verify.py(7) for the first time, the HPSS subsystem database must be catalogued. See the *GHI Installation* for information on cataloging the subsystem database. abc

Run:

```
% /opt/ghi/bin/ghi_verify.py -d1 <GHI database name> -d2 <HPSS subsystem database name> -f
<Spectrum Scale file system name> -o <output filename>
```

To check for UDA mismatch, first create a GHI mapping table, load all current valid (that is, successful) backups into the map, unload all deleted backups, and run:

```
% /opt/ghi/bin/ghiverifyfs -u
```

This tool does not need to be run very often. The Spectrum Scale metadata inconsistency check runs as part of a GHI backup. Use the orphan file flag to check the system about once a year or after a suspected discrepancy between Spectrum Scale and HPSS. You will be asked by HPSS support to run the UDA verification if there is a problem; otherwise, it does not need to be run. Performance of each command will vary from system to system. The number of files in a file system also impacts the performance; the more files, the longer it takes.

```
% /opt/ghi/bin/ghiverifyaggr
```

This tool verifies the Spectrum Scale ordinal to HPSS aggregate mapping in a file system. It works through a multi-step process based on an initial policy run output file. The steps are listing the SOIDs for the input list, sorting the files based on aggregates, ordering the aggregates based on tape volume, comparing the contents of an index vs the aggregates, and finally fixing the DMAPI ordinal. The admin can either run the tool to identify mismatched aggregate metadata or fix the metadata

based on the aggregate index information.

This tool does not need to be run regularly. It should be run once a year or after a suspected discrepancy between Spectrum Scale and HPSS.

13.4. File system logging

GHI messages are sent to **rsyslog**, which is an open-source software utility that permits the logging of data in a central repository. The **rsyslog** utility needs to be configured to print DEBUG log level messages. There are 18 levels of logging available for each GHI process. Processes do not have to be restarted if the logging level is changed. Configure logging in `/etc/rsyslog.conf` to filter specific log messages. The following changes to `/etc/rsyslog.conf` are recommended:

- Change the system log rate limit interval to 0:
  `$SystemLogRateLimitInterval 0`
- Change the format of log messages:
  ```
  $template myFormat,"%timegenerated% %HOSTNAME% %syslogseverity-text%\9
  [%programname%]\9 %msg%\n

  $ActionFileDefaultTemplate myFormat
  ```
- Turn off repeated log messages:
  `$RepeatedMsgReduction on`
- Allow anything DEBUG or higher to be routed to `/var/log/messages`:
  `*.debug;mail.none;authpriv.none;cron.none /var/log/messages`

Starting in GHI 3.0, the log facility number can be configured. To configure the log facility number, follow the steps below:

1. Copy the file `/var/hpss/ghi/config/templates/syslog.conf.template` to `/var/hpss/ghi/etc/syslog.conf`.
2. Open the file and set the FACILITY_CODE to one of the following values and save the file.

```
# The following are valid facility codes
# LOG_USER - Messages generated by user processes. This is the default facility
# when none is specified.
# LOG_MAIL - Messages generated by the mail system
# LOG_DAEMON - Messages generated by system daemons.
# LOG_AUTH - Messages generated by the authorization system: login, su, and so
# on.
# LOG_AUTHPRIV - private security/authorization messages
# LOG_FTP - Messages generated by the ftp daemon
# LOG_LPR - Messages generated by the line printer spooling system.
# LOG_LOCAL0 - Reserved for local use.
# LOG_LOCAL1 - Reserved for local use.
# LOG_LOCAL2 - Reserved for local use.
# LOG_LOCAL3 - Reserved for local use.
```

```
# LOG_LOCAL4 - Reserved for local use.
# LOG_LOCAL5 - Reserved for local use.
# LOG_LOCAL6 - Reserved for local use.
# LOG_LOCAL7 - Reserved for local use.
# LOG_NEWS - Messages generated by USENET news subsystem
# LOG_UUCP - Messages generated by UUCP subsystem
```

If you need to redirect your log messages to somewhere other than `/var/log/messages`, modify the `/etc/rsyslog.conf` file to indicate where you want to redirect them to. If you change the `rsyslog.conf` file, restart the **rsyslog** service using the command:

```
% systemctl restart rsyslog
```

See the *GHI Install Guide*, "Configuring Logging with Multiple Filesystems", for information on configuring multiple filesystems with separate logging.

13.5. Stage on demand

The purpose of this tool is to allow administrators to turn off DMAPI stages to prevent users from excessively issuing stages that lock up HPSS tape resources. If a file is being staged by the IOM when stage-on-demand is enabled, the file will continue to stage uninterrupted. Future stages will fail, and return an error number (configured by the administrator), until stage-on-demand is disabled.

13.6. ISHTAR

ISHTAR is a tool that GHI uses to aggregate small files from Spectrum Scale to HPSS. Refer to Installing ISHTAR for information on how to install and configure ISHTAR.

13.7. Pinning

Pinning a file in GHI means putting an extended attribute on the file which can be used in policies to prevent the file from being purged from disk. The attribute is an integer named _GHI_PIN, and its value is the seconds since epoch value of when the attribute was last set.

The ghi_pin(7) tool is provided for administrators to pin or unpin GHI files.

It is important to note that running ghi_pin(7) by itself does not prevent the file from being purged. Rather, it facilitates this by setting an attribute that can be referenced in a purge policy. For example:

Example rule showing a purge where pinned files are ignored

```
RULE 'toHsm1' MIGRATE FROM POOL 'system'
              TO POOL 'hsm_punch'
                 WHERE XATTR_INTEGER('dmapi._GHI_PIN') IS NULL
                 AND path_name NOT LIKE '%/%scratch%'
```

You can also run a policy to generate a list of attributes either as seconds since epoch or as

timestamp values from the current time. See the policy templates `pin_time_list.policy` and `pin_duration_list.policy` in the templates directory for examples.

Chapter 14. Process failure and recovery

GHI provides a fault-tolerant system to keep the file system online and available. Spectrum Scale supports a means for GHI to provide exit scripts to be notified when there are changes in the quorum. This mechanism will allow GHI to either move the processes to another node or do what is needed to stay running on the existing node. There are currently eight events that can be captured for this purpose: init, ready, up, down, node failure, file system recovery, pre-unmount, and quorum loss. The events will invoke either a single "user" defined script, a High Available/NFS defined script, or both. The scripts will be invoked on those nodes that have the exit script installed.

14.1. Node failures

14.1.1. Session node

The node defined as the session node is selected by Spectrum Scale when the system is brought online. It is typically the node that is the Spectrum Scale cluster configuration manager node. GHI utilizes the Spectrum Scale heartbeat mechanism to monitor the nodes in the cluster that are potential session node candidates. During startup, Spectrum Scale executes the script, **hpssEventNotify**, to start all GHI processes and mount the file systems. Likewise, during failure, Spectrum Scale executes the script, **hpssEventNotify**, to unmount the file systems and stop all GHI processes. If the node fails and another node needs to take over, Spectrum Scale selects the new session node.

14.1.2. Manager node

The nodes running the I/O managers start the processes using **systemd**. The I/O managers, once started, remain idle until the file system is mounted on that node. If a file system is subsequently unmounted, its associated IOM will go back to being idle. If an I/O manager node fails, there are two scenarios that can follow:

1. The scheduler loses the connection to the I/O manager, cancels all requests to the failed I/O manager, and sends them to a new I/O manager that is active.
2. If a policy script was running on the node that failed, the policy manager will be notified that the request failed. In the case of a backup, the backup must be rerun. In the case of a migration, recall, or purge, no user action is needed because the process will resume from the time of the failure.

14.1.3. Client node

There is no special failover logic for a client node. If the node fails, the I/O manager detects a failure completing the data transfer. There is retry logic in the IOM to retry the data transfer, if the request is for a non-aggregate. For an ISHTAR request, the IOM does not spawn a new ISHTAR process if the return from ISHTAR indicates a failure.

14.2. Single process failures

14.2.1. ILM client

The processes *hpssmigrate*, *hpssrecall*, and *hpsslist* are started from the corresponding ghi_migrate(7), ghi_recall(7), and **ghi_list** policy scripts to perform the requested action. They are used to bridge the communication between the scripts and the GHI scheduler. If one of the GHI scripts detects that the HPSS process has terminated abnormally, the process will be restarted. The new process will start processing the policy file from the beginning.

14.2.2. Process daemon

In the case of an error or termination of Spectrum Scale on the GHI session node, the **hpssEventNotify** script will be executed. This script shuts down the GHI processes by notifying the process manager to shut the other processes down and then terminate itself.

14.2.3. Mount daemon

If the mount daemon abnormally terminates, the process manager automatically restarts it. There is no special recovery logic for this process. File systems cannot be mounted if the mount daemon fails. The mount or unmount request hangs and the user will need to kill and retry the mount or unmount request. Mount and unmount requests can only be handled when the mount daemon is registered to receive those DMAPI events. Refer to the *GHI Installation* and review the **mmlscallback** output.

14.2.4. Configuration manager daemon

Failure of the GHI configuration manager daemon will impact system operability by inhibiting the capability to make configuration changes and by the loss of periodic cross-cluster consistency checks. If the GHI configuration manager daemon abnormally terminates, the process manager automatically restarts it. There is no special recovery logic for the mount daemon.

14.2.5. Event daemon

The event daemon is started and monitored by the process manager via a request from the mount daemon when a file system is mounted, and terminated via a request from the mount daemon when a file system is unmounted. Failure of the event daemon will have a severe effect on the file system since the process is tightly coupled to file system user activity. For example, if the event daemon stops responding to synchronous events, the user processes that generated the events will block indefinitely.

If the event daemon abnormally terminates, the process manager automatically restarts it. Upon restart, the ED will assume the current session ID and check for outstanding DMAPI events. The ED will add the events to the internal queue and wait for responses from the scheduler. It will then do normal processing and wait for new DMAPI events.

14.2.6. Scheduler daemon

The scheduler daemon is started and monitored by the process manager via a request from the mount daemon when a file system is mounted, and terminated via a request from the mount daemon when a file system is unmounted. If the scheduler process abnormally terminates, the process manager will restart it. There is no special recovery logic for this process. The outstanding scheduled tasks will be lost, as well as the tasks being worked by the IOMs. All client requests that were being processed by the scheduler at the time it terminated will result in failures to the client.

14.2.7. I/O manager

I/O managers are started using the **systemd** on the IOM host systems. If the I/O manager abnormally terminates, it is automatically restarted by **systemd**. It will then wait for new requests from the scheduler. If the scheduler daemon detects that an I/O manager has abnormally terminated, it attempts to reassign its workload to other IOMs. There is no special recovery logic for this process.

14.3. Multiple process failures

14.3.1. ILM client and scheduler

If one or more ILM clients abnormally terminate and the scheduler terminates as well, the original request must be resent when the client and scheduler are restarted.

14.3.2. Scheduler and event daemon

If the scheduler and event daemon abnormally terminates, the process manager will automatically restart both processes. Upon restart, the event daemon sends any outstanding DMAPI requests to be processed. Those requests are sent to one or more I/O managers, and if the files have already been staged, the managed regions will be updated if needed and a successful response will be sent back to the application. Otherwise, the I/O manager will stage the file.

14.3.3. Schedule and I/O manager

If the scheduler and one or more I/O managers abnormally terminate, the process manager will restart the scheduler, and the I/O manager(s) will be restarted by **systemd**. All requests at the time of the failure must be retried by the application. The event daemon will resend all DMAPI requests which may cause duplicate stages.

14.4. HPSS unavailable

When HPSS is unavailable, most file system operations will continue to work. The operations that require data to be transferred between Spectrum Scale and HPSS will fail. The following operations will fail:

- User read events on co-managed files. Files where the data only reside in HPSS cannot be staged back to Spectrum Scale. When a user requests the file through a DMAPI event, an abort will be sent to the application.

- All policy manager executions.
  - Files that are recalled using the ILM interface will return an error to ghiapplypolicy(7). Files that are to be migrated/pre-migrated using the ILM interface will return an error to **ghiapplypolicy**.
  - Backups will fail with an error.

Chapter 15. Backup and recovery

15.1. Taking a GHI backup

A GHI backup will back up the Spectrum Scale namespace, file system configuration, including file sets, and file system metadata. There is one type of GHI backup currently supported: image backups. Image backups use the Spectrum Scale image backup functionality, officially known as Scale-Out Backup and Restore (SOBAR), via the **mmimgbackup** command. SOBAR is available with IBM Spectrum Scale Standard Edition or higher.

GHI backups work in two phases. The first phase is to run a backup migration. GHI needs to backup to be as complete as possible, so it starts with a migration. This migration is the same migration that is normally run, except there is an extra argument in the policy file, **-b**. This argument tells GHI to be as complete as possible, and if needed, ignore things like the minimum files per aggregate.

The second phase is the metadata migration phase. Image backup calls the **mmimgbackup** Spectrum Scale command, which instead of processing each file individually backs up the metadata as an "image" to an image file that is then backed up to HPSS.

GHI backups use the Spectrum Scale snapshot feature to take a point-in-time image of the file system. When running a backup, a snapshot of the Spectrum Scale namespace is saved after the backup migration policy and any other running migration policies complete, and the state of each of the files is saved. When migrating metadata, GHI uses the snapshot, instead of the active file system. If a file is modified after the snapshot has been taken, neither the updated file contents nor the metadata will appear in the snapshot or the resulting backup. If the modified file still exists at the next backup, any updates will appear in the snapshot and in the GHI backup.

To start a GHI backup, use the ghi_backup(7) command. The ghi_backup(7) calls the Spectrum Scale commands internally. For an image backup, the **mmimgbackup** command is called. The Spectrum Scale commands will call the ILM policy management engine. Each file system to be backed up uses its own copy of each of the following backup policy files that reside in the `/var/hpss/ghi/policy` directory. Customize the templates in the directory to suit your needs.

- **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 should select all the files to be backed up. This policy should match the **migrate.policy** with the addition of the **-b** argument.
- **backup_error.policy**. The backup error policy contains the rules that are used to validate the capture of the file system's metadata. This policy must not be modified by the administrator.

To run a ghi_backup(7), execute:

```
% /opt/ghi/bin/ghi_backup <file system> <type>
```

A second Spectrum Scale file system, of at least equivalent size to the original, is needed for temporary space to perform a production image restore. Test restores

may be done on smaller file systems as described in the read-only restore section.

Create a script or cron job to start the backups automatically. Run a GHI backup at a time when the Spectrum Scale system is the least busy to minimize impact to user operations. Work with HPSS support to determine how often GHI backups should be taken; most sites run a GHI backup at least once a day.

Once the GHI backup completes, it is critical to view the backup logs and output to ensure there are no errors. One of the important errors that you should look for is "Administrator action". Files with "Administrator action" are not included in the backup and must be investigated.

15.2. Restoring files from GHI

In case of the catastrophic loss of Spectrum Scale, GHI can restore the entire namespace and all data from its HPSS backups. For information on restoring a single file from HPSS, see Restoring deleted files prior to garbage collection.

The ghi_restore(7) utility displays the image backups stored in HPSS.

15.2.1. Read-only file system restore

Administrators can restore files from a backup to a read-only file system without needing to recall data for the entire file system. Files restored this way can be copied back to the full-access file system. This is commonly used for backup validation and for retrieving files which have not been garbage-collected.

The read-only file system does not need to match the name of the full access file system. The ghilsfs(7) settings **--junct**, **--basep**, and **--bupath** must be the same as the full access file system. The read-only file system will need its own IOMs to recall data from HPSS and for restore validation processing.

To perform a read-only restore of a Spectrum Scale file system, follow these steps:

1. Create a read-only file system to restore to. This file system should have sufficient space to restore the namespace metadata. Use configuration settings from the source file system to the extent possible.
   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
2. Create a read-only GHI file system with the file system you want to restore as the argument of the **-r** option.
   `ghiaddfs <read-only file system> -r <file system target> <···>`

   

   For image restores, the file system must have never been mounted.

3. Restore the file system to the read-only file system.
   `/opt/ghi/bin/ghi_restore <read-only file system> [-g temp GPFS file space]`

   

   Any attempt to take a GHI backup of a GHI read-only file system will be rejected. A

> *read-only restore of an image backup requires three Spectrum Scale file systems, the primary file system, the read-only file system, and the temporary file system. If the read-only file system is smaller than the primary file system, use the -t **ghi_restore** option.*

Every quarter, or, at a minimum, twice a year, test the backup by restoring to a read-only file system. This gives the administrator the ability to verify the backup was good.

15.2.2. Restoring to a full-access file system

To restore to a full-access file system:

1. Unmount the file system to be restored:
   `mmumount <file system> -a`
2. Note the configuration of the file system to be restored:
   `mmlsfs <file system>`
3. Delete the file system to be restored:
   `mmdelfs <file system>`
4. Re-create the file system to be restored, using the configuration information previously gathered:
   `mmcrfs <⋯>`
5. If this is to be an image restore, *make sure the file system is not mounted.*
6. Restore the file system with **ghi_restore**:
   `/opt/ghi/bin/ghi_restore [-g temp GPFS file space] <file system>`

Once the restore is successful, all backups newer than the restore point will be marked as invalid. All files associated with those backups will be marked for garbage collection when those backups are deleted.

Once the restore completes successfully, recall file data resident on the file system when the backup was taken. The administrator can run ghi_recall(7) or ghi_stage(7) commands to recall file data, or files will be recalled on-demand when users attempt to access a file. Batching up stage requests will result in a more efficient use of resources.

In order to restore, a link must exist at `/opt/hpss/lib/libhpssghi_restore.so` to `/opt/ghi/lib/libhpssghi_restore.so`. The RPM install of ghi-lib creates this link. If restores are failing, verify that this link exists; create it if necessary.

15.3. Managing GHI backups

Administrators can manage backups using the ghi_backup_manager(7) utility. This utility allows you to delete backups, resume deletion of a backup, and validate backup checksums.

The ghi_backup_manager(7) utility should only be run from a GHI session node. Note that **ghi_backup_manager** requires empowered access to HPSS for some

operations such as verifying migration attributes. Make sure that the appropriate users have been created on the node as described in the Users and Groups.

15.3.1. Deleting a backup

To delete a backup:

1. Run the ghi_backup_manager(7):
   `/opt/ghi/bin/ghi_backup_manager <file system>`
2. Enter the number 1 to select the menu option "Select a backup for deletion" and press enter.
3. A list of available backups is shown. Enter the selection number of the backup you want to delete and press enter.
4. A list of backups associated with the selection will be listed. Enter the selection number of the backup you want to delete and press enter.
5. Type "Delete" and press enter to start the delete.

15.3.2. Resuming deletion of a backup

To resume deletion of a backup:

1. Run the ghi_backup_manager(7):
   `/opt/ghi/bin/ghi_backup_manager <file system>`
2. Enter the number 2 to select menu option "Resume deletion of a backup" and press enter.
3. To resume the deletion of an individual backup enter the selection number of the backup you want to resume deletion on and press enter. If you want to resume the deletion of all backups where the deletion failed, type "All" and press enter.

15.3.3. Validating the checksums of a backup

To validate the checksums of a backup:

1. Run the ghi_backup_manager(7):
   `/opt/ghi/bin/ghi_backup_manager <file system>`
2. Enter the number 3 to select menu option "Validate backup checksums" and press enter.
3. A list of available backups is shown. Enter the selection number of the backup you want to validate the checksums on and press enter.
4. A list of backups associated with the selection will be listed. Enter the selection number of the backup you want to validate the checksums on and press enter.
5. Type "Validate" and press enter to start the validation process. If the validation is successful the following message will be shown:

```
************************************************
* Successfully validated the backup
************************************************
```

Chapter 16. GHI configuration settings

The GHI Configuration tools allow the administrator to configure GHI via a command-line interface. The GHI configuration manager daemon (the **ghi_cm** [CM] server process runs on the GHI session node) does an integrity check once a day of the configuration across the cluster. It checks for things like the file format, whether variables are defined, and so on. If there is a problem with a file, the CM will override it with the last working copy of that file. The CM also checks that the configuration files on all GHI nodes are the same as on the session node. If there is any difference, the CM will push the file(s) from the session node to the appropriate node(s). The update is quick — a few seconds to complete — and may also be executed on-demand by sending a SIGHUP to the CM process. This is an operating system signal that requires executing the **kill -SIGHUP** <*process id*> to the **ghi_cm** process.

The configuration files are:

`/var/hpss/ghi/etc/ghi.conf`

`/var/hpss/ghi/etc/ghinode.conf`

`/var/hpss/ghi/etc/ghi_<file system>.conf`

These files should never be edited unless directed to do so by HPSS support. GHI configuration can be broadly divided into three areas: cluster-wide, file system-specific, and IOM-specific. See Man Pages for details on each command.

Further, there is a GHI memcached configuration file. This should be configured when the GHI node is setup.

`/var/hpss/ghi/etc/memcached.conf`

Cluster-wide commands are:

ghicrcluster
: Create the GHI cluster the first time.

ghiupdate
: After a GHI version update, this pushes the configuration to all GHI nodes.

ghilsclusterconfig
: List GHI cluster configuration.

ghilsnodes
: List GHI nodes within the cluster.

ghiaddnode
: Add a GHI node within the cluster.

ghidelnode
: Delete a GHI node within the cluster.

ghilsfsdefaults

List the GHI file system configuration default settings.

ghichfsdefaults

Change the default configuration when you first add a file system.

ghichlog

Change the logging settings for the GHI cluster.

File system-specific commands are used to manage the event daemon and scheduler daemon, provide general configurations of the IOMs, and define how the file system's data are stored in HPSS. Configuration changes made with these commands generally do not adversely affect GHI beyond the file system to which they are applied. File system commands are:

ghilsfs

List the GHI file system configuration settings.

ghiaddfs

Add a Spectrum Scale file system to GHI.

ghichfs

Change the GHI file system configuration.

ghidelfs

Delete a file system from the GHI configuration.

And finally, the IOM-specific commands manage individual IOMs:

ghilsiom

List the IOMs configured for a GHI file system.

ghiaddiom

Add an IOM to a GHI file system.

ghichiom

Change the IOM configuration for a GHI file system.

ghideliom

Delete an IOM from a GHI file system.

All the *GHI Configuration Utility* commands may be executed from any Spectrum Scale node in the cluster. A command begins by retrieving the underlying configuration files from the current GHI session node and copying them to temporary files on the node on which the command is running. All processing is done to these temporary files. GHI configuration commands require both Spectrum Scale and GHI to be running. When one command is running, no other *GHI Configuration Utility* command can run. This is done by a locking mechanism built into the **ghi_cm**.

The **ghils*** commands do nothing more than pull data from the temporary files and print it for display. The other commands apply the requested updates to the temporary files and push the

updated files to all GHI nodes to replace their configuration files. As a command executes, it maintains a record of steps executed since startup. If an error should occur, it uses this list to undo all changes made prior to the error. The **ghils*** commands take the **-v** option for verbose output. This adds the output from the call to the **ghi_cm** server process on the GHI session node.

Commands that modify the GHI configuration — that is, commands other than **ghils*** — can be executed with the **-v** option to enable verbose output. Although this can result in a substantial amount of output, it should be used in case an error occurs and the automated error-recovery fails and must be corrected manually.

Each GHI configuration item consists of three parameters: key, description, and value.

Key — Is in the form *--xxx*, where *xxx* is a short alphanumeric string and identifies the configuration setting.

Description — A longer string, which may contain spaces and special characters. The description provides more information about the configuration setting.

Value — May include a comment. The format of value depends on what data is being conveyed and may be a string, number, Boolean, member of a list, etc. If value contains a "#" preceded by at least one space, then anything following the "#" is considered a comment, and anything preceding is the value.

Each **ghich*** command takes "<key_value>" parameters. A <key_value> can be specified via the key or description, which can be mixed when specifying multiple <key_value> parameters into a single command. The first way of supplying a <key_value> to a command uses the configuration item's key followed by the desired value or comment and results in two command-line parameters:

```
--Key "value [ # comment]"
```

The value need not be enclosed in quotes if it does not contain spaces or special characters, and a comment is not included. Note that single or double quotes are necessary with a comment.

The second way of supplying a <key_value> is through the configuration item's description and requires the entire <key_value> to be enclosed within single or double quotes, and results in a single command-line parameter:

```
"Description = value [ # comment]"
```

Since the description contains spaces, the "=" is required to separate it from the value and comment. Regardless of which method is used to specify a <key_value>, the "[" and "]" are not specified when including a comment, but are shown here merely to indicate that specification of a comment is optional.

To reduce typing when entering commands at the terminal, the key form is used to specify a <key_value>. The description form is used for better readability from within a script. For example, to change the network speed of an IOM because a new network was added to the node:

```
# ghichiom davis-1 davis.clearlake.ibm.com:8012 --etr "10GB # New Network added"
Distributing updated GHI FS config to all GHI nodes...
Done.
```

```
Key        Description               Value (# comment)
 ------------------------------------------------------------
IOM Node:Port             davis.clearlake.ibm.com:8012
--asn    Active Session Node       TRUE
--etr    Estimated Transfer Rate   10GB (# New Network added)
```

When a command that modifies the GHI configuration completes normally, it displays "Done.", as shown above. Otherwise, it will display an error and attempt to back out the changes. An example of an error message follows:

```
# ghichiom davis.clearlake.ibm.com:8012 --etr "10GB # New Network added"
There was a command execution error:

<IOM> must be specified as '<node>:<port>'
*** Command terminating with error
*** No changes have been made
```

If an error occurs prior to any permanent changes made, there is nothing to be backed out and "*** No changes have been made" is displayed. If an error occurs after permanent changes have been made anywhere in the cluster, the message "!!! ABORTING -- UNDOING CHANGES !!!" is displayed and it attempts to back-out the changes. You would see this message in the ghiupdate(7) command.

A few configuration items are not displayed with a key from the **ghils*** command, which signifies they cannot be modified or their entry doesn't follow as described above. For example, An IOM's node or port cannot be changed with ghichiom(7). The IOM must be deleted and re-added via ghideliom(7) and ghiaddiom(7).

Each of the **ghich*** commands keep a date-stamped history of successfully applied changes. There is currently not a *GHI Configuration Utility* command to extract this history. If this history is needed, it can be obtained from the files located in "$HPSS_GHI_PATH". History records appear under the configuration item to which they apply, most recent first and are formatted as follows:

```
# <date> = <old_value> <old_comment>
```

where:

<date>

When the change was made.

<old_value>

The value before the change.

<old_comment>

The comment before the change.

If a history of applied changes is not required, issue the **ghich*** command with the option **-H**.

A list of items that can be configured, grouped by their associated *GHI Configuration Utility* commands, are described in the subsections below.

16.1. GHI logging configuration

GHI logging has been enhanced to use **rsyslog**. The administrator can configure **rsyslog** so that all GHI logs across all GHI nodes point to a single central log file, making troubleshooting much simpler. The administrator can also configure the format of each log message to suit any log monitoring tools they have implemented. There are some recommended **rsyslog** settings that produce more useful logging:

- Change the system log rate limit interval to 0:

```
$SystemLogRateLimitInterval 0
```

- Change the format of log messages in syslog. Note that the `$template` line here has been split into two separate lines solely for the purpose of this documentation. Consider these two lines as parts of the same formating line, with a single space between the two parts.

```
$template myFormat, "%timegenerated% %HOSTNAME% %syslogseverity-text%\9
  [%programname%]\9 %msg%\n"

$ActionFileDefaultTemplate myFormat
```

- Turn off repeated messages:

```
$RepeatedMsgReduction on
```

- Allow anything DEBUG or higher to be routed to `/var/log/messages`:

```
*.debug;mail.none; authpriv.none;cron.none
```

Increasing the log level on a process will increase the number of messages recorded for that specific process. GHI processes do not have to be restarted after logging is changed. GHI has separated as many individual processes to give the administrator the most granular control over logging. The processes that can have their individual levels increased are:

-p Process manager

-m Mount daemon

-c Configuration manager

-t Tools

-s Scheduler (requires the file system name)

-e Event daemon (requires the file system name)

-q Policy (requires the file system name)

-i IOM (requires the file system name)

16.2. Logging levels

There are 18 levels of logging available for each GHI process and they can be expressed in any order. The default levels are ALERT, CRIT, ERR, INFO, and NOTICE.

ALERT Log events that require operator investigation into messages in `/var/log/messages`.

BACKUP These log events can be anything related to GHI backups. Depending on the message displayed, the operator may have action.

CONFIG Debug log events related to opening configuration files, locating configuration settings, and modifying configuration settings (for example, log level) within GHI.

CRIT Log events for critical errors such as "Failed to start Event Daemon". These errors signify the loss of multiple capabilities and GHI will require immediate operator intervention.

DB2 Debug log events related to Db2 operations performed by GHI, such as garbage collection updates or SQL queries performed.

DEBUG Log events and data used to debug error conditions. These log messages are written only to the process log. Normally meaningful only to HPSS support. Operator intervention is not required unless instructed by HPSS support.

ERR Log events for major errors such as "Failed to open file". These kinds of errors signify that some substantial GHI processing or capabilities are lost or malfunctioning. Operator intervention will be required.

GPFS Debug log events related to Spectrum Scale DMAPI operations performed by GHI, such as setting or retrieving file attributes.

HPSS Debug log events related to HPSS client interface calls performed by GHI, such as connecting to HPSS and reading or writing a file to HPSS.

INFO These are informational messages, such as "Sending stripe group info to clients". These are not error messages. Operator intervention should not be required.

NOTICE Log events for when a process starts or stops. These are not error messages. Operator interaction is not required.

POLICY This is information from a policy run. Operator interaction is not required.

QUEUE Debug log events related to adding work items to the SD queue and log events related to the IOM processing those work items on the queue within GHI.

RPC Debug log events related to the Remote Procedure Calls (RPC), such as the success or failure of an RPC connection or registration within GHI.

THREAD Debug log events related to adding work items to the work list. Each item in the work list is handled on its own separate thread.

TRACE Log events that provide additional information which allows HPSS development or support to trace what is happening in the code at a particular time (for example, the starting or stopping of a process). Normally meaningful only to HPSS support. Operator intervention is not required unless otherwise instructed by HPSS support.

TXN Log events that provide information on GHI file transactions from migrations, purges, stages, pins, recalls, and others.

WARN Log events for minor errors such as "Failed to create a directory". These warning messages are more like an inconvenience, but they may lead to additional problems. Operator intervention is probably not immediately required.

16.3. GHI cluster configuration

The environment variable, GHI_IGNORE_COMM_FAILURE, causes GHI to ignore communication failures when Db2 or HPSS is down. This will allow DESTROY, APPEND, and TRUNCATE DMAPI events to succeed. It will log an ALERT level log message with the information for the file that will be orphaned. GHI_IGNORE_COMM_FAILURE can be set to on in `/var/hpss/etc/env.conf`. The default behavior is for this variable to be set to off. When GHI_IGNORE_COMM_FAILURE is set to on, it will generate orphaned data in HPSS which can be cleaned up using ghiverifyfs(7) and the associated log messages. Anytime this environment variable is changed the ED process will need to be restarted so that the new value is picked up.

16.4. GHI file system configuration

The GHI file system configuration settings define how GHI works within the given file system and how data is handled during migrations, purges, stages, and backups. In addition, the GHI file system configuration settings define how to monitor the system and the access control of the file system. These settings can be listed with ghilsfs(7) and changed with ghichfs(7). For certain file system configuration changes, the IOMs for that file system will need to be manually restarted. Refer to the description of the file system configuration setting to determine if an IOM restart is required. Here is a list and a brief description of each GHI file system configuration setting:

Minimum Number of Files in an Aggregate (ghichfs --minagg)

This value, together with "Max Files Per Aggr" (see the following setting) determines whether all files selected for aggregation are migrated. Selected files are migrated into an aggregate file

containing "Max Files Per Aggr" until the number remaining to be migrated is under *minagg* value. If there are fewer remaining files than the *minagg* value, they will not migrate until a subsequent migration policy is executed and the number of files are greater than the *minagg* value. During backups, this minimum value does not have any effect because all selected files get migrated.

Maximum Number of Files in an Aggregate (ghichfs --maxagg)

The maximum number of files to be placed in an aggregate.

Aggregate Index COS (ghichfs --aggcos)

The HPSS class of service used to store ISHTAR index files. This COS should not purge the index files from an HPSS disk to maintain ISHTAR listing performance. Acceptable values are integers greater than or equal to zero that correspond to a COS configured in HPSS. A value of zero will use the default HPSS COS. If this setting is changed the IOMs for the file system will need to be manually restarted.

Aggregate Thread Pool Size (ghichfs --aggtps)

The maximum number of threads to use for copying Spectrum Scale files to the ISHTAR file in HPSS. Refer to the **–T** option in ISHTAR. Acceptable values are integers greater than zero. If this setting is changed the IOMs for the file system will need to be manually restarted.

Bulk Count Backup (ghichfs --bbc)

This is the file grouping count used by backups. If you divide the total number of files being backed up by this number you get the number of backup files generated.

Backup COS (ghichfs --bucos)

The HPSS class of service used to store namespace and attribute files (metadata files) created as part of a backup. Acceptable values are integers greater than or equal to zero that correspond to COS configured in HPSS. Zero will use the default HPSS COS. If this setting is changed the IOMs for the file system will need to be manually restarted.

HPSS Junction (ghichfs --junct)

The HPSS junction name used to link to the fileset of a subsystem in the HPSS namespace. It must not be blank; use a forward slash if a directory is not specified. If this setting is changed the IOMs for the file system will need to be manually restarted.

HPSS Base Path (ghichfs --basep)

The high-level directory to store migrated files into HPSS. This will be placed directly under "HPSS Junction". If this setting is changed the IOMs for the file system will need to be manually restarted.

HPSS Backup Path (ghichfs --bupath)

The directory in HPSS under which all system data will reside. If this setting is changed the IOMs for the file system will need to be manually restarted.

Performance Logging (ghichfs --perf)

Types of file-transfer performance logging. If this setting is changed the IOMs for the file system will need to be manually restarted. To set the type of performance logging, create a space-

separated or comma-separated concatenation of the following values, which can be expressed in any order:

ED Log performance of the ED's handling of DMAPI events.

ILM Log performance of policies executed via the ghiapplypolicy(7) script.

IOM Log performance of the IOM's handling of data transfer requests.

ISHTAR Log performance of the ISHTAR's handling of data transfers into/out of aggregates.

PIO Log performance of the PIO portion of non-aggregate data transfers.

SD Log performance of the SD's handling of data transfer requests.

Two additional values are allowed, which are mutually exclusive with each other and with all the above-listed values:

All Produce all possible types of performance logging.

None Produce no performance logging.

The default setting is "None". Performance logging can result in exceedingly large data files because unlike the GHI error logs, which are capped at 10 MB each, the performance log has no upper limit. For this reason, performance logging is normally enabled only to gather supporting data if performance issues are suspected.

Purge Only if on Tape (ghichfs --poiot)

This setting determines whether GHI checks that a Spectrum Scale file that has already been migrated to an HPSS disk before it executes a GHI purge on that file. The possible values are "true" and "false". A setting of "true" means checking with HPSS that every to-be-purged file has been migrated to HPSS Tape. If the file is not on HPSS tape, GHI will not purge it from Spectrum Scale. This is more resource-intensive, but will increase the integrity of the data by making sure it is on tape. A setting of "false" means that all files selected for purging will be purged if they have been migrated to HPSS disk regardless of whether the files have been migrated to HPSS tape.

Purge File Size (ghichfs --pblock)

The amount of file data, in bytes, to remain resident in the Spectrum Scale file system after a file is purged. Acceptable values are integers greater than or equal to zero. The actual amount of data which remains in a file after purging is dependent upon Spectrum Scale.

ED Max Connections (ghichfs --edmaxc)

The number of concurrent open connections the event daemon will allow. Acceptable values are integers greater than or equal to zero. A value of zero will cause the service to reject all connections.

ED Thread Pool Size (ghichfs --edtps)

When the ED captures DMAPI events from Spectrum Scale it sends them to this thread pool. The thread pool sends them to the SD. On most production systems, the default setting is more than sufficient. Acceptable values are integers greater than or equal to zero. A value of zero will cause the service to report being busy and reject all new work.

ED Request Queue Size (ghichfs --edrqs)

The number of slots in the wait list of DMAPI Events. The wait list is used to keep the overflow requests when the ED Thread Pool is filled with DMAPI events. Acceptable values are integers greater than or equal to zero. A value of zero will disable the request queue, causing requests to be rejected if all threads are busy.

ED Port Number (ghichfs --edport)

The port used by the scheduler daemon to communicate with the event daemon. Valid values are integers from 0 to 65535.

IOM Max Connections (ghichfs --iomaxc)

The maximum number of concurrent open connections the IOM will allow. This should be the number of connections to the scheduler daemon and any administrative utility. Acceptable values are integers greater than or equal to zero. A value of zero will cause the service to reject all connections. If this setting is changed the IOMs for the file system will need to be manually restarted.

IOM Thread Pool Size (ghichfs --iotps)

The number of requests each IOM will process at a time. The value should be based on the number of required concurrent transfers. Acceptable values are integers greater than or equal to zero. A value of zero will cause the service to report being busy and reject all new work. Do not alter without input from HPSS support. If this setting is changed the IOMs for the file system will need to be manually restarted.

IOM Request Queue Size (ghichfs --iorqs)

The number of slots in the wait list. The wait list is used to keep the overflow requests when the thread pool fills up. Acceptable values are integers greater than or equal to zero. A value of zero will disable the request queue, causing requests to be rejected if all threads are busy. If this setting is changed the IOMs for the file system will need to be manually restarted.

IOM Monitor Flag (ghichfs --iomon)

This flag indicates that the scheduler daemon should perform IOM activity monitoring (value = "on") or should not be performed (value = "off"). If this setting is changed the IOMs for the file system will need to be manually restarted.

IOM Monitor Frequency (ghichfs --iomonf)

This is the frequency in seconds (integer greater than zero) at which information will be logged to <IOM Monitor Output Path>. If this setting is changed the IOMs for the file system will need to be manually restarted.

IOM Monitor Output Path (ghichfs --iomonp)

This is the path used to store the monitor output log. There is no size limit imposed. With each

<IOM Monitor Frequency> iteration, approximately 70 bytes plus 100 bytes for each idle IOM and 200 bytes for each IOM actively transferring a file will be written to the monitor output log. Thus, if the frequency is every 10 seconds and there are 50 IOMs, the log could grow by as much as (70 + 200 × 50) × 6 = approximately 60 KB per minute. This monitor file will display information about the IOM in the form of columns. If this setting is changed the IOMs for the file system will need to be manually restarted. Here is an example of what the IOM monitor prints out:

```
********** Thu Sep 28 13:38:11 2017 **********
Active , TC: 4, F: 0, R: 0, WL: 0 , RP:* 164MB, Node: davis.clearlake.ibm.com:8012
********** Thu Sep 28 13:38:41 2017 **********
Active , TC: 4, F: 0, R: 0, WL: 0 , RP:* 164MB, Node: davis.clearlake.ibm.com:8012
```

Each abbreviation means:

Active This is the operational state of the IOM. It can be in one of three states:

- **Active**. Ready to transfer data.
- **Econn**. Disconnected. The IOM is not able to connect to GHI in this state and needs troubleshooting.
- **Standby**. Connected to the SD, but the IOM is unable to process jobs and needs time to become active; or if it is in standby for longer than 30 minutes, troubleshoot why it is still in standby.

TC The number of transactions this IOM has processed.

F Number of failures this IOM has processed.

R Number of requests left to be processed.

WL Amount of work left for the current job in MB.

RP Rate processed.

Node Hostname of the IOM.

LJ Longest job IOM has been working on. There are many jobs possible in this field:

- **migrate**. Migrating data from Spectrum Scale to GHI.
- **purge**. Punching holes or purging data from Spectrum Scale.
- **stage**. Synchronous recall of data from HPSS.
- **recall**. Asynchronous recall of data from HPSS.
- **reset**. Resetting attributes from a Spectrum Scale file so that it no longer points to an HPSS file.
- **backup**. Backing up the metadata from Spectrum Scale to HPSS.
- **restore**. Restoring the metadata from a GHI backup to Spectrum Scale.
- **sort_TS**. Used by **ghi_stage**; sorting the files into tape unit list.
- **order_TS**. Used by **ghi_stage**; ordering the tape unit lists.
- **TS_recall**. Used by **ghi_stage**; recalling data from HPSS.

SD IOM Max Connections (ghichfs --simaxc)

The number of connections the scheduler can have. The value should be the number of connections to the IOMs and any administrative utility. At a minimum, this value needs to be the number of IOM connections plus one.

SD IOM Thread Pool Size (ghichfs --sitps)

The number of threads used to work off DMAPI events. Acceptable values are integers greater than or equal to zero. A value of zero will cause the service to report being busy and reject all new work.

SD IOM Request Queue Size (ghichfs --sirqs)

This is the overflow queue for IOM transfer responses. If it is exceeded, then it will fail the transfers. If enough of these failures occur, the system will deadlock. The value should be larger than the number of IOMs multiplied by the number of worker threads. It should only be changed with feedback from HPSS support. Acceptable values are integers greater than or equal to zero. A value of zero will disable the request queue, causing requests to be rejected if all threads are busy. The sum of the following SD IOM Thread Share values (sidm, sibu, simg, sirc, siad) should not exceed the SD IOM Request Queue Size.

SD IOM DMAPI Thread Share (ghichfs --sidm)

Share of SD IOM threads for processing DMAPI requests in a round-robin schedule. Acceptable values are integers greater than or equal to zero. If the thread share is zero, SD will not schedule any DMAPI requests.

SD IOM Backup Thread Share (ghichfs --sibu)

Share of SD IOM threads for processing backup requests in a round-robin schedule. Acceptable values are integers greater than or equal to zero. If the thread share is zero, SD will not schedule any backup requests.

SD IOM Migrate Thread Share (ghichfs --simg)

Share of SD IOM threads for processing migrate requests in a round-robin schedule. Acceptable values are integers greater than or equal to zero. If the thread share is zero, SD will not schedule any migrate requests.

SD IOM Recall Thread Share (ghichfs --sirc)

Share of SD IOM threads for processing recall requests in a round-robin schedule. Acceptable values are integers greater than or equal to zero. If the thread share is zero, SD will not schedule any recall requests.

SD IOM Restore Thread Share (ghichfs --sirs)

Share of SD IOM threads for processing restore requests in a round-robin schedule. Acceptable values are integers greater than or equal to zero. If the thread share is zero, SD will not schedule any restore requests.

SD IOM Admin Thread Share (ghichfs --siad)

Share of SD IOM threads for processing admin requests in a round-robin schedule. Acceptable values are integers greater than or equal to zero. If the thread share is zero, SD will not schedule any admin requests.

SD Client Max Connections (ghichfs --scmaxc)

This is the connection between ILM client workers and the SD. The SD requires one connection per client. It should be set to the lower of either the number of files in the file system divided by the backup bulk count or by the GPFS limit of ILM scripts multiplied by the number of clients allowed to run GHI policies. That will set it to the number of clients expected to run. The default value should be fine unless the file system exceeds one billion files. Acceptable values are integers greater than or equal to zero. A value of zero will cause the service to reject all connections.

SD Client Thread Pool Size (ghichfs --sctps)

The number of threads used to work off DMAPI events. Acceptable values are integers greater than or equal to zero. A value of zero will cause the service to report being busy and reject all new work.

SD Client Request Queue Size (ghichfs --scrqs)

This is the overflow queue for SD ILM requests. If it is exceeded, then it will fail the transfers. Acceptable values are integers greater than or equal to zero. A value of zero will disable the request queue, causing requests to be rejected if all threads are busy. It should only be changed with feedback from HPSS support.

SD Port Number (ghichfs --sdport)

The port used by the scheduler daemon to communicate with the I/O managers. Valid values are integers from 0 to 65535. If this setting is changed the IOMs for the file system will need to be manually restarted.

SD Monitor Flag (ghichfs --sdmon)

This flag indicates that the scheduler daemon should perform monitoring (value = "on") or should not be performed (value = "off").

SD Monitor Frequency (ghichfs --sdmonf)

This is the frequency in seconds (integer greater than zero) at which information will be logged to <SD Monitor Output Path>.

SD Monitor Output Path (ghichfs --sdmonp)

This is the path used to store the monitor output log. There is no size limit imposed. With each <SD Monitor Frequency> iteration, approximately 250 bytes (that is, 1/4 KB) will be written to the monitor output log. Typically, archiving this file once per year is sufficient. This monitor file will display information about what is happening in the SD in the form of columns. Here is an example of what the SD monitor prints out:

```
********************** Thu Sep 28 12:50:12 2024 **********************
Active Q:0|F:none W:0|F:0 M:0/0/0|F:0/0/0 R:0/0|F:0/0 P:0|F:0 D:0|F:0 B:0|F:0 I:1/1

********************** Thu Sep 28 12:50:42 2024 **********************
Active Q:0|F:none W:0|F:0 M:100/1/0|F:0/0/0 R:200/10|F:0/0 P:0|F:0 D:0|F:0 B:0|F:0
I:1/1
```

Each abbreviation means:

Active Q The number of events.

W The number of items all GHI IOMs are currently processing.

M The number of migrated files to HPSS (Total aggregate files/Aggregates/Non-aggregates).

R The number of staged files from HPSS. (Total files staged / stage request complete).

P The number of purged Spectrum Scale files.

D The number of destroyed processes.

B Latest valid backup index.

I The number of IOMs (active/total).

F Associated with each column there is an "F" for "Failure". This is the number of failures for the respective operation.

These statistics will increment as GHI completes jobs, but if the SD is restarted, these statistics will be reset to zero, except for the backup and IOM columns. There is no way to reset these statistics without restarting the SD.

Checksum on Migrate (ghichfs --com)

By default, checksum-on-migrate is turned off and GHI does not checksum any user files. Care should be taken when enabling this option that the IOMs have sufficient CPU bandwidth to generate checksums without impacting migration performance. When checksum on migrate is

enabled, only non-aggregate files are checksummed during the migration process, and those checksums are stored in HPSS. HPSS file checksums can be viewed using the ghi_ls tool. HPSS will verify the checksum again when the files are written to tape.

Stage on Demand (ghichfs --sod)

By default, stage-on-demand is turned on and GHI allows files that are purged from the Spectrum Scale file system to automatically be staged when a user accesses the files. Stage-on-demand allows the user to turn DMAPI stages on or off. If DMAPI stages are turned off, GHI will report an error when a file is purged from the Spectrum Scale file system and an end user tries to access it.

DMAPI Stage Errno (ghichfs --dse)

This is the error number that is reported by GHI when stage-on-demand is turned off. The default number is 7500, but it is configurable to any positive number.

16.5. GHI IOM configuration

IOM_node:port (ghideliom <IOM_node>:<port>)

IOMs are specified using a pseudo key or description. They are specified as *--Node:Port* for the "Key" (for example, `--node3:8023`) or if the "--" is omitted, it becomes a "Description". For example:

```
% ghilsiom gpfs3fs
Key        Description               Value (# comment)
 ------------------------------------------------------
IOM Node:Port
miami.clearlake.ibm.com:8032
--asn       Active Session Node       TRUE
--etr       Estimated Transfer Rate   1000
--chunksize Transfer Chunk Size       1TB
```

An IOM's node or port cannot be changed with ghichiom(7). If these settings need to be changed, the entire IOM must be deleted and re-created via ghideliom(7) and ghiaddiom(7).

Active Session Node (ghichiom --asn)

A value of "true" indicates the IOM should be allowed to run whenever the node the IOM is configured on is the GHI session node. A value of "false" means the IOM process will not run when the node is also the GHI session node.

Estimated Transfer Rate (ghichiom --etr)

This is the estimated transfer rate available for the node, in bytes per second, and is used as the initial value for load-balancing the IOMs. The value is adjusted in real-time once the actual transfer rate is determined. Acceptable values are an integer greater than zero optionally followed by a unit indicator: KB, MB, GB. If this argument is omitted, the default is GB.

Transfer Chunk Size (ghichiom --chunksize)

The amount of file data, in bytes, for non-aggregate transfers to HPSS per I/O request. Setting this

value too low can cause performance issues due to increased protocol overhead. Setting this value too high can cause HPSS I/O to time out, depending on transfer rates. Acceptable values are integers greater than zero.

Chapter 17. System monitoring

GHI provides a utility, **ghi_mon**, to monitor the scheduler daemon and the progress of IOM activities. Monitoring can be scheduled to start when the SD or IOMs are online. This is done by configuring it via the *GHI Configuration Utility* or the user can call **ghi_mon** <*task*> to start the task.

Additionally, each server (except the MD) may be sent a SIGHUP to dump its connections and RPC subsystem state. This can be helpful in identifying bottlenecks and some kinds of throttling.

17.1. Scheduler

The figure depicts the internals of the scheduler.

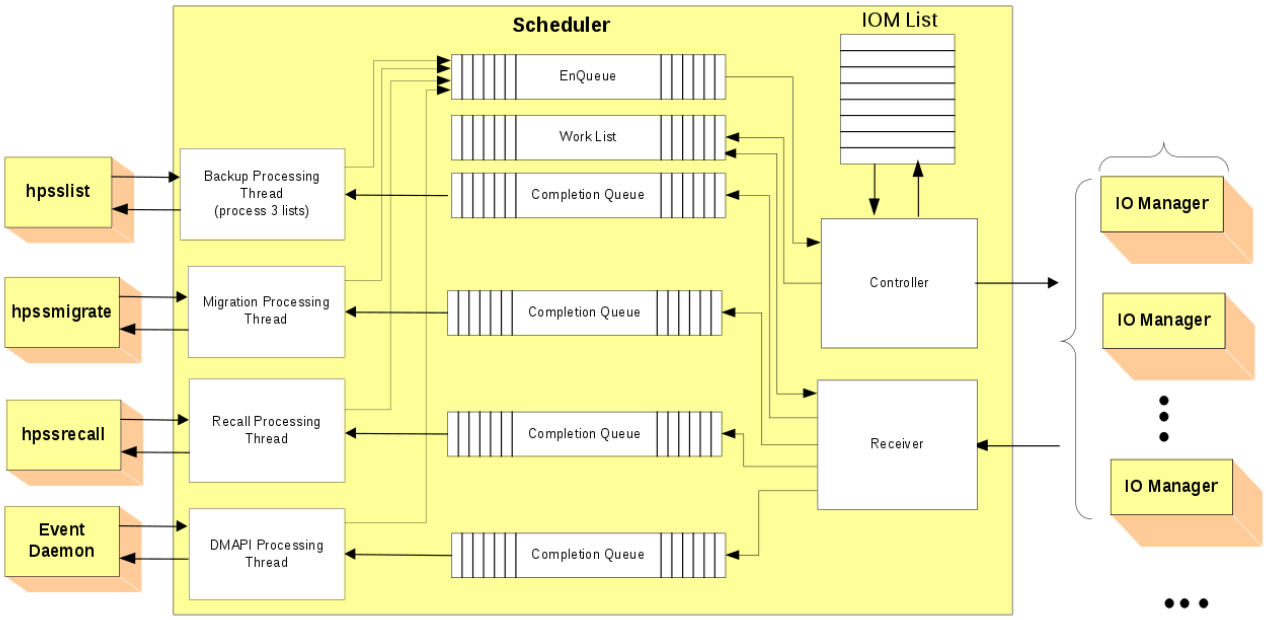

Figure 11. Scheduler internals

The scheduler contains four different schedule queues:

- Namespace backup requests
- Migration requests
- Recall requests
- DMAPI requests

As requests come in, the scheduler places the items on the appropriate queue. As they are worked off, they are placed on the work list. When monitoring the scheduler, a single line item is printed containing the following information:

Mode

The mode of the SD. "Active": the SD is running in active mode; "Backup": the SD is running a backup.

Queued

Current number of requests on the scheduler queue.

Working

Current number of entries being worked off.

Migrations(A/N)

Total number of aggregates/non-aggregates that have been processed since the scheduler was started. Aggregates are counted as one per aggregate and not the total number of requests within an aggregate.

Recalls

Total number of files that have been recalled since the scheduler was started. This can be misleading if there were multiple recalls in a single aggregate. The aggregate is only counted as a single increment.

Stages

Total number of stage request.

Purged

Total number of files that have been purged.

Failed

Total number of migrations, recalls, stages, or purges that have failed.

IOM(A/T)

Total number of IOMs that are active or total number of IOMs configured.

17.2. I/O manager

The figure depicts the internals of the I/O Manager.

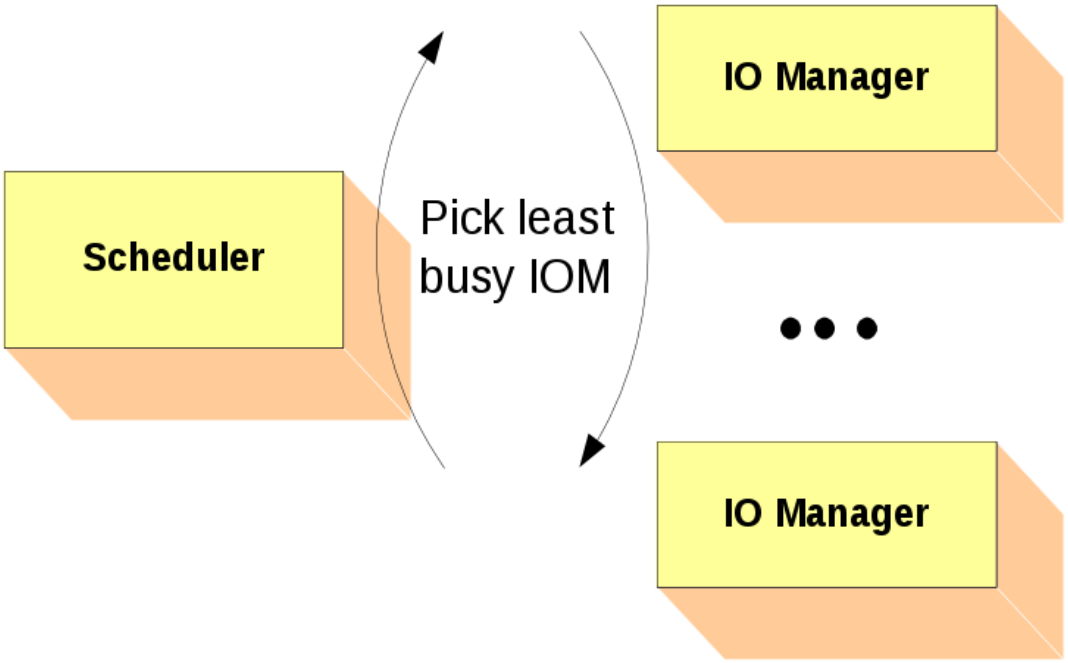

Figure 12. I/O manager internals

The scheduler sends requests to the configured/active IOMs in a round-robin fashion. The IOMs spawn off requests as they are received from the scheduler. The IOM can be in four different states:

Active The IOM has the file system mounted and all the connections are valid.

Inactive The IOM is running on the session node and it is configured to be in an inactive state.

Standby Either the file system is not mounted on the node or the connection from the IOM to the scheduler is not valid.

Econn The scheduler has lost connection to the IOM.

When monitoring the IOMs, one or more lines are printed for each configured IOM. The first line contains the following information:

State The current state of the IOM (see node definitions above).

Requests Number of completed requests.

Errors Number of errors, total and by request type, encountered since the IOM was started.

Processing Rate How busy the I/O manager is. An asterisk ("*") indicates that the I/O manager will be selected next for work.

Node Hostname/port where the IOM is running.

If the IOM is currently transferring a file, a second line is displayed that shows information on the longest running active transfer.

17.3. File transfer performance

Monitor the performance of a file transfer at various stages in the process to determine possible bottlenecks. Statistics are written to `/var/hpss/ghi/tmp/ghi_perf.log`. This file can grow to unlimited size therefore, performance monitoring is usually disabled. File transfer requests can enter the system in three ways: via the ED from a DMAPI event; via an ILM policy scan; or via the SD from the ghi_stage(7) utility.

File transfers can be monitored for the following processes and are configured via the GHI Configuration Utility. See "Performance Logging (**ghichfs --perf**)" in the GHI file system configuration section.

ED Transfers can be monitored starting with the ED's receipt of a DMAPI event from Spectrum Scale until it returns the response back to Spectrum Scale. The reported elapsed time includes all processing needed to generate the response from the SD, IOM, and PIO or ISHTAR.

ILM The complete lifetime of an ILM process as it handles its share of policy output resulting from execution of the ghiapplypolicy(7) script. The reported elapsed time (the process' lifetime) includes all processing by the SD, IOM, and PIO or ISHTAR. Execution of **ghiapplypolicy** may result in launching multiple ILM processes, each with its own reporting trail.

SD Transfers can be monitored starting with the SD's receipt of a file transfer request from either the ED, **ghi_stage**, or an ILM process until it returns a response back to the ED, **ghi_stage**, or ILM process. The reported elapsed time includes processing by the IOM and PIO or ISHTAR.

IOM Extends from the IOM's receipt of a file transfer request from the SD until it returns the response back to the SD. The reported elapsed time includes the PIO or ISHTAR processing.

ISHTAR For aggregate file transfers, extends from IOM's launching of the ISHTAR process until it detects that the process has terminated. ISHTAR does all the DMAPI processing so this is not reported under the IOM.

PIO For non-aggregate file transfers, extends just around the actual PIO processing to affect the data transfer. PIO does no DMAPI processing so this is reported under the IOM.

17.3.1. Tracking file transfer requests

There are times when it is necessary to gather the records generated by a given file transfer request from the ILM policy. To do this, run with the **-d** or **-D** in the policy's OPTS field so that the generated filelists remain available after the process is completed. As a file transfer request gets processed, one or more records for each of the above types (ED, ILM, SD, IOM, ISHTAR, and PIO) may get

written to the performance log file. For example, an ILM policy run (ghiapplypolicy(7)) on non-aggregated files will result in: 1) one ILM and one SD record for each ILM process that gets spawned; and 2) one IOM and one PIO record for each file selected by the policy run. Records will most likely get written to performance log files on multiple nodes. The ED and SD records are always written on the GHI session node, while ILM, IOM, PIO, and ISHTAR records are written on the node on which the process runs. When multiple ILM processes get spawned from **ghiapplypolicy**, the processes may be on multiple nodes. The SD parcels out work to the IOMs independent of the requested source.

And lastly, the order of records within a file may not correspond to the actual processing order. For instance, the ILM record may precede an IOM record, or an IOM record may precede an ISHTAR record even though one would expect the IOM to complete its work and write its record prior to reporting its status to the SD. The SD reports status to the ILM, which would then write its record (likewise with the IOM and ISHTAR). Underlying timing uncertainties with parallel processing account for this effect-before-cause behavior. Keeping all the above in mind, multiple queries may need to be made because the transfer identifier may be different as the transfer proceeds through the system. It is either the inode, igen, and snapID for non-aggregate transfers, or the filelist's pathname for aggregate transfers or in the ILM portion of processing before the SD extracts individual non-aggregate requests from the policy output. Once a file (or filelist) of interest has been identified, all records of the associated performance log files need to be queried for records containing the associated identifier(s). For example, assuming the inode and igen are available for the file of interest and it was included in the policy output as shown below, run the following on all nodes in the cluster to pull the associated records from the performance log file(s):

```
% PL="/var/hpss/ghi/tmp/ghi_perf.log
% GO="/common_disk/grep_output"
% grep "/ghi3/scratch/.ghi/mmPolicy.ix.25190.C3F04B6B.1" $PL >>$GO
% grep "Inode: 0.194799, Igen: 65555" $PL >>$GO
```

Note the use of the double right-arrows (">>") to append to the previous run's output to "grep_output". Once all the records of interest are collected, they can (based upon the discussions of each record type presented above) be sorted into the proper order to produce the processing trail. It is not sufficient to merely sort on the timestamps because the times on the nodes may not be in sync with each other.

17.3.2. Performance log record formats

This section describes the format for each of the record types which may be written to the performance log file; that is, ED, ILM, SD, IOM, PIO, and ISHTAR. Additionally, at GHI startup, daemon restart, or refresh via SIGHUP, each daemon that will be writing performance data to the log outputs has a timestamp record like this:

```
    ---- 2010-02-25-14:09:15 2486231360 ED /ghi4
```

The four dashes ("----") in the example above is where measurement data would normally be shown. In this example, the "----" is displayed because this line in the log output is indicating the

daemon-startup for the ED, so no measurement data needs to be displayed for this line. Following the timestamp is the daemon's PID, the daemon type (ED, SD, or IOM), and the final field is the associated mount point. As with GHI daemon logs, each IOM writes to its own local performance log file.

ED transfer record

An ED transfer record is created (but not yet written to the performance log file) as soon as the ED has queried enough information from Spectrum Scale to determine that the file is not already resident in Spectrum Scale and to make the recall request to HPSS. The newly-created record will contain the file data and current (server clock) time for the start time. The request will be forwarded to the SD and the ED will wait on a response. When the response arrives, the response data from the SD will be added to the record and the completed record written to the performance log file. The ED record format is:

```
ED timestamp PID recall status size elapsed_time file_data
```

Where:

Timestamp	Date and time the record was written to file.
PID	Process ID of the ED which created the record.
status	Either "SUCCESS" or "FAILURE".
size	Size of the file (bytes).
elapsed_time	End time minus start time (seconds).
file_data	RqstID: 0 Inode: Inode, Igen: Igen, SnapID: 0.0.

Example of an ED transfer record:

```
ED 2011-01-12-13:47:31 1074145600 recall SUCCESS 512000 0.000389099
RqstID: 0 Inode: 0.141457, Igen: 65542, SnapID: 0.0
```

The record format will be identical for both aggregates and non-aggregates because DMAPI requests do not differ between the two file types.

ILM transfer record

An ILM transfer record is created (but not yet written to the performance log file) as soon as the ILM process gets far enough in its processing to create a record. The newly-created record will contain the pathname of the associated filelist and the current (server clock) time for the start time. The request will be forwarded to the SD and the ILM process will wait on a response. When the response arrives, the response data from the SD will be added to the record and the completed record written to the performance log file. The ILM record format is:

```
ILM timestamp PID operation status 0 elapsed_time filelist
```

Where:

timestamp Date and time the record was written to file.

PID Process ID of the ILM process which created the record.

operation Either "migrate", "purge", "recall", or "list".

status Either "SUCCESS" or "FAILURE".

elapsed_time End time minus start time (seconds).

filelist Pathname of the associated filelist.

Example of an ILM transfer record:

```
ILM 2011-01-12-19:54:51 3970355248 purge SUCCESS 0 16.7519
/gpfs3/scratch/.ghi/mmPolicy.ix.17350.45C7188E.1
```

SD transfer record

An SD transfer record is created (but not yet written to the performance log file) as soon as the SD gets far enough in its processing to create a record. For aggregate requests, this is as soon as the SD receives the request. For non-aggregates, a record is created for each file in the associated filelist as soon as it is read from the filelist. The newly-created record will contain either the pathname of the aggregate filelist or the inode and igen of the non-aggregate file, and the current (server clock) time for the start time. When the transferring IOM indicates transfer is complete, the response data from the IOM will be added to the record and the completed record written to the performance log file. The SD record format is:

```
SD timestamp PID operation status size elapsed_time file_data
```

Where:

timestamp Date and time the record was written to file.

PID Process ID of the SD which created the record.

operation Either "migrate", "purge", or "recall".

status Either "SUCCESS" or "FAILURE".

size Size of the file or aggregate (bytes).

elapsed_time End time minus start time (seconds).

file_data Depends upon whether the request for an individual file or a filelist.

If a file: RqstID: RqstID Inode: Inode, Igen: Igen, SnapID: SnapID.

If a filelist: RqstID: RqstID Filename: pathname, Size: size

Example of an SD transfer record:

```
SD 2011-01-12-13:53:33 1076185408 migrate SUCCESS 1048576000 133.899
RqstID: 1520863553 Inode: 0.26117, Igen: 65544, SnapID: 0.0
```

IOM transfer record

An IOM transfer record is created (but not yet written to the performance log file) as soon as the IOM determines that PIO needs to be called to complete the transfer. The newly-created record will contain the inode and igen of the non-aggregate file and the current (server clock) time for the start time. When the transfer is complete, the transfer data will be added to the record and the completed record written to the IOM's local performance log file.

The IOM transfer record format is:

```
IOM timestamp PID operation status size elapsed_time file_data
```

Where:

Timestamp	Date and time the record was written to file.
PID	Process ID of the IOM which created the record.
operation	Either "migrate", "recall", or "backup".
status	Either "SUCCESS", "XFER_FAILED", "GETATTR_FAILED", "SETATTR_FAILED", or "REGIONS_FAILED".
size	Size of the file or aggregate (bytes).
elapsed_time	End time minus start time (seconds).
file_data	Depends upon whether the request is for an individual file or a filelist. If file: RqstID: RqstID Inode: Inode, Igen: Igen, SnapID: SnapID If filelist: RqstID: RqstID Filename: pathname, Size: size

An example of an IOM record follows. Note that the third line (beginning with "/gpfs3/scratch··· ") has been split into two separate lines for the purpose of this documentation. That line and the fourth (indented) line should thus be considered as parts of the same record line, with no breaks or spaces.

```
IOM 2011-03-23-15:56:32 1090451776 backup SUCCESS 3717 0.278038 RqstID:
1315662170 Filename:
/gpfs3/scratch/.ghi/backup_file_mtime/
  mmPolicy.ix.5900.09AFFAB9.2/mmPolicy.ix.5900.09AFFAB9.2_0.data,
Size: 0.0
```

PIO transfer record

A PIO transfer record is created (but not yet written to the performance log file) just prior to the IOM initiating the PIO process with HPSS. The newly created record will contain the inode and igen of the non-aggregate file and the current (server clock) time for the start time. As soon as PIO processing is completed, the transfer data will be added to the record the completed record written to the IOM's local performance log file. The PIO record format is:

```
PIO timestamp PID operation status size elapsed_time throughput
file_data
```

Where:

Timestamp	Date and time the record was written to file.
PID	Process ID of the IOM which created the record.
operation	Either "migrate", "recall", or "backup".
status	Either "SUCCESS" or "XFER_FAILED".
size	Size of the file (bytes).
elapsed_time	End time minus start time (seconds).
throughput	Size / elapsed_time / 1000.
file_data	Depends upon whether the request is for an individual file (migrate, recall) or a filelist. If file: RqstID: RqstID Inode: Inode, Igen: Igen, SnapID: SnapID If filelist: RqstID: RqstID Filename: pathname, Size: size

Example of a PIO record:

```
PIO 2011-03-31-11:24:06 1121962304 migrate SUCCESS 524288 9.48336
55.2851 RqstID: 1321016045 Inode: 0.136201, Igen: 65538, SnapID: 0.0
```

ISHTAR transfer record

An ISHTAR transfer record is created (but not yet written to the performance log file) as soon as the IOM determines that ISHTAR needs to be called to complete the transfer. The newly created record will contain the pathname of the aggregate filelist and the current (server clock) time for the start time. When the transfer is completed, the transfer data will be added to the record and the completed record written to the IOM's local performance log file. The ISHTAR record format is:

```
ISHTAR timestamp PID operation status size elapsed_time throughput
file_data
```

Where:

timestamp	Date and time the record was written to file.
PID	Process ID of the IOM which created the record.
operation	Either "migrate" or "recall".
status	Either "SUCCESS" or "FAILURE".
size	Size of the aggregate (bytes).
elapsed_time	End time minus start time (seconds).
throughput	Size / elapsed_time / 1000.
file_data	RqstID: RqstID Filename: pathname, Size: size.

Example of an ISHTAR record:

```
ISHTAR 2011-03-31-13:50:41 1120872768 migrate SUCCESS 204856 6.2463
32.7964 RqstID: 1310898517 Filename:
/gpfs3/scratch/.ghi/mmPolicy.ix.19415.F41F1DF9.1, Size: 0.204600
```

Chapter 18. Problem diagnosis and resolution

It is important to monitor the GHI system daily and address problems early before they become severe. System administrators need to take these actions daily:

- **Policy runs**. Save and review the output from each of the policy runs. The policy runs can take several hours; therefore, we recommend saving the output for each of the runs in a specific location so they can be reviewed. The policy runs generate *.ok and *.exc files. Use the "-b" option in the policy to tell GHI to save the files instead of automatically cleaning them up. It is up to the site administrator to delete these files.
- **Back up**. Take a backup and review the output. The output indicates success or failure and details of each failure for investigation.
- **SD and IOM**. Monitor the SD and IOM output from the **ghi_mon** utility. If configured in the GHI file system configuration, the output is activated automatically. Refer to the ghi.conf configuration file.

This chapter provides some common problems seen in GHI and how to resolve them. A problem may have more than one diagnosis and resolution.

18.1. Common infrastructure (communication) problems

Some commonly seen RPC and security-related problems with GHI are below:

18.1.1. A GHI server cannot communicate with another

Diagnosis 1: The target server may not have registered its RPC endpoint.

Resolution: Verify proper registration of the server with RPC. If shutting down the target server and restarting it does not fix the problem, you may have to manually delete the server's RPC entry. using the **rpcinfo -d <program number> <version>** command. Once you've deleted the registration you can try restarting GHI to re-register the service

Diagnosis 2: A network communication failure may exist or security may be blocking the communication.

Resolution: First verify the network is up and the server is running. A less obvious cause for the problem may be that the server is not accepting calls from the client because of security reasons. To fix this problem, make sure that the client and server are using consistent security policies and that they have authenticated properly.

Diagnosis 3: The `/var/hpss` file system may be full.

Resolution: Determine what is causing the file system to fill up. Common problems are `/var/hpss` is

too small or log files are not being archived out of this directory to free up space. Sometimes, a debug log turned on for problem resolution was not turned off and filled the file system.

Diagnosis 4: A server may be too busy to respond.

Resolution: If a server is very busy, other servers will not be able to communicate with it. To solve the problem, decrease the load on the server. For example, try increasing the server's thread pool size and/or maximum connection count, or moving the server to a different machine, or adjusting one of the server-specific configuration parameters.

Diagnosis 5: A node does not have a network route to an interface used by a server on a different host.

Resolution: Verify that all nodes (session node and IOM nodes) have network routes to the network interfaces required.

Diagnosis 6: The server may not have enough RPC connections configured that are necessary for communication.

Resolution: Increase the number of maximum connections for the appropriate server configuration.

Diagnosis 7: The server may have other configuration issues.

Resolution: Verify the appropriate server configuration, such as ulimits, xinetd settings, firewalls, hostnames, and so on.

Diagnosis 8: The Domain Name Service (DNS) is not reachable.

Resolution: Add all necessary entries to the `/etc/hosts` file. Terminate all GHI servers, Db2, and Kerberos. Restart the system without DNS, then fix the DNS.

18.1.2. A server cannot register its RPC info

Diagnosis: Stale RPC information may exist for the server in the RPC table.

Resolution: Issue the **rpcinfo -p** command to see if the RPC program number for the server interface is already registered. If the interface is registered it can be removed using the **rpcinfo** -d <*program number*> <*version*> command. Once you've deleted the registration you can try restarting GHI to re-register the service.

18.1.3. A server cannot obtain its credentials

Diagnosis: There may be a problem with the keytab file.

Resolution: Make sure the keytab file (usually in `/var/hpss/etc/`) is readable by the UNIX username under which the server is running. Make sure the key contained in the keytab file is the correct one. Look for extra versions of the server's key, which can interfere with the authentication process.

18.1.4. The connection table may have overflowed

Diagnosis: The server may be so heavily loaded that it is unable to free up connections quickly

enough.

Resolution: Reduce the load on the server. The problem may also indicate that a server is configured incorrectly or there is a software problem in handling connections properly. One solution is to increase the Maximum Connections parameter in the configuration for the specific server.

18.2. Server problems

18.2.1. The process manager dies after a mount request (PPC only)

Diagnosis: The PM core dumps with a stack dump.

Resolution: Set the HPSS_PTHREAD_STACK to 262144 in `/var/hpss/etc/env.conf`.

18.2.2. The mount daemon fails to get events

Diagnosis: Failed to retrieve DMAPI events.

Resolution: Verify the Session ID is valid for that node and is not owned by another node.

18.2.3. The mount daemon fails to respond to an event

Diagnosis: Failed to respond to a file system mount/unmount request.

Resolution: Verify the file system still exists.

18.2.4. The mount daemon fails to mount a file system

Diagnosis: Failed to mount the file system.

Resolution: Verify the file system still exists and it is not a GHI read-only file system.

18.2.5. The mount daemon fails to dismount a file system

Diagnosis: Failed to dismount the file system.

Resolution: Verify there are no open files in the file system and that no one else is currently using the file system. Use **lsof** to determine what is using the file system.

18.2.6. The event daemon fails to get events

Diagnosis: Failed to retrieve DMAPI events.

Resolution: Verify the file system still exists. Verify the Session ID is valid for that node and is not owned by another node.

18.2.7. The event daemon fails to respond to events

Diagnosis: Failed to respond to a request to access a file.

Resolution: Verify the event daemon process has not been restarted.

18.2.8. The event daemon fails to get attributes of a file

Diagnosis: When trying to get the attributes of a file, the call failed.

Resolution: Verify the file still exists. Verify the session ID is valid for that node and is not owned by another node.

18.2.9. The scheduler daemon is stuck in "QUIESCING_FS" mode

Diagnosis: All file transfer requests terminate with error code -19 (ENODEV) and the monitor output is showing the SD's state to be "QUIESCING_FS". This occurs when either an **unmount** command has been given or a GHI shutdown has been requested but not yet completed.

Resolution: It may be that the SD has begun to make the file system inactive but the file system could not be unmounted. There may be ongoing file transfers to or from an HSM which needs to complete before the SD can terminate and GHI can shut down completely. If a command to unmount the file system failed, try to remove the condition which is blocking its completion and re-try it, or try a forced unmount (first see the Spectrum Scale documentation on the **mmunmount** command). If a GHI **shutdown** command has failed, retry the command until it succeeds or use the **-f** (force) option. If it is desired to place the SD back into its normal operating mode, send a SIGHUP signal to the SD process ID on the GHI session node. View the SD monitor output to verify the SD's state returns to active. The area below the "Active Q" text in the monitor output should be blank and not showing any error messages. This should not be done after a GHI shutdown because the status of other GHI system processes is uncertain. This may be done after a failed unmount of a file system if the file system still shows as mounted, and you wish to continue normal operations without attempting another unmount.

18.2.10. Out of completion queues

Diagnosis: There are too many requests for the scheduler.

Resolution: The additional requests will not be lost; they will wait until some existing connections are complete and then the request will get scheduled.

18.2.11. Failed to set regions (punching a hole)

Diagnosis: Unable to punch a hole in a file using **dm_punch_hole**.

Resolution: Verify the file exists.

18.2.12. Recovery started for an IOM

Diagnosis: An IOM abnormally terminated. The requests that it was working on are being redirected to another IOM.

Resolution: There is no action to be taken.

18.2.13. Failed to get a DMAPI handle for a file

Diagnosis: A failure occurred when attempting to read the DMAPI handle for a file before performing the data transfer.

Resolution: Verify the file has not been deleted. Verify the session ID is still valid for that IOM.

18.2.14. The IOM is in ECONN mode

Diagnosis 1: The IOM shows ECONN during initializing.

Resolution: The IOM is initializing; wait until it finishes the connection logic.

Diagnosis 2: The IOM is configured incorrectly in `/etc/inittab`.

Resolution: Fix the entry and recycle **inittab (kill –1 <inittab pid>)**. The entry is added automatically by GHI, so this is unlikely the cause unless an administrator or configuration manager (such as Puppet) changed it.

Diagnosis 3: The authentication configuration is incorrect in `/var/hpss/etc`.

Resolution: Copy the `/var/hpss/etc/` directory from the HPSS Core Server location.

Diagnosis 4: There is a time difference between the IOM node and the session node that is greater than five (5) minutes.

Resolution: Run an NTP daemon on all the nodes or verify NTP is working, or sync-up the date and time with the **date** command.

18.2.15. IOM is in STANDBY mode

Diagnosis: IOM loses connection to the SD.

Resolution: Verify the scheduler daemon is running or recycle the IOM.

18.2.16. IOM failed to get a handle for a file

Diagnosis: Unable to get a handle for a file.

Resolution: A handle is an integer value assigned by the operating system to access a file. When the IOM is unable to get this handle, you will want to verify first that the file still exists. Second, verify the session ID is still valid for that IOM.

18.2.17. ISHTAR fails to run

Diagnosis 1: The `ndapi.log` reports that ISHTAR failed with `"ndad_keytab_check: failed (code=22) for principal <ghi cluster principal>"`.

Resolution: Issue a new `hpss.htar.keytab` and distribute the keytab file to all of the GHI nodes.

Verify the `/var/hpss/etc/unix.master.key` does not contain all zeros.

Diagnosis 2: ISHTAR failed with "18431" (unable to open file).

Resolution: Review the syslog messages for logs regarding the missing file.

Diagnosis 3: ISHTAR failed with "18432".

Resolution: The `htar.ksh` file contains the incorrect value for HPSS_HOSTNAME. Fix the `htar.ksh` file and run a quick command-line test to verify the fix:

```
/var/hpss/hsi/bin/htar.ksh cvf /ghi/testfile /etc/motd
```

Diagnosis 4: ISHTAR fails with `"-52 htar_GhiClose, Error setting managed regions"`.

Resolution: Verify the session ID is correct.

Diagnosis 5: ISHTAR fails to open the file in Spectrum Scale.

Resolution: Verify the spectrum scale file still exists.

18.2.18. ISHTAR appears to be hung or locked up

Diagnosis: The location for the ISHTAR temporary files is full: "WARNING: OUT OF SPACE writing HPSS archive - delaying/retrying".

Resolution: Verify the file system is not full. Kill the `htar.ksh` process. Clean up the file system or allocate more space. Rerun the policy.

18.2.19. A "-1 makeXHandle" error was encountered from a migration policy run

Diagnosis: Failed with a call to makeXHandle because the `/var/hpss/ghi/etc` directory is inconsistent with the rest of the nodes in the cluster.

Resolution: Issue a **kill -SIGHUP** <*pid*> command to the GHI configuration manager (**ghi_cm**) on the GHI session node to have it execute a cluster-wide configuration scan to validate and correct anything wrong. Recycle the IOM.

18.2.20. A "Failed to migrate files, RPCError = 0, rc = -1" error from a migration policy run

Diagnosis: The output from a migration policy reported a failure because the file system is GHI read-only.

Resolution: No resolution is possible. Allowing files to be migrated into HPSS from a GHI read-only file system could result in corruption of the HPSS data for the associated GHI full-access file system. If a file must be migrated into HPSS, it will need to be copied to the full-access file system and migrated from there.

18.2.21. A "-5 PIOXferMgr" error from a migration policy run

Diagnosis: The HPSS Movers are having issues (they are not in green state as shown in the HPSS GUI), there are errors in the Alarms & Events, or the `local.log` file.

Resolution: See "Chapter 1: HPSS Problem Diagnosis and Resolution" in the *HPSS Error Manual*.

18.2.22. A "-19 sd_quiesce_FS" error from a migration policy run

Diagnosis: The file system was unmounted or a shutdown of GHI was requested after the migration had been requested, but before the actual file transfer request was sent to an IOM.

Resolution: Re-run the migration after mounting the file system or after GHI is restarted.

18.2.23. A "-28 PIOXferMgr" error from a migration policy run

Diagnosis: The HPSS Movers are having issues (they are not green), or there are errors in the Alarms & Events or the `local.log` file.

Resolution: See "Chapter 1: HPSS Problem Diagnosis and Resolution" in the *HPSS Error Manual*.

18.2.24. A "-78 PIOXfer" error from a migration policy run

Diagnosis 1: The inetd was configured incorrectly.

Resolution: See the IOM problems above.

Diagnosis 2: There is a problem with one or more HPSS Movers.

Resolution: Verify the Movers are green. Look at the HPSS `local.log` file for error messages.

18.2.25. A "-19 sd_quiesce_FS" error from a recall policy run

Diagnosis: The file system was unmounted or a shutdown of GHI was requested after the recall had been requested, but before the actual file transfer request was sent to an IOM.

Resolution: Re-run the recall after mounting the file system or after GHI is restarted.

18.2.26. A "-78 PIOXfer" error from a recall policy run

Diagnosis: The xinetd was configured incorrectly.

Resolution: See the IOM problems above.

18.2.27. Problems with mounting file systems

Diagnosis 1: The **mount** command returned an error for "Stale NFS Handle".

Resolution: There is potentially a disk problem with the file system; run ghi_state(7). Also, verify the file system is configured with the ghilsfs(7) command.

Diagnosis 2: The **mount** command returned an error saying there was no handle for the mount event.

Resolution: Verify the PM and MD are running.

Diagnosis 3: The mount daemon is not running.

Resolution: Verify the session node did not failover. Verify the mount daemon is running on the session node and could take over the session ID.

Diagnosis 4: The **mount** command returned an error for "can't read superblock".

Resolution: Try to mount the file system on the Spectrum Scale cluster manager node first, then mount on the other nodes.

Resolution: There is potentially a disk problem with the file system; run ghi_state(7). Also, verify the file system is configured by executing the ghilsfs(7) command.

Resolution: The file system is read-only and no backup has yet been restored to it. To check, issue the following command:

```
% ghilsfs <FS> --bustat
```

The following result will confirm if a backup needs to be restored:

```
Key Description Value (# comment)
 ---------------------------------------------
File System Name <FS>
 --bustat  Backup Status   needs restore
```

18.2.28. Error indicating file is not managed by HPSS

Diagnosis: The purge policy that was configured is not excluding files that are already co-managed.

Resolution: Verify the policy has an entry to exclude files that are not managed by HPSS:

```
Rule "exclude_rule" EXCLUDE FROM POOL "system" WHERE MISC_ATTRIBUTES NOT
LIKE `%M%'
```

18.2.29. A file fails to purge data blocks from Spectrum Scale

Diagnosis 1: Verify the file was not deleted after it was selected as a purge candidate.

Resolution: There is nothing to be done here.

Diagnosis 2: Verify the file meets the purge policy criteria.

Resolution: Review the policy and the location of the file (via ghi_ls(7)).

Diagnosis 3: Verify the file is larger than one data block.

Resolution: Files that are less than or equal to one data block will not be selected as purge candidates.

18.2.30. Failed to read or write a file in the system

Diagnosis: The request returns an Input/Output error.

Resolution: Verify neither the event daemon nor the scheduler have been recycled recently. Verify the HPSS Movers are green. Also, look at the HPSS `local.log` to see if there are any error messages.

18.2.31. Reading or writing a file appears to hang

Diagnosis: The read or write request times out.

Resolution: Find out where the file resides in HPSS. If it is on tape, verify there is not an outstanding tape mount request and there is a free tape drive.

18.2.32. General problems with a utility or tool

Diagnosis: Problems with a utility.

Resolution: Check the syntax. Most utilities will print a usage summary if they are invoked with the **-?** option. Some utilities require several parameters to be specified that may not be obvious. Make sure the default arguments are being overridden when necessary. Many utilities use default values for several parameters. If the parameter is not overridden with a specific value, unexpected behavior may result. Refer to the limits in Starting GHI for the First Time to verify they were not exceeded.

18.2.33. The ghi_mon SD error count increases

Diagnosis: Numerous errors are encountered during migrations, recalls, stages, or purges.

Resolution: Review the actions performed and determine increase rates to determine which action is generating the errors. Review the **ghi_mon** output for the IOM as well to determine which IOM is encountering the problem.

18.2.34. The ghi_mon IOM error count increases

Diagnosis: The errors displayed from **ghi_mon** indicate the IOM was failing when performing data transfers.

Resolution: Review the exception files ending with the `.exc` extension located in `<mount point>/scratch/` for that IOM (if running with the **–d** or **–D** option). Otherwise, run a migration and review the output from the policy run to determine what are the issues. Also, review the `local.log` file to see what errors are occurring from the HPSS Movers.

18.2.35. The ghi_mon shows the SD restarted

Diagnosis: The session node failed over.

Resolution: View the system log to determine why the scheduler was recycled. If the scheduler daemon terminates again, turn on additional logging so that the next time the scheduler is recycled, additional error information can be viewed.

18.2.36. The ghi_mon shows the SD to be "QUIESCING_FS"

Diagnosis: The file system has been unmounted, GHI is shutting down, or both.

Resolution: See The scheduler daemon is stuck in "QUIESCING_FS" mode.

18.2.37. Failed to connect to the SD

Diagnosis: The SD is not running because the file system is unmounted.

Resolution: Mount the file system and verify the SD is started.

18.2.38. GHI backup cannot communicate with Db2

Diagnosis: Db2 is not running.

Resolution: Verify Db2 is running and restart if necessary. Authenticate as the Db2 instance owner and start Db2 with the **db2start** command. There is no harm in executing this if the Db2 instance is already running.

18.2.39. Failed to back up a file from a snapshot

Diagnosis: The output from a backup shows that GHI failed to back up a file's data from a snapshot.

Resolution: Check the migration problems above to determine why the file failed to migrate. Refer to the limits in Starting GHI for the first time to verify they were not exceeded.

18.2.40. Too many open files from image backup

Diagnosis: Too many open files from an image backup.

Resolution: The **ulimit** settings need to be set on GHI nodes for image backups. Refer to the Set ulimits section for how to set these limits.

18.2.41. Failed to backup namespace information

Diagnosis: The output from a backup shows that GHI failed to back up a namespace file. One possible reason is there was a failure transferring the file to HPSS.

Resolution: Refer to IOM problems above.

Appendix A: Glossary of terms and acronyms

AIX	Advanced Interactive Executive. An operating system provided on many IBM machines.
Alarm	A log record message type used to log high-level error conditions.
ANSI	American National Standards Institute.
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.
Debug	A log record message type used to log lower-level error conditions.
Directory	An HPSS object that can contain files, symbolic links, hard links, and other directories.

Dismount	An operation in which a cartridge is either physically or logically removed from a device, rendering it unreadable and non-writable. In the case of tape cartridges, a dismount operation is a physical operation. In the case of a fixed disk unit, a dismount is a logical operation.
DMAPI	Data Management Application Programming Interface.
DNS	Domain Name Service.
DOE	Department of Energy.
Drive	A physical piece of hardware capable of reading and/or writing mounted cartridges. The terms device and drive are often used interchangeably.
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.
File family	An attribute of an HPSS file that is used to group a set of files on a common set of tape virtual volumes.
Fileset	A collection of related files that are organized into a single easily managed unit. A fileset is a disjoint directory tree that can be mounted in some other directory tree to make it accessible to users.
Fileset ID	A 64-bit number that uniquely identifies a fileset.
Fileset name	A name that uniquely identifies a fileset.
FSID	File system unique identifier.
GB	Gigabyte (2^{30}).
GHI	Spectrum Scale/HPSS Interface.
Hierarchy	See "Storage Hierarchy".
HPSS	High Performance Storage System.
HSI	Hierarchical Storage Interface.
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.
Junction	A mount point for an HPSS fileset.
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.
Name Service	The portion of the Core Server that provides a mapping between names and machine-oriented identifiers. In addition, the name service performs access verification and provides the Portable Operating System Interface (POSIX).
Namespace	The set of name-object pairs managed by the HPSS Core Server.
NSL	National Storage Laboratory.

PB	Petabyte (2^{50}).
PM	Process Manager.
POSIX	Portable Operating System Interface (for computer environments).
RPC	Remote Procedure Call.
SD	Scheduler Daemon.
SNL	Sandia National Laboratories.
Spectrum Scale	New name of General Parallel File System (GPFS).
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.
Transaction	A programming construct that enables multiple data operations to possess the following properties: • All operations commit or abort/roll-back together such that they form a single unit of work. • All data modified as part of the same transaction are guaranteed to maintain a consistent state whether the transaction is aborted or committed. • Data modified from one transaction are isolated from other transactions until the transaction is either committed or aborted. • Once the transaction commits, all changes to data are guaranteed to be permanent.

Appendix B: References

1. ***HPSS Admin Guide***
2. ***HPSS Error Manual***
3. ***HPSS User's Guide***
4. ***Spectrum Scale Data Management API Guide***
5. ***Spectrum Scale Administration and Programming Reference***
6. ***Spectrum Scale Advanced Administration***
7. ***POSIX 1003.1-1990 Tar Standard***

Appendix C: Man Pages

The following appendix contains man pages for GHI tools. These can be viewed on systems where GHI has been installed.

C.1. dmapishell(7)

C.1.1. Name

dmapishell - Monitor events in a GPFS file system.

C.1.2. Syntax

dmapishell

C.1.3. Description

GPFS supports DMAPI (Data Management Application Programming Interface). DMAPI allows the user to monitor events associated with a GPFS file system or with an individual file. DMAPI also can manage and maintain file system data.

Data Management attributes (DM attrs) are associated with individual files. There are opaque and non-opaque attributes. Opaque attributes are not interpreted by DMAPI. Non-opaque attributes, such as managed regions and event lists, are available to DMAPI.

Using this tool, DM Extended Attrs can be set or displayed for a given file, all DM Attrs can be set or displayed, DM stat info can be displayed, and sessions can be displayed or deleted.

C.1.4. Parameters

cleanup
: Deletes all but the current session

sessions
: Displays all active sessions

names
: Displays all session names

dmstat *file*
: Displays select DM stat info for *file*

lsattr *file*
: Displays all DM attrs for *file*

getxattr *file*
: Displays extended attrs for *file*

setxattr *file*
: Sets extended attrs for *file*

setuda *file idx*
: Sets extended attrs for *file*

getallattr *file*

Displays extended attrs for *file*

setallattr *file*

Sets extended attrs for *file*

igetattrs *file mtpt*

calls gpfs_igetattrs

igetattrs *file mtpt*

calls gpfs_igetattrs

boomdmattr *fs file*

zap *file* GPFS MIDX

fixdmattr *fs file*

recreate *file* GPFS MIDX with HPSS UDA

snapattr *file*

Displays all sessions

exit, stop, quit

Exits dmapishell and deletes the session

C.1.5. Examples

```
dmapishell
> sessions
> Session Name: "dmapishell" (ID:1199476491)
Number of Tokens = (0)
Found 80 bytes of disposition information
Dispositions and Event Lists for file system 13882589226889280592:
EVENT | DISPOSITIONED |      ENABLED
-------------------------------------------------
DM_EVENT_PREUNMOUNT |            NO |      YES
DM_EVENT_UNMOUNT |             NO |      YES
DM_EVENT_NOSPACE |             NO |      YES
```

C.2. ghiaddfs(7)

C.2.1. Name

ghiaddfs - Add a file system to the GHI configuration

C.2.2. Syntax

ghiaddfs [-k|KTv] *file_system* [-c "*# comment*"] [-r *read_only_target*] *mount_point* [*SD_port ED_port*]

C.2.3. Description

Use the ghiaddfs command to add a new file system to the GHI configuration. This command adds a file system with the default configuration. The ghichfs command can be used to alter the configuration after it has been added.

C.2.4. Options

-k

If GHI backup-related DB2 tables already exist for the new file system, use them as-is.

-K

If GHI backup-related DB2 tables already exist for the new file system, re-create them.

-T

Test only — do all processing steps required but do not alter any files or DB2 tables, or restart any processes.

-v

Verbose.

C.2.5. Parameters

file_system

Name of the GPFS file system to add to the GHI configuration. Must match the GPFS configuration, and be unique among all GHI clusters which connect to the same HPSS.

-c "# comment"

Add a user comment.

-r *read_only_target*:; Create <file_system> as a read-only version of file system <read_only_target>, which must already be configured and not itself be a read-only FS. <read_only_target> may reside either on the same or a separate cluster as <file_system>.

mount_point

The mount point for the GPFS file system. Must match the GPFS configuration, and be unique among all GHI clusters which connect to the same HPSS.

SD_port

Port that the ghi_sd should use for this file system. GHI will assign one if one is not specified.

ED_port

Port that the ghi_ed should use for this file system. GHI will assign one if one is not specified.

C.2.6. Notes

Options *-k* and *-K* are mutually exclusive.

If neither option is specified, it is an error if DB2 tables exist. Else, they will be created.

If either option is specified, it is not an error if DB2 tables do not exist: they will be created. If tables do exist, they will be handled per the option as explained above.

To determine the default configuration used by ghiaddfs, run the ghilsfsdefaults command. Run ghichfsdefaults if it needs to be modified.

C.3. ghiaddiom(7)

C.3.1. Name

ghiaddiom - Add a new IOM to the GHI IO Managers configuration

C.3.2. Syntax

ghiaddiom [-RTv] *file_system* [-c "# *comment*"] *IOM_Node*[:_Port_] *Active_On_Session_Node Est_XFER_Rate Chunksize*

C.3.3. Description

Use the ghiaddiom command to add an IOM to the GHI IOM configuration.

C.3.4. Parameters

-R

Push configuration changes to all GHI nodes and restart the GHI SD. (If this option is omitted, changes are only posted to the GHI session manager and the IOM node, and the SD is not restarted. The new IOM will not be used until the SD is restarted.)

-T

Test only — do all processing steps required but do not alter any files, or restart any processes.

-v

Verbose.

file_system

The GHI file system the IOM will be associated with.

-c "# *comment*"

Add a user comment.

***IOM_Node*[:_Port_]**

The GHI node on which the IOM will execute. If the port is not specified, GHI will select one.

Active_On_Session_node

If the node becomes the GHI session node...should the IOM stay active? Acceptable values are TRUE or FALSE.

Est_XFER_Rate

The estimated transfer rate (per second) GHI should use when calculating load balancing before actual transfer rates can be determined. May append *KB*, *MB*, or *GB* as a units multiplier.

Chunksize

Maximum number of bytes to transfer per non-aggregate HPSS I/O request. Append *KB*, *MB*, *GB*, or *TB* as a units multiplier.

C.3.5. Notes

Both *file_system* and *IOM_Node* must be configured in GHI prior to executing this command.

If only a single IOM is to be added, use the *-R* option unless some reason exists why the change should not immediately be pushed to all GHI nodes and the SD restarted.

To speed up the addition of multiple IOMs for an FS, do not use the *-R* option until adding the final IOM.

C.4. ghiaddnode(7)

C.4.1. Name

ghiaddnode - Add a node to the GHI configuration

C.4.2. Syntax

ghiaddnode [-Tv] *node*

C.4.3. Description

Use the ghiaddnode command to add a node to the GHI configuration. The GHI configuration and appropriate binaries will be copied to the node. After a node is added, GHI client applications (ghi_ls or ghi_stage) will be available on the node. Also, the node will be available for utilization during ghiapplypolicy runs.

C.4.4. Parameters

-T

Test only — do all processing steps required but do not alter any files, or restart any processes.

-v

Verbose.

node

Node to be added.

C.4.5. Notes

The node must be known to GPFS and an Admin node as displayed by the GPFS *mmlscluster* command..

C.5. ghi_admin(7)

C.5.1. Name

ghi_admin - An interactive GHI admin tool

C.5.2. Syntax

ghi_admin

C.5.3. Description

ghi_admin is a GHI utility which facilitates locating, reprioritizing, and removing file transfer requests within GHI. It also allows users to clear any links to HPSS which may be associated with a file within GPFS, read in data from HPSS (that's associated with a GPFS file), as well as to cancel an ongoing GHI backup and to cause GHI to reinitialize itself.

```
GHI_ADMIN Action Menu
-----------------------------------------------------
1) Set the filename (currently set to: not set)
2) Lookup file's transfer request within GHI
3) Cancel transfer request for file
4) Move file's transfer request to top of Scheduler's queue
5) Reset a file's xattrs/regions
6) Read a file from HPSS
7) Stop an in-progress backup
8) Reinitialize GHI
9) Get status of transfer requests for files in filelist
10) Cancel transfer requests for files in filelist
11) Reset xattrs/regions for files in filelist
H) Help
E) Exit
-----------------------------------------------------
```

C.5.4. Actions

This section describes the actions available.

Action (E)

causes ghi_admin to terminate.

Action (H)

displays this document.

Action (1)

Set the filename.

Action (1) is used to select the file to which actions (2) thru (5) will apply. The complete pathname

must be supplied and it must be a file which resides on a GPFS file system. The entered filename will be validated to reside on a GPFS file system, after which its mount point, inode, and igen will be displayed for verification.

When the menu choice for action (1) shows the filename as "not set", action (1) can be skipped and it will implicitly be called when one of actions (2) thru (5) is selected. Otherwise, action (1) will display the currently set filename and a new filename must be entered to change the filename to be applied to actions (2) thru (5).

Example:

```
Enter your option > 1
Enter the filename > /ghi3/IT/root/FUNC_02/func_02_file_29
Mount point = /ghi3 Inode = 0.331336 Igen = 65545
```

Action (2)

Lookup file's transfer request within GHI

Select action (2) to locate a transfer request for the currently set filename. The transfer request may be either pending or in-progress.

Example:

```
Enter your option > 2
/ghi3/IT/root/FUNC_02/func_02_file_29 in queue
```

Action (3)

Cancel transfer request for file

Select action (3) to cancel a transfer request. The request may still be pending in the queue or the data transfer may already have been initiated for it. As evidenced by the possibility of "(marked for deletion)" appearing within responses to action (2) and by some of the responses to this action, a request may remain in the system for an extended period of time before it is actually removed. When a request is cancelled, the status code written to the associated exception file that is displayed upon completion of the policy run will be -7003.

Example:

```
Enter your option > 3
Cancel request for /ghi3/IT/root/FUNC_02/func_02_file_29 sent to
IOM napa
```

Action (4)

Move file's transfer request to top of Scheduler's queue

Select action (4) to raise the priority of a pending transfer request. Once a request has been selected

for processing and removed from the schedule queue, its priority cannot be altered.

Example:

```
Enter your option > 3
Moved /ghi3/IT/root/FUNC_02/func_02_file_29 to top of schedule queue
```

Action (5)

Reset a file's xattrs/regions

Select action (5) to make a file appear as if it has not been copied into HPSS. This action should be used with caution because it is possible to leave the file in a state such that its contents are lost. If it already appears that the file has not been copied into HPSS, ghi_admin will abort the action with the following message:

File not in HPSS, nothing to reset...

If it is determined that the file's contents have been purged from GPFS, ghi_admin will display the following prompt:

File has been purged from GPFS, stage file first (Y/N)?

If the user's response is Y, the file's contents will be staged back from HPSS. If the response is N, then this prompt will be displayed:

All file data will be lost...continue with reset (Y/N)?

Responding Y to this prompt will result in loss of the link to the file's data within HPSS, and it will henceforth be shown as having no associated data within HPSS. There will exist no way to restore the purged data from HPSS. The file's length will also be set to zero to indicate that it contains no data. Enter N to abort the reset or Y to continue.

If the file's contents are in GPFS (after possibly having just been staged back from HPSS), the following prompt will be displayed:

The file's HPSS data will be lost...continue with reset (Y/N)?

If the reset is allowed to proceed, the link to the file's data within HPSS will be lost and it will henceforth be shown as having no associated data within HPSS. Enter N to abort the reset or Y to continue.

Upon completion of the reset, a "ghi_ls -l" will be issued for the file for verification.

Example:

```
Enter your option > 5
Enter the filename > /ghi2/IT/FUNC_01/file_3
Mount point = /gpfs/ghi2 Inode = 0.8671709 Igen = 65539
File has been purged from GPFS, stage file first (Y/N)? y
Staging...
```

```
The file's HPSS data will be lost...continue with reset (Y/N)? y
G -rw-r--r-- 1 root system 512000 Nov 03 16:02 /ghi2/IT/FUNC_01/file_3
```

If a processing error causes the action to abort, one or more messages will be displayed to indicate the nature of the error.

Action (6)

Read a file from HPSS

Select action (6) to retrieve a copy of a GPFS file's HPSS data and store it in a separate file. The file receiving the HPSS data will not be supplied with any link to the HPSS data. This action is useful to verify the contents of a file's HPSS data without overwriting its contents within GPFS. Two prompts will be displayed:

Enter the GPFS pathname > Enter the destination pathname >

For the first prompt (GPFS pathname), enter the complete pathname of the GPFS file for which the HPSS data is desired. For the second prompt (destination pathname), enter the complete pathname of the file which is to receive the data. The file need not reside on a GPFS file system, it may be anywhere under "/", and the required directory structure will be created if necessary. Upon successful completion of retrieving the HPSS data, a "ghi_ls -l" command will be executed for both the GPFS and destination pathnames as a quick verification that the HPSS data was retrieved.

Example:

```
Enter your option > 6
Enter the GPFS pathname > /ghi2/IT/FUNC_01/file_4
Enter the destination pathname > /xxx
H -rw-r--r-- 1 root system 512000 Nov 03 16:02 /ghi2/IT/FUNC_01/file_4
- -r--r-x--- 1 root system 512000 Nov 04 16:02 /xxx
```

If a processing error causes the action to abort, one or more messages will be displayed to indicate the nature of the error.

Action (7)

Stop an in-progress backup

Select action (7) to abort a currently running GHI backup. The following prompt will be displayed:

Enter the mountpoint >

If an entry for the specified mountpoint can be located within configuration file "/var/hpss/ghi/etc/ghi.conf", the file system field will be extracted from that entry and displayed for verification. Next, all nodes listed in configuration file "/var/hpss/ghi/etc/ghinode.conf" will be queried for a GHI backup process for the displayed file system, and if found, a SIGINT signal will be sent to it. The command used to send the SIGINT, which will include the PID and the hostname if the process is found on a node other than on which ghi_admin is running, will be displayed for verification.

Example:

```
Enter your option > 7
Enter the mountpoint > /ghi3
Filesystem is (gpfs3fs)
Looking for backup process...
ssh miami kill -2 14865 # ghi_backup
```

If no GHI backup process can be located anywhere within the cluster, the text "No process found". will replace the line containing the "kill" command.

If a processing error causes the action to abort, one or more messages will be displayed to indicate the nature of the error.

Action (8)

Reinitialize GHI

Select action (8) to cause the GHI daemon processes to reinitialize themselves. The processes which may be targeted are "ghi_pm", "ghi_md", "ghi_sd", "ghi_ed", and "ghi_iom" — all to include the complete pathname to the GHI bin directory, e.g., "/opt/ghi/bin/ghi_pm".

Either all GHI processes on all nodes in the cluster may be targeted, or just the ED (ghi_ed), SD (ghi_sd), and IOM (ghi_iom) that are associated with a specific file system.

The following prompt will be displayed:

Enter the FS name, or *all* to re-init all GHI processes >

If *all* is selected, the following prompt will be displayed for verification:

Re-init all GHI processes...are you sure (Y/N)?

Respond Y to continue with reinitialization. All nodes listed in configuration file "/var/hpss/ghi/etc/ghinode.conf" will be queried for appropriate processes and a SIGHUP will be sent to each as it is found.

Example:

```
Enter your option > 8
Enter the FS name, or 'all' to re-init all GHI processes > all
Re-init all GHI processes...are you sure (Y/N)? y
Re-init'ing GHI via the following commands...
ssh miami kill -1 3656 # /opt/ghi/bin/ghi_iom gpfs3fs 8032
ssh miami kill -1 3657 # /opt/ghi/bin/ghi_iom gpfs4fs 8042
ssh miami kill -1 6127 # /opt/ghi/bin/ghi_iom gpfs5fs 8052
ssh miami kill -1 14280 # /opt/ghi/bin/ghi_pm
ssh miami kill -1 14834 # /opt/ghi/bin/ghi_md
ssh miami kill -1 15014 # /opt/ghi/bin/ghi_ed gpfs3fs
ssh miami kill -1 15109 # /opt/ghi/bin/ghi_sd gpfs3fs
ssh miami kill -1 15273 # /opt/ghi/bin/ghi_ed gpfs4fs
```

```
ssh miami kill -1 15274 # /opt/ghi/bin/ghi_sd gpfs4fs
ssh miami kill -1 15516 # /opt/ghi/bin/ghi_sd gpfs5fs
ssh miami kill -1 15981 # /opt/ghi/bin/ghi_ed gpfs5fs
kill -1 3699 # /opt/ghi/bin/ghi_iom gpfs3fs 8032
kill -1 3700 # /opt/ghi/bin/ghi_iom gpfs4fs 8042
kill -1 3701 # /opt/ghi/bin/ghi_iom gpfs5fs 8052
ssh worcester kill -1 3996 # /opt/ghi/bin/ghi_iom gpfs4fs 8042
ssh worcester kill -1 4703 # /opt/ghi/bin/ghi_iom gpfs3fs 8032
ssh worcester kill -1 4705 # /opt/ghi/bin/ghi_iom gpfs5fs 8052
```

Example:

```
Enter your option > 8
Enter the FS name, or 'all' to re-init all GHI processes > gpfs3fs
Re-init'ing gpfs3fs via the following commands...
ssh miami kill -1 3656 # /opt/ghi/bin/ghi_iom gpfs3fs 8032
ssh miami kill -1 15014 # /opt/ghi/bin/ghi_ed gpfs3fs
ssh miami kill -1 15109 # /opt/ghi/bin/ghi_sd gpfs3fs
kill -1 3699 # /opt/ghi/bin/ghi_iom gpfs3fs 8032
ssh worcester kill -1 4703 # /opt/ghi/bin/ghi_iom gpfs3fs 8032
```

Action (9)

Get status of transfer requests for files in filelist

Select action (9) to display the status of each transfer request specified in a filelist — a file containing a list of filenames as either being the output of a GPFS policy run or a listing of complete pathnames, one per line. If from a policy run, the inode and igen will be retrieved from the filelist; else, they will be gotten by making a query to GPFS.

Example:

```
Enter your option > 9
Enter the filename of the filelist file > mmPolicy.ix.19718.3969A2EA.1
Filelist is from a GPFS policy run...
/ghi3/IT/root/FUNC_01/func_01_file_50 -- Mount point = /ghi3 Inode = 0.203534 Igen =
65549
/ghi3/IT/root/FUNC_01/func_01_file_50 not found
/ghi3/IT/root/FUNC_01/func_01_file_53 -- Mount point = /ghi3 Inode = 0.203535 Igen =
65552
/ghi3/IT/root/FUNC_01/func_01_file_53  being processed by IOM cranston
/ghi3/IT/root/FUNC_01/func_01_file_62 -- Mount point = /ghi3 Inode = 0.203537 Igen =
65552
/ghi3/IT/root/FUNC_01/func_01_file_62  at top-of-queue
/ghi3/IT/root/FUNC_01/func_01_file_20 -- Mount point = /ghi3 Inode = 0.203538 Igen =
65552
/ghi3/IT/root/FUNC_01/func_01_file_20  in queue
/ghi3/IT/root/FUNC_01/func_01_file_24 -- Mount point = /ghi3 Inode = 0.203538 Igen =
65552
```

```
/ghi3/IT/root/FUNC_01/func_01_file_24  being processed by IOM cancelled_job (marked
for deletion)
```

Action (10)

Cancel transfer requests for files in filelist

Select action (10) to cancel all transfer requests specified in a filelist — a file containing a list of filenames as either being the output of a GPFS policy run or a listing of complete pathnames, one per line. If from a policy run, the inode and igen will be retrieved from the filelist; else, they will be gotten by making a query to GPFS.

Example:

```
Enter your option > 10
Enter the filename of the filelist file > mmPolicy.ix.20921.ED43A22B.1
Filelist is from a GPFS policy run...
/ghi3/IT/root/FUNC_01/func_01_file_50 -- Mount point = /ghi3 Inode = 0.203534 Igen =
65549
Cancel request for /ghi3/IT/root/FUNC_01/func_01_file_50 sent to IOM cranston
/ghi3/IT/root/FUNC_01/func_01_file_62 -- Mount point = /ghi3 Inode = 0.203537 Igen =
65552
/ghi3/IT/root/FUNC_01/func_01_file_62 not found -- cannot delete
/ghi3/IT/root/FUNC_01/func_01_file_51 -- Mount point = /ghi3 Inode = 0.203812 Igen =
65543
Deleted /ghi3/IT/root/FUNC_01/func_01_file_51 awaiting assignment to an IOM
/ghi3/IT/root/FUNC_01/func_01_file_88 -- Mount point = /ghi3 Inode = 0.204512 Igen =
65541
Deleted /ghi3/IT/root/FUNC_01/func_01_file_88 from the schedule queue
```

Action (11)

Reset xattrs/regions for files in filelist

Select action (11) to make all files in a list of files appear as if they have not been copied into HPSS. This action should be used with caution because it is possible to leave the files in a state such that their contents are lost. GHI will process each file individually If the file being processed already appears that it has not been copied into HPSS, ghi_admin will abort the action with the following message:

File not in HPSS, nothing to reset…

If it is determined that the file's contents have been purged from GPFS, ghi_admin will display the following prompt:

File has been purged from GPFS, stage file first (Y/N)?

If the user's response is Y, the file's contents will be staged back from HPSS. If the response is N, then this prompt will be displayed:

All file data will be lost…continue with reset (Y/N)?

Responding Y to this prompt will result in loss of the link to the file's data within HPSS, and it will henceforth be shown as having no associated data within HPSS. There will be no way to restore the purged data from HPSS. The file's length will also be set to zero to indicate that it contains no data. Enter N to abort the reset or Y to continue.

If the file's contents are in GPFS (after possibly having just been staged back from HPSS), the following prompt will be displayed:

The file's HPSS data will be lost…continue with reset (Y/N)?

If the reset is allowed to proceed, the link to the files' data within HPSS will be lost and they will henceforth be shown as having no associated data within HPSS. Enter N to abort the reset or Y to continue.

Upon completion of the reset, a "ghi_ls -l" should be issued for the files for verification.

Example:

```
Enter your option > 11
Enter the filename of the filelist file > /marsfs1/scratch/.ghi/list.list
Stage any purged files before reset? (y/n) y
Filelist is from a GPFS policy run...
/marsfs1/admin/file.9 -- Mount point = /marsfs1 Inode = 16451 Igen = 1932359307
/marsfs1/admin/file.1 -- Mount point = /marsfs1 Inode = 16452 Igen = 1424912955
/marsfs1/admin/file.2 -- Mount point = /marsfs1 Inode = 16453 Igen = 2137630364
/marsfs1/admin/file.3 -- Mount point = /marsfs1 Inode = 16454 Igen = 2112163256
```

C.6. ghiapplypolicy(7)

C.6.1. Name

ghiapplypolicy - Deletes files or migrates file data between storage pools in accordance with policy rules.

C.6.2. Syntax

ghiapplypolicy {*Device*|*Directory*} [-P *PolicyFile*] [-I {*yes*|*defer*|*test*}] [-L *n*] [-D *yyyy-mm-dd[@hh:mm[:ss]]*] [-M *name=value*] [-N {*Node[,Node...]* | *NodeFile* | *NodeClass*}] [-b] [-r] [-s *MountPoint*]

C.6.3. Description

Use the ghiapplypolicy command to manage the placement and replication of data within GPFS storage pools. It can also be used to delete files from GPFS. You may issue the ghiapplypolicy command from any node in the GPFS cluster that has the file system mounted.

Any given file is a potential candidate for at most one MIGRATE or DELETE operation during one invocation of the ghiapplypolicy command.

A file that matches an EXCLUDE rule is not subject to any subsequent MIGRATE or DELETE rules. You should carefully consider the order of rules within a policy to avoid unintended consequences.

C.6.4. Parameters

For options other than [-b] [-r] and [-s], please see documentation for mmapplypolicy

-b

Tells ghiapplypolicy to use the -I defer option and allow ghi_migrate to filter the aggregate candidates and non-aggregate files through ghi_backup_helper

-r

Tells ghiapplypolicy to use the -I defer option, and allow ghi_recall to bulk files based on aggrs/non-aggrs/tape-position

-s *MountPoint*

Scratch flag that sets the scratch area to <MountPoint>/scratch/.ghi

C.7. ghi_backup(7)

C.7.1. Name

ghi_backup - Backup a GHI-managed file system

C.7.2. Syntax

ghi_backup <*filesys*> <*type*> [--keep-snapshot] [--debug] [--verbose] [[-E *Fsets* | -E -F *List of files*] [-U *mmapplypolicy_user_args*] [-I *mmimgbackup_user_args*]]

C.7.3. Description

Use ghi_backup command to take a backup of a GHI-managed file system.

This will automatically load the mapping table for the backup in the background.

C.7.4. Parameters

--keep-snapshot
: Keep the Spectrum Scale snapshot that is taken during a backup

--verbose
: Display command output

--debug
: Leave backup temporary files in scratch directory for debugging purposes (implies verbose)

-E *Fsets*
: Filesets which need not be linked-in, either space-separated list of filesets

-F *List of Fsets*
: Space-separated list of files containing list of filesets

-U *user_args*
: Additional args to be passed to mmapplypolicy

-I *mmimgbackup_user_args*
: Additional args to be passed to mmimgbackup

filesys
: Name of the GHI-managed file system that needs to be backed up.

type
: This is the type of backup to perform. The available options are:

1. **image** - Image backup using GPFS sobar capability

C.8. ghi_backup_manager(7)

C.8.1. Name

ghi_backup_manager - Backup a GHI-managed file system

C.8.2. Synopsis

ghi_backup_manager <*file system*> [-d <*delete_thread_count*>]

C.8.3. Description

Use ghi_backup_manager to delete backups for the given file system. The tool is an interactive menu that prompts the user to select which backup index they would like to delete.

The tool will automatically unload the mapping table.

C.8.4. Parameters

- <*file system*> - GHI-managed file system to delete associated backups
- -d <*delete_thread_count*> - Number of threads to use for deleting HPSS files (default=20)

C.8.5. Example

```
[root@rock ghi_backup_manager]# ghi_backup_manager ghi-rockfs1
***********************************
* GHI Backup Manager Main Menu     *
***********************************
Menu options are
1. Select a backup for deletion.
2. Resume deletion of a backup
Type Quit to exit
> 1

************************************************
* List of Available Backups
************************************************

Selection Index Type        Status      Timestamp
1         227   Image       Completed   Mon Mar 21 12:31:51 2016
2         230   Image       Completed   Mon Mar 21 13:04:51 2016
3         234   Image       Completed   Mon Mar 28 10:15:23 2016
4         235   Image       Completed   Mon Mar 28 15:33:50 2016
5         236   Image       Completed   Mon Mar 28 15:35:54 2016
6         239   Image       Completed   Mon Apr 18 11:29:12 2016

************************************************
* Input the selection to list all the associated backups.
* Type List to display the entries again.
```

```
* Type Back to return to the main menu.
* Type Quit to exit.
************************************************
> 1

************************************************
* List of all backups associated with selection
* Dated Mon Mar 21 12:31:51 2016
************************************************

Selection Index Type        Status      Timestamp
1         227   Image       Completed   Mon Mar 21 12:31:51 2016

************************************************
* Input the selection to choose a backup to delete.
* Type List to return to display the entries again.
* Type Back to return to the main menu.
* Type Quit to exit.
************************************************
> 1
************************************************
* Backup to delete.
************************************************

Backup Index 227         Timestamp Mon Mar 21 12:31:51 2016

************************************************
* Please check the above information and
* confirm that it is correct!  Deleting the backup
* will remove all associated files from HPSS.
*
* Type Delete to start the delete process
* Type Back to return to the main menu
* Type Quit to exit
************************************************
> Delete
Starting Deletion
Retrieving entries to unlink from HPSS.
************************************************
* Finished unlinking unreferenced files from HPSS.
* Beginning unlinking backup metadata files.
************************************************

************************************************
* Successfully deleted the backup and unlinked all
* unreferenced files from HPSS
************************************************
```

C.9. ghichfs(7)

C.9.1. Name

ghichfs - Update a GHI file system configuration

C.9.2. Syntax

ghichfs [-Tv] <*file_system*> [-c "# <*comment*>"] [--junct <*HPSS_junction*>] [--basep <*HPSS_path*>] [--bupath <*HPSS_path*>] [--maxagg <*number*>] [--minagg <*number*>] [--aggcos <*number*>] [--aggtps <*number*>] [--bbc <*number*>] [--bucos <*number*>] [--pblock <*number*>] [--perf <*perf_logging*>] [--com *TRUE*|*FALSE*] [--poiot *TRUE*|*FALSE*] [--sod *ON*|*OFF*] [--dse <*number*>] [--edmaxc <*number*>] [--edtps <*number*>] [--edrqs <*number*>] [--edport <*number*>] [--iomaxc <*IOM Max Connections*>] [--iotps <*number*>] [--iorqs <*number*>] [--iomon *ON*|*OFF*] [--iomonf <*number*>] [--iomonp <*GPFS_path*>] [--simaxc <*number*>] [--sitps <*number*>] [--sirqs <*number*>] [--sidm <*number*>] [--sibu <*number*>] [--simg <*number*>] [--sirc <*number*>] [--sirs <*number*>] [--siad <*number*>] [--scmaxc <*number*>] [--sctps <*number*>] [--scrqs <*number*>] [--sdport <*number*>] [--sdmon *ON*|*OFF*] [--sdmonf <*number*>] [--sdmonp <*GPFS_path*>]

C.9.3. Description

Use the ghichfs command to update a file system in the GHI configuration. Some updates will require the GHI servers to be restarted so they can take effect.

C.9.4. Options

-T

Test only. Do all processing steps required but do not alter any files, or restart any processes.

-v

Verbose

C.9.5. Parameters

<*file_system*>

Name of the GHI file system which configuration is to be modified.

-c "# comment"

Add or change a user comment to be associated with the FS.

--junct <*HPSS_junction*>

The HPSS junction under which all system and user data will reside.

--basep <*HPSS_path*>

The path in HPSS under which all user data will reside.

--bupath <*HPSS_path*>

The path in HPSS under which all system data will reside.

--maxagg <*number*>

Maximum number of GPFS files which GHI will place into a single aggregated HPSS file.

--minagg <*number*>

Minimum number of GPFS files which GHI will place into a single aggregated HPSS file.

--aggcos <*number*>

HPSS Class-Of-Service into which the associated index file for aggregated files will be placed.

--aggtps <*number*>

Aggr Thread Pool Size.

--bbc <*number*>

Maximum number of GPFS files which GHI will place into a single GHI backup file.

--bucos <*number*>

HPSS Class-Of-Service into which the files used to store GHI backups will be placed.

--pblock <*number*>

Number of GPFS data blocks to be left GPFS-resident when a file is purged.

--perf <*perf_logging*>

Performance logging, <*perf_logging*> is one or more of the following: ED ILM SD IOM HTAR PIO.

--com TRUE|FALSE

Checksum on Migrate.

--poiot TRUE|FALSE

Purge Only If On Tape.

--sod ON|OFF

Stage On Demand.

--dse <*number*>

DMAPI Stage Errno.

--edmaxc <*number*>

ED Max Connections.

--edtps <*number*>

ED Thread Pool Size.

--edrqs <*number*>

ED Request Queue Size.

--edport <*number*>

Port to be used by the ED for communicating with its associated SD.

--iomaxc <*number*>

IOM Max Connections.

--iotps <*number*>

IOM Thread Pool Size.

--iorqs <*number*>

IOM Request Queue Size.

--iomon ON|OFF

Enable/disable output to the IOM monitor file.

--iomonf <*number*>

Interval, in seconds, at which the SD is to write to the IOM monitor file.

--iomonp <*GPFS_path*>

Pathname of the IOM monitor file.

--simaxc <*number*>

SD IOM Max Connections.

--sitps <*number*>

SD IOM Thread Pool Size.

--sirqs <*number*>

SD IOM Request Queue Size.

--sidm <*number*>

Share of SD IOM threads for processing DMAPI requests in a round-robin schedule. Acceptable values are integers greater than or equal to zero. If the thread share is zero, SD will not schedule any DMAPI requests.

--sibu <*number*>

Share of SD IOM threads for processing Backup requests in a round-robin schedule. Acceptable values are integers greater than or equal to zero. If the thread share is zero, SD will not schedule any Backup requests.

--simg <*number*>

Share of SD IOM threads for processing Migrate requests in a round-robin schedule. Acceptable values are integers greater than or equal to zero. If the thread share is zero, SD will not schedule any Migrate requests.

--sirc <*number*>

Share of SD IOM threads for processing Recall requests in a round-robin schedule. Acceptable values are integers greater than or equal to zero. If the thread share is zero, SD will not schedule any Recall requests.

--sirs <*number*>

Share of SD IOM threads for processing Restore requests in a round-robin schedule. Acceptable values are integers greater than or equal to zero. If the thread share is zero, SD will not schedule any Restore requests.

--siad <*number*>

Share of SD IOM threads for processing Admin requests in a round-robin schedule. Acceptable values are integers greater than or equal to zero. If the thread share is zero, SD will not schedule any Admin requests.

--scmaxc <*number*>

SD Client Max Connections.

--sctps <*number*>

SD Client Thread Pool Size.

--scrqs <*number*>

SD Client Request Queue Size.

--sdport <*number*>

Port to be used in communicating with the SD to request and monitor file transfers.

--sdmon ON|OFF

Enable/disable output to the SD monitor file.

--sdmonf <*number*>

Interval, in seconds, at which the SD is to write to the SD monitor file.

--sdmonp <*GPFS_path*>

Pathname of the SD monitor file.

C.9.6. Notes

This command cannot be used to change whether an FS is full-access or read-only, or to alter the associated FS. To accomplish this, run ghidelfs and ghiaddfs.

C.10. ghichfsdefaults(7)

C.10.1. Name

ghichfsdefaults - Update the GHI default file system configuration

C.10.2. Syntax

ghichfsdefaults [-Tv] [-c "# <*comment*>"] [--junct <*HPSS_junction*>] [--basep <*HPSS_path*>] [--bupath <*HPSS_path*>] [--maxagg <*number*>] [--minagg <*number*>] [--aggcos <*number*>] [--aggtps <*number*>] [--bbc <*number*>] [--bucos <*number*>] [--pblock <*number*>] [--perf <*perf_logging*>] [--com *TRUE*|*FALSE*] [--poiot *TRUE*|*FALSE*] [--sod *ON*|*OFF*] [--dse <*number*>] [--edmaxc <*number*>] [--edtps <*number*>] [--edrqs <*number*>] [--edport <*number*>] [--iomaxc <*IOM Max Connections*>] [--iotps <*number*>] [--iorqs <*number*>] [--iomon *ON*|*OFF*] [--iomonf <*number*>] [--iomonp <*GPFS_path*>] [--simaxc <*number*>] [--sitps <*number*>] [--sirqs <*number*>] [--scmaxc <*number*>] [--sctps <*number*>] [--scrqs <*number*>] [--sdport <*number*>] [--sdmon *ON*|*OFF*] [--sdmonf <*number*>] [--sdmonp <*GPFS_path*>]

C.10.3. Description

Use the ghichfsdefaults command to update the default file system in the GHI configuration. This is the configuration that any subsequently added file systems will have. Also, ghilsfs will display a * next to a value as it displays the configuration for an FS when that value differs from the default configuration value, regardless of when the FS was added to GHI or when the value was last changed.

C.10.4. Options

-T

Test only — do all processing steps required but do not alter any files, or restart any processes.

-v

Verbose.

C.10.5. Parameters

-c "# comment"

Add or change a user comment to be associated with the FS.

--junct <*HPSS_junction*>

The HPSS junction under which all system and user data will reside.

--basep <*HPSS_path*>

The path in HPSS under which all user data will reside.

--bupath <*HPSS_path*>

The path in HPSS under which all system data will reside.

--maxagg <*number*>

Maximum number of GPFS files which GHI will place into a single aggregated HPSS file.

--minagg <*number*>

Minimum number of GPFS files which GHI will place into a single aggregated HPSS file.

--aggcos <*number*>

HPSS Class-Of-Service into which the associated index file for aggregated files will be placed.

--aggtps <*number*>

Aggr Thread Pool Size.

--bbc <*number*>

Maximum number of GPFS files which GHI will place into a single GHI backup file.

--bucos <*number*>

HPSS Class-Of-Service into which the files used to store GHI backups will be placed.

--pblock <*number*>

Number of GPFS data blocks to be left GPFS-resident when a file is purged.

--perf <*perf_logging*>

Performance logging, <*perf_logging*> is one or more of the following: ED ILM SD IOM HTAR PIO.

--poiot TRUE|FALSE

Purge Only If On Tape.

--sod ON|OFF

Stage On Demand.

--dse <*number*>

DMAPI Stage Errno.

--edmaxc <*number*>

ED Max Connections.

--edtps <*number*>

ED Thread Pool Size.

--edrqs <*number*>

ED Request Queue Size.

--edport <*number*>

Port to be used by the ED for communicating with its associated SD.

--iomaxc <*number*>

IOM Max Connections.

--iotps <*number*>

IOM Thread Pool Size.

--iorqs <*number*>

IOM Request Queue Size.

--iomon ON|OFF

Enable/disable output to the IOM monitor file.

--iomonf <*number*>

Interval, in seconds, at which the SD is to write to the IOM monitor file.

--iomonp <*GPFS_path*>

Pathname of the IOM monitor file.

--simaxc <*number*>

SD IOM Max Connections.

--sitps <*number*>

SD IOM Thread Pool Size.

--sirqs <*number*>

SD IOM Request Queue Size.

--scmaxc <*number*>

SD Client Max Connections.

--sctps <*number*>

SD Client Thread Pool Size.

--scrqs <*number*>

SD Client Request Queue Size.

--sdport <*number*>

Port to be used in communicating with the SD to request and monitor file transfers.

--sdmon ON|OFF

Enable/disable output to the SD monitor file.

--sdmonf <*number*>

Interval, in seconds, at which the SD is to write to the SD monitor file.

--sdmonp <*GPFS_path*>

Pathname of the SD monitor file.

C.10.6. Notes

Any of the assigned values may include a reference to a parameter to be supplied with the ghiaddfs

command used to add a file system to the GHI configuration. There are five such references, and they are indicated by including the following strings for all or part of the default value:

- ?FS_Name - The File system name.
- ?Mount_Point - The mount point.
- ?UID - The unique identifier assigned by the ghiaddfs command.
- ?ED_Port - The ED port number, supplied by the user or assigned by the ghiaddfs command.
- ?SD_Port - The SD port number, supplied by the user or assigned by the ghiaddfs command.

For example, the following command would make the default path for the SD Monitor output to be a file in the root directory which name begins with the mount point and ends with "_mo_sd.out":

```
ghichfsdefaults --sdmonp "/?Mount_Point_mon_sd.out"
```

If the following ghiaddfs command is entered:

```
ghiaddfs ghifs2 ...
```

The associated SD Monitor file would be "/ghifs2_mon_sd.out".

C.11. ghichnode(7)

C.11.1. Name

ghichnode - Update the list of GHI nodes

C.11.2. Syntax

ghichnode [-Tv] --sort

C.11.3. Description

Use the ghichnode command to modify the list of nodes which make up the GHI cluster.

C.11.4. Parameters

-T

Test only — do all processing steps required but do not alter any files, or restart any processes.

-v

Verbose.

--sort

Sort the GHI node list. Whenever any of the GHI configuration management commands list or process nodes, it is done in the order in which nodes are added to the GHI cluster. Sorting the nodes will result in nodes being listed and processed in alphanumeric order.

C.12. ghichiom(7)

C.12.1. Name

ghichiom - Change configuration for an IOM

C.12.2. Syntax

ghichiom [-Tv] <*file_system*> <*IOM_node*>:<*port*> [-c "# <comment>"] [--asn {TRUE|FALSE}] [--etr <*rate*>] [--chunksize <*bytes*>]

ghichiom [-Tv] <*file_system*> --sort

C.12.3. Description

Use the ghichiom command to update an IOM in the GHI IOM configuration.

C.12.4. Options

-T

Test only — do all processing steps required but do not alter any files, or restart any processes.

-v

Verbose.

C.12.5. Parameters

<file_system>

The GHI-managed file system.

<IOM_node>:<port>

The IOM to be updated.

-c "# <comment>"

Add a comment to be associated with the IOM.

--asn {TRUE|FALSE}

Should the IOM remain active if the node becomes the GHI session node?

--etr <*rate*>

The estimated transfer rate (per second) GHI should use when calculating load balancing before actual transfer rates can be determined. Append *KB*, *MB*, or *GB* as a units multiplier.

--chunksize <*bytes*>

Maximum number of bytes to transfer per non-aggregate HPSS I/O request. Append *KB*, *MB*, *GB*, or *TB* as a units multiplier.

--sort

Alphanumerically sort the list of IOMs for this FS.

C.13. ghichlog(7)

C.13.1. Name

ghichlog - Change log levels for GHI servers

C.13.2. Syntax

ghichlog -l add|change|delete {loglevel1[,loglevel2,...,loglevelN] | ALL } [-T] [-v] [-a] [-p] [-m] [-c] [-t] [-f <*file system*>] [-s] [-e] [-q] [-i <*iom name*>] [-x]

C.13.3. Description

Use the ghichlog command to change the log levels for one or more GHI processes.

C.13.4. Parameters

-l add|change|delete {loglevel1[,loglevel2,...,loglevelN] | ALL }

This required argument is used to specify an action of add, change, or delete followed by one or more comma-separated log levels. Use *add* to add the log levels specified, *delete* to remove the log levels specified, or *change* to change the log levels to what is specified.

Valid log levels are:

- ALERT
- BACKUP
- CONFIG
- CRIT
- DB2
- DEBUG
- ERR
- GPFS
- HPSS
- INFO
- NOTICE
- POLICY
- QUEUE
- RPC
- THREAD
- TRACE
- TXN

- WARN

  Special log levels:

- ALL - apply the action to all log levels.

  See the admin guide for information on what each log level represents.

-T

Test only
do all processing steps required, but do not alter any log levels.

-v
Verbose. Display additional output during processing.

-a
Apply the specified log levels to all processes.

-p
Change the process manager's log levels.

-m
Change the mount daemon's log levels.

-c
Change the configuration manager's log levels.

-t
Change the tool's log levels.

-f <*file system*>
The name of the GHI file system to be modified must be specified if the Scheduler, Event Daemon, Policy, or IOM log levels are going to be changed.

-s
Change the scheduler's log levels. The file system must be specified with *-f*.

-e
Change the event daemon's log levels. The file system must be specified with *-f*.

-q
Change the policy's log levels. The file system must be specified with *-f*.

-i <*IOM name*>
Change the IOM's log levels. The IOM name must be of the form <*node*>:<*port*>. The file system must be specified with *-f*.

-x
Enable transaction logging for a GHI file system or tools. If enabling transaction logging for the

file system, *-f* must be specified with this option along with the component to modify (**-e** for event daemon, **-s** for scheduler, **-q** for policy, or **-i** for IOM). For tools the *-t* option must be specified.

C.13.5. Examples

```
ghichlog -l add DEBUG,THREAD,INFO -f demofs -s
Adds DEBUG, THREAD, and INFO log levels to the scheduler process for file system
demofs.

ghichlog -l delete DEBUG,WARN,QUEUE -p -m -c -t
Deletes DEBUG, WARN, and QUEUE from the log levels for processes process manager,
mount daemon, configuration manager, and tools.

ghichlog -l delete all -a
Changes the log levels for all processes to be just ALERT and CRIT.

ghichlog -l add all -a
Adds all log levels to all processes.

ghichlog -l change ALERT,CRIT,NOTICE -f demofs -i demofs1:8012
Changes the log levels to ALERT, CRIT, and NOTICE for the IOM named demofs1 on port
8012 for file system demofs.

ghichlog -f demofs -x
Adds TXN log level to the Scheduler, Event Daemon, Policy, and IOM processes for the
file system demofs.
```

C.13.6. Notes

1. The log levels CRIT and ALERT cannot be removed from any process.
2. The log levels may be entered in a case-insensitive manner.

C.13.7. See Also

ghilslog

C.14. ghicrcluster(7)

C.14.1. Name

ghicrcluster - Begin the GHI cluster configuration

C.14.2. Syntax

ghicrcluster [-v] <*GHI_node1*> <*GHI_node2*> <*GHI_node3*> ...

ghicrcluster [-v] -N <*nodelist_file*>

ghicrcluster [-v] -r

C.14.3. Description

Use the ghicrcluster command to create a new GHI cluster. GPFS must already be configured on the system and running on at least one node.

C.14.4. Options

-N

Accept a node file input.

-r

Resume a failed ghicrcluster.

-v

Verbose.

C.14.5. Parameters

GHI_node

Add node to GHI configuration.

nodelist_file

Add nodes listed in file to GHI configuration.

C.14.6. Notes

- The *-r* option is used to resume after a failed ghicrcluster command.
- Nodes to be included in the GHI configuration must be known to GPFS and specified as they are displayed in the "Admin node name" column from a GPFS *mmlscluster* command.

C.15. ghiCreateMapping(7)

C.15.1. Name

ghiCreateMapping - Create mapping tables for a file system

C.15.2. Syntax

ghiCreateMapping <*filesystem*>

C.15.3. Description

Use the ghiCreateMapping utility to create the mapping tables for file systems. ghiaddfs will automatically run this tool to create the mapping table when the filesystem is created. This utility only needs to be run if the file system does not have a mapping table, but can be run if one exists without causing damage to the existing tables.

C.15.4. Parameters

<filesystem>

Name of the GPFS file system you would like to create the mapping tables for.

C.16. ghi_debug(7)

C.16.1. Name

ghi_debug - A debug utility used to debug GPFS and HPSS interface (GHI) tasks like SD, IOM, and PM.

C.16.2. Synopsis

ghi_debug

C.16.3. Description

The ghi_debug command writes the current status of active processes of GHI to ghi_debug_<file system>.log at /var/hpss/ghi/log directory. This utility is used to get information about Scheduler daemon (SD), Process Manager (PM), and Input/Output Manager (IOM) task for the specified file system.

- sd dump - Dumps the Scheduler Queue, Work List, Completion Queue debug information and IOM list queue to ghi_debug.log file.
- pm dump - Dumps info about the Process Manager to ghi_debug_<file system>.log file at /var/hpss/ghi/log/ directory.
- iom dump - Dumps info about the SD connection, Non-Aggregate entries, Aggregate entries and Backup list to ghi_debug_<file system>.log at /var/hpss/ghi/log directory.

C.16.4. Parameters

use <*File System Mount Point*>

mention the File System to be queried.

sd dump

Dumps all of the above.

iom dump

Dumps the output of the IOM type.

pm dump

Dumps the PIDs on the pm.

exit, stop, quit

Exits and Deletes the Session.

C.17. ghidelfs(7)

C.17.1. Name

ghidelfs - Remove a file system from the GHI configuration

C.17.2. Syntax

ghidelfs [-fTv] <*file_system*>

C.17.3. Description

Use the ghidelfs command to remove a file system from the GHI configuration. Once it's removed, all GHI functionality is lost. (If the FS is subsequently re-added to the system, the data it contains may or may not still be good.)

C.17.4. Options

-f

Force — do not output an "Are you sure?" prompt for confirmation.

-T

Test only — do all processing steps required but do not alter any files, or restart any processes.

-v

Verbose.

C.17.5. Parameters

<file_system>

File system to be deleted.

C.17.6. Notes

- All IOMs configured for the FS must first be de-configured using the ghideliom command.
- The FS cannot be deleted if it has an associated GHI read-only FS. The read-only FS must be deleted first. To determine if it has an associated read-only FS, and if so, its name and the GHI cluster on which it resides so that the appropriate ghidelfs command can be executed on the required GHI cluster, execute the command:

```
ghilsfs <file_system> --afs --aclstr
```

C.18. ghideliom(7)

C.18.1. Name

ghideliom - Remove an IOM from the GHI configuration

C.18.2. Syntax

ghideliom [-RfTv] <*file_system*> <*IOM_node*>:<*port*>

C.18.3. Description

Use the ghideliom command to remove an IOM from the GHI configuration.

C.18.4. Options

-R

Push configuration changes to all GHI nodes and restart the GHI SD. (If this option is omitted, changes are only posted to the GHI session manager and the IOM node, and the SD is not restarted. The SD will continue to try to use the deleted IOM until the SD is restarted.)

-f

Force — do not output an "Are you sure?" prompt for confirmation.

-T

Test only — do all processing steps required but do not alter any files, or restart any processes.

-v

Verbose.

C.18.5. Parameters

<file_system>

FS associated with the IOM to be deleted.

<IOM_node>:<port>

The IOM to be deleted.

C.18.6. Notes

- If only a single IOM is to be deleted, use the *-R* option unless some reason exists why the change should not immediately be pushed to all GHI nodes and the SD restarted.
- To speed up the deletion of multiple IOMs for an FS, do not use the *-R* option until deleting the final IOM.
- If <*FS_name*> is to be deleted from the GHI configuration, all IOMs for it must first be deleted.

C.19. ghidelnode(7)

C.19.1. Name

ghidelnode - Remove a node from the GHI configuration

C.19.2. Syntax

ghidelnode [-fTv] <*node*>

C.19.3. Description

Use the ghidelnode command to remove a node from the GHI configuration.

C.19.4. Options

-f

Force — do not output an "Are you sure?" prompt for confirmation.

-T

Test only — do all processing steps required but do not alter any files, or restart any processes.

-v

Verbose.

C.19.5. Parameters

<*node*>

Node to be deleted

C.19.6. Notes

The node to be deleted must not be referenced from anywhere within the GHI configuration. To determine if a node is still referenced, use the command "ghilsnodes -c <*node*>". The appropriate GHI configuration commands can then be issued to remove any references.

C.20. ghi_dump_fs(7)

C.20.1. Name

ghi_dump_fs - List the contents of a GHI managed file system for inspection or future disaster recovery purposes.

C.20.2. Synopsis

ghi_dump_fs.py [-h] [-a] [-c *GPFS_CONFIG_FILE*] [-q] [-H] [-o *OUTPUT_DIR*] [-p *START_PATH*] [-i] [-n] [-t *THREAD_COUNT*] [-v] [-x] *file_system*

C.20.3. Description

Use the ghi_dump_fs command to list the contents of a GHI managed file system. The tool creates two output files: ghi_dumps_fs.log contains the list of contents on the file system and ghi_dump_fs.error.log contains any errors encountered while trying to list a directory or file on the file system. The output files are created in the /<file_system_mount_point>/scratch/.ghi directory by default.

By default, the command lists each file or directory in the output file on a single line, in a long listing format, and if applicable includes HPSS attributes and HPSS extended attributes.

C.20.4. Options

-h, --help

Show this help message and exit.

-a

Include hidden files/directories (names which begin with a .).

-c *GPFS_CONFIG_FILE*

Set the path to the GPFS Config File to <*config_file*>.

-q

Do not display the progress estimate.

-H

Display HPSS attributes for all files which reside in HPSS. UNIX and HPSS data will appear on separate lines to minimize chances of text wrapping around terminal screen.

-o *OUTPUT_DIR*

The directory to store the output files from this tool. (Default: /<file_system_mount_point>/scratch/.ghi)

-p *START_PATH*

List only the files/directories under this path on the file system. (Default: /<file_system_mount_point>)

-i

Show inode number.

-n

Output the UserID and GroupID as numeric values.

-t *THREAD_COUNT*

Maximum number of threads to run. (Default: 10)

-v

Displays the SOID version for files which reside in HPSS.

-x

Convert SOIDs to latest version for files which reside in HPSS.

C.20.5. Parameters

<file_system>

The name of a file system.

C.20.6. Notes

In order to configure log rotation, the ghi_dump_fs_config_logging tool should be run prior to using the ghi_dump_fs.py tool. Refer to the *GHI Admin Guide* for more information.

The performance of the ghi_dump_fs tool can be tuned by adjusting the thread count using the -t option. By default the thread count is set to 10. To increase throughput, where resources allow, the number of threads can be increased. It is best to be conservative with the thread pool size as a high number will reduce the performance on the system for other things. Also, increasing the thread pool size over a certain amount will lead to a decrease in the performance of the tool. Since the HPSS core server performs the brunt of the work, it is recommended that the thread pool size be set based on the number of CPU cores on the core server. As a starting point the thread pool size can be set to n+2, where n is the number of CPU cores available on the core server.

C.21. ghi_dump_fs_config_logging(7)

C.21.1. Name

ghi_dump_fs_config_logging - Creates a logrotate configuration file to be used by the ghi_dump_fs tool for a given GHI managed file system.

C.21.2. Synopsis

ghi_dump_fs_config_logging [-l *logrotate_dir*] [-m *max_age*] [-o *output_dir*] [-r *rotate*] [-h] *file_system*

C.21.3. Description

Use the ghi_dump_fs_config_logging command to create a logrotate configuration file for each GHI managed file system you intend to run the ghi_dump_fs tool on.

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.

Once the configuration file is created it is updated to reflect the rotate value specified with the "-r" option and maximum age value specified with the "-m" option. If these options are not specified the rotate value is set to a default of 5 and a maximum age directive will not be included in the configuration file. The configuration file is also updated with the output directory where you intend to store the output logs created with the ghi_dump_fs tool. By default the output directory location is /<file_system_mount_point>/scratch/.ghi. The "-o" option will allow a different output directory location to be specified. == Parameters

<file_system>

The name of the file system that will be dumped with the ghi_dump_fs tool. This must be the last parameter.

C.21.4. Options

-l

Location of the logrotate install directory (Default: /etc/logrotate.d)

-m

Set the maxage option in the logrotate configuration file. Removes rotated logs older than <*max_age*> days. (Default: not set)

-o

Set the output directory location where the ghi_dump_fs log files are intended to be stored. This should match the argument provided to the -o option of the ghi_dump_fs tool. (Default: /<file_system_mount_point>/scratch/.ghi)

-r

Set the rotate option in the logrotate configuration file. The number of archived log files to retain. (Default: 5)

-h

Display help text

C.22. ghi_dumpgc(7)

C.22.1. Name

ghi_dumpgc - Inspect rows in the garbage collection table

C.22.2. Synopsis

ghi_dumpgc [-h] *filesystem* [--show | --count | --badversion] [-s *SOID*] [-m *MIDX*] [-d *DIDX*] [-o *ORDINAL*] [-t *STATE*] [-c *COUNT_COL*] [-v *VERSION*] [-H *HASH*] [-i *INODE*] [-g *IGEN*]

C.22.3. Description

Use the **ghi_dumpgc** command to query the contents of the garbage collection table with optional filters on each column. By default, the command prints matching rows (**--show**). You may instead print only the number of matching rows using **--count**.

All filters are optional; if none are specified, the tool operates on **all rows** in the table. Filters perform **exact equality** on their respective column (e.g., `MIDX = 10`).

C.22.4. Options

-h, --help

Show this help message and exit.

filesystem

GHI filesystem name.

--show

Print matching rows to standard output. If neither `--show` nor `--count` is specified, `--show` is the default behavior.

--count

Print only the count of matching rows.

--badversion

Print only rows with an invalid version.

-s, --soid *SOID*

Filter by **SOID**. Format is e.g. 00000001070000000401F0D473D5272BB6.

-m, --midx *MIDX*

Filter by **MIDX**.

-d, --didx *DIDX*

Filter by **DIDX**.

-o, --ordinal *ORDINAL*
Filter by **ORDINAL**.

-t, --state *STATE*
Filter by **STATE**.

-c, --delete-count *COUNT_COL*
Filter by **COUNT**.

-v, --version *VERSION*
Filter by **VERSION**.

-H, --hash *HASH*
Filter by **HASH**.

-i, --inode *INODE*
Filter by **INODE**.

-g, --igen *IGEN*
Filter by **IGEN**.

C.22.5. Usage Notes

- If no filters are specified, all rows match.

C.22.6. Examples

List all rows (default behavior):

```
ghi_dumpgc myfs --show
```

Count all rows:

```
ghi_dumpgc myfs --count
```

Filter by SOID only and show rows:

```
ghi_dumpgc myfs --soid 00000539070000000301F0C700175EDC58 --show
```

Filter by multiple columns and return only the count:

```
ghi_dumpgc myfs --midx 10 --didx 5 --state 2 --count
```

C.22.7. Exit Status

0

Command completed successfully.

1

General error (e.g., SQL execution failure, connection error, or fetch error).

C.22.8. Files

None.

C.23. ghiFetchMapping(7)

C.23.1. Name

ghiFetchMapping - Fetch the mapping table for a backup index

C.23.2. Syntax

ghiFetchMapping <*filesystem*> <*backup index*> <*target dir*>

C.23.3. Description

Use the ghiFetchMapping utility to fetch the mapping table created while taking a backup. The command will put the .map file in the target directory specified in the command line.

C.23.4. Parameters

<*filesystem*>

Name of the GPFS file system you would like to fetch the mapping table for.

<*backup index*>

This is the backup index that you want to fetch the mapping table for.

<*target dir*>

This is the directory where you would like to place the mapping file in.

C.24. ghi_filehash(7)

C.24.1. Name

ghi_filehash - Manage GHI file hashes in GPFS

C.24.2. Synopsis

ghi_filehash *ACTION*

C.24.3. Description

Use the ghi_filehash command to manage GHI file hashes in GPFS. The tool can be used to add, list, and remove GPFS file hashes.

GHI can use these file hashes in some circumstances to validate the data during migration.

See the GHI Admin Guide for details.

C.24.4. Actions

-a, --add

filename hashtype hashvalue

-d, --delete

filename

-l, --list

filename

C.24.5. Parameters

-v

List all GHI hash attributes (type, hash, path)

Supported hash types (*hashtype*)

- MD5

C.24.6. Examples

```
# Set the file hash for a GPFS file
% ghi_filehash -a /path/to/file MD5 764efa883dda1e11db47671c4a3bbd9e

# List the checksums for a GPFS file
% ghi_filehash -l /path/to/file
MD5 764efa883dda1e11db47671c4a3bbd9e

# List all GHI hash attributes for a GPFS file
```

```
% ghi_filehash -v -l /path/to/file /path/to/file2
MD5 764efa883dda1e11db47671c4a3bbd9e /path/to/file
MD5 764efa883dda1e11db47671c4a3bbd9e /path/to/file2

# Delete the checksum for a GPFS file
% ghi_filehash -d /path/to/file
```

C.25. ghiListMapping(7)

C.25.1. Name

ghiListMapping - List the backup indexes loaded into the mapping table.

C.25.2. Syntax

ghiListMapping *<file system>*

C.25.3. Description

Use the ghiListMapping utility to list the backup indexes that are currently loaded into the mapping table for the specified file system.

C.25.4. Parameters

<file system>

Name of the GPFS file system to process.

C.26. ghiLoadMapping(7)

C.26.1. Name

ghiLoadMapping - Load the mapping for a backup index into the mapping table.

C.26.2. Syntax

ghiLoadMapping <*filesystem*> <*backup index* | *all*>

C.26.3. Description

Use the ghiLoadMapping utility to load the mapping information into the mapping table.

It can also be used to add all valid mapping information into the mapping table.

C.26.4. Parameters

<filesystem>

Name of the GPFS file system you would like to load the mapping information for.

<backup index | all>

The backup index that you would like to load the mapping information for.

If *all* is specified, then all valid backups will be loaded. Any already loaded backups will be skipped.

C.26.5. Examples

```
# Load backup 1 for "myFS"
% ghiLoadMapping myFS 1

# Load all valid backups for "myFS"
% ghiLoadMapping myFS all
```

C.27. ghi_ls(7)

C.27.1. Name

ghi_ls - file status utility

C.27.2. Syntax

ghi_ls [-a] [-c <*config_file*>] [-C] [-e] [-E] [-f] [-h] [-H] [-i] [-l] [-n] [-R] [-u] [-v] [-x] [-o] <*file/directory*> ...

C.27.3. Description

This utility is used to status files in an HPSS GPFS managed file system. The utility tells the user if the file's data is in HPSS, in GPFS, or both. Depending on the options specified, additional information about the file's HPSS data will be displayed.

If ghi_ls is run with a directory, then all the files in the directory will be displayed.

Because of how GPFS handles pathnames, GHI does not handle pathnames longer than 1024 characters. (GPFS limits individual components of a path to 256 chars.) Any pathname which exceeds this length will be displayed with "/.../" next to the final component to indicate that the total length of the path name exceeds the GHI maximum and that one or more directories are missing from the displayed pathname. For example, if the max path length were 24 instead of 1024 characters, the file "/234567890/234567890/XYZ123/ABC/abcd1234" would be displayed as "/234567890/234567890/.../abcd1234". And, if a command-line argument specifies a path longer than the GHI max pathname limit, ghi_ls will display a message to indicate that it cannot process it. For example, if the max path length were 24 instead of 1024 characters, the command "ghi_ls /234567890/234567890/XYZ123/ABC" would fail with an indication that the supplied pathname is too long. You may *cd* into a parent directory and reissue the command, e.g., "cd /234567890/234567890;ghi_ls [-R] XYZ123".

C.27.4. Options

-a

Include hidden files/directories (names which begin with a .).

-c

Set the path to the GPFS Config File to <config_file>.

-C

Display in classic *ls* format, i.e., GHI file and directory names by level. The standard display format is that directories are not shown, while files are shown with the complete pathname from the root position, i.e., from either . or the applicable command-line argument of a directory to traverse. The classic format is as *ls* would show, i.e., directories will be displayed on a line by themselves with a *:* appended to the directory name, and subordinate files and directories will be listed underneath. See below under EXAMPLES.

-e

Display HPSS extended attributes for all files which reside in HPSS. The extended attributes are the Migration Index, Class Of Service, File Family, HPSS checksum (if available), and a map of its use of the storage hierarchy within HPSS.

-E

Display HPSS extended attributes for all files which reside in HPSS. UNIX and HPSS data will appear on separate lines to minimize chances of text wrapping around terminal screen. (Shows the same data as *-e*.)

-f

<file/directory> is a file containing a list of files to display. Filelists are output of *ls* or *ghi_stage* command.

-h

Display HPSS attributes for all files which reside in HPSS. The HPSS attributes are the HPSS filename and the ordinal within the HTAR if the file was migrated into an HTAR (else, the ordinal will show as *N/A*).

-H

Display HPSS attributes for all files which reside in HPSS. UNIX and HPSS data will appear on separate lines to minimize chances of text wrapping around terminal screen. (Shows the same data as *-h*.)

-i

Show inode number.

-l

Long format, i.e., include UNIX details similar to "ls -l".

-n

Like option *-l* except that UserID and GroupID will be numeric.

-R

Recursively list subdirectories.

-u

Produce unsorted listing. Files and subdirectories are normally shown in alphabetical order. Unsorted listings require less time to produce. The time savings may become significant for directories that grow to thousands of files and directories.

-v

Displays the SOID version for files which reside in HPSS.

-x

Converts SOID version for files which reside in HPSS up to the current level.

-o

The path is wrapped in quotes to handle spaces or special characters, ensuring compatibility with file systems and shell environments. Use this option when paths may contain characters that might otherwise cause issues during processing.

C.27.5. Parameters

file/directory GPFS file or directory to check.

C.27.6. Examples

```
List files under the specified directory:
  % ghi_ls /gpfs_test_directory
  H   /gpfs_test_directory/file1
  G   /gpfs_test_directory/file2
  B   /gpfs_test_directory/file3

An example showing the difference the '-C' makes and how it compares
with 'ls':

  % ls -l * */*1
  lrwxrwxrwx 1 root root    12 Feb 28 09:55 xxx -> /ch1/IT/root

  root:
  total 80
  drwxr-xr-x 2 root root 16384 Mar  5 11:41 FUNC_01
  drwxr-xr-x 2 root root 65536 Feb 27 16:53 FUNC_02

  root/FUNC_01:
  total 10048
  -rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_0
  -rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_1
  -rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_2
  -rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_3
  -rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_4
  -rw-r--r-- 1 root root     1024 Mar  4 18:03 func_01_file_5
  -rw-r--r-- 1 root root     1024 Mar  4 18:04 func_01_file_6
  -rw-r--r-- 1 root root 10240000 Mar  5 11:41 func_01_file_7

  xxx/FUNC_01:
  total 10048
  -rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_0
  -rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_1
  -rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_2
  -rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_3
  -rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_4
  -rw-r--r-- 1 root root     1024 Mar  4 18:03 func_01_file_5
  -rw-r--r-- 1 root root     1024 Mar  4 18:04 func_01_file_6
  -rw-r--r-- 1 root root 10240000 Mar  5 11:41 func_01_file_7
```

```
       % ghi_ls -l * */*1
       -  lrwxrwxrwx   1  root      root            12 Feb 28 09:55 xxx ->
/ch1/IT/root
       -  drwxr-xr-x   2  root      root         16384 Mar 05 11:41 root/FUNC_01
       -  drwxr-xr-x   2  root      root         65536 Feb 27 16:53 root/FUNC_02
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21
root/FUNC_01/func_01_file_0
       BP -rw-r--r--   1  root      root          1024 Feb 25 16:21
root/FUNC_01/func_01_file_1
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21
root/FUNC_01/func_01_file_2
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21
root/FUNC_01/func_01_file_3
       BP -rw-r--r--   1  root      root          1024 Feb 25 16:21
root/FUNC_01/func_01_file_4
       BP -rw-r--r--   1  root      root          1024 Mar 04 18:03
root/FUNC_01/func_01_file_5
       H  -rw-r--r--   1  root      root          1024 Mar 04 18:04
root/FUNC_01/func_01_file_6
       B  -rw-r--r--   1  root      root      10240000 Mar 05 11:41
root/FUNC_01/func_01_file_7
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21
xxx/FUNC_01/func_01_file_0
       BP -rw-r--r--   1  root      root          1024 Feb 25 16:21
xxx/FUNC_01/func_01_file_1
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21
xxx/FUNC_01/func_01_file_2
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21
xxx/FUNC_01/func_01_file_3
       BP -rw-r--r--   1  root      root          1024 Feb 25 16:21
xxx/FUNC_01/func_01_file_4
       BP -rw-r--r--   1  root      root          1024 Mar 04 18:03
xxx/FUNC_01/func_01_file_5
       H  -rw-r--r--   1  root      root          1024 Mar 04 18:04
xxx/FUNC_01/func_01_file_6
       B  -rw-r--r--   1  root      root      10240000 Mar 05 11:41
xxx/FUNC_01/func_01_file_7

       % ghi_ls -lC * */*1
       -  lrwxrwxrwx   1  root      root            12 Feb 28 09:55 xxx ->
/ch1/IT/root

       root:
       -  drwxr-xr-x   2  root      root         16384 Mar 05 11:41 FUNC_01
       -  drwxr-xr-x   2  root      root         65536 Feb 27 16:53 FUNC_02

       root/FUNC_01:
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_0
       BP -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_1
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_2
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_3
```

```
       BP -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_4
       BP -rw-r--r--   1  root      root          1024 Mar 04 18:03 func_01_file_5
       H  -rw-r--r--   1  root      root          1024 Mar 04 18:04 func_01_file_6
       B  -rw-r--r--   1  root      root      10240000 Mar 05 11:41 func_01_file_7

       xxx/FUNC_01:
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_0
       BP -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_1
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_2
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_3
       BP -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_4
       BP -rw-r--r--   1  root      root          1024 Mar 04 18:03 func_01_file_5
       H  -rw-r--r--   1  root      root          1024 Mar 04 18:04 func_01_file_6
       B  -rw-r--r--   1  root      root      10240000 Mar 05 11:41 func_01_file_7

    An example showing the use of '-o'. The path is wrapped in quotes,
    making it easier to read the entire path if it contains spaces or
    special characters.

      % ghi_ls -o /gpfs_test_directory
      -  "/gpfs_test_directory/Test Dir2/"
      -  "/gpfs_test_directory/Test "2""
      -  "/gpfs_test_directory/Test "3\'/"

      An example showing the HPSS extended attributes. Note that the level data
      and checksum data can vary depending on the hierarchy and whether checksumming
      is enabled. The checksum displayed reflects the checksum on the file in HPSS.
      Automatic checksum generation for non-aggregate files can be enabled with
ghichfs.

      % ghi_ls -e *
      BP /gpfs_test_dir/a MIDX:110 COS:1 FF:0 L0-DISK:HY000500:6656 L1-
TAPE:E0101000:23061:0:6656
      H  /gpfs_test_dir/b MIDX:3 COS:1 FF:0 L0-DISK:HY000500:10 L1-
TAPE:E0100900:17:1064318283:10 md5:X'ED4E6832867D7FD384D8DEBEB98F9E2A'
      H  /gpfs_test_dir/c MIDX:3 COS:1 FF:0 L0-DISK::0 L1-
TAPE:E0100900:17:1064309721:5632

      Another example using the -E on the same files. Note that the attributes
      are multi-line.

      % ghi_ls -E *

      BP /gpfs_test_dir/a
         MIDX:110 COS:1 FF:0 L0-DISK:HY000500:6656 L1-TAPE:E0101000:23061:0:6656
      H  /gpfs_test_dir/b
         MIDX:3 COS:1 FF:0 L0-DISK:HY000500:10 L1-TAPE:E0100900:17:1064318283:10
         md5:X'ED4E6832867D7FD384D8DEBEB98F9E2A'
      H  /gpfs_test_dir/c
         MIDX:3 COS:1 FF:0 L0-DISK::0 L1-TAPE:E0100900:17:1064309721:5632
```

C.28. ghilsclusterconfig(7)

C.28.1. Name

ghilsclusterconfig - list the GHI cluster configuration

C.28.2. Syntax

ghilsclusterconfig [-Hv]

C.28.3. Description

Use the ghilsclusterconfig command to display the GHI cluster configuration.

The configuration is displayed like in the following example:

```
Field                     Value
-----------------------------------------------------------
GHI Version               4.2.0.0u0
Syslog Facility           8
DMG Principal             hpssdmg
SSM Principal             hpssssm
```

C.28.4. Options

-H

Display history of the key changes and the dates they were changed on.

-v

Verbose.

C.29. ghilsfs(7)

C.29.1. Name

ghilsfs - list the GHI file system configuration

C.29.2. Syntax

ghilsfs [-Hv] <*file_system*> ... <*key*> ...

C.29.3. Description

Use the ghilsfs command to display the GHI file system configuration for GHI-managed file system(s).

If neither file system nor key is specified, a name-only listing of all configured file systems will be displayed. The list is displayed like in the following example:

```
Key        Description                      Value (# comment)
-------------------------------------------------------------------------------
File System Name                gpfs1   # Example file system 1
File System Name                ghi2    # Example file system 2
```

If a file system(s) is specified without any keys, the complete configuration for each specified file system will be displayed. The configuration is displayed like in the following example:

```
Key        Description                      Value (# comment)
-------------------------------------------------------------------------------
File System Name                gpfs1   # Example file system 1
--mp       Mount Point                      /gpfs1
--uid      Unique Identifier                1
--junct    HPSS Junction                    /
--basep    HPSS Base Path                   /ghi
--bupath   HPSS Backup Path                 /ghi
--maxagg   Max Files Per Aggr             * 50000
--minagg   Min Files To Make Aggr         * 45000
--aggcos   Aggr Index COS                   1
--aggtps   Aggr Thread Pool Size            40
--bbc      Backup Bulk Count              * 50000
--bucos    Backup COS                       1
--pblock   Purged File Size                 0
--perf     Performance logging              NONE
--com      Checksum on Migrate              FALSE
--poiot    Purge Only If On Tape            FALSE
--edmaxc   ED Max Connections               5
--edtps    ED Thread Pool Size              50
--edrqs    ED Request Queue Size            10
--edport   ED Port Number                   8011
```

```
--iomaxc   IOM Max Connections           * 13
--iotps    IOM Thread Pool Size          * 7
--iorqs    IOM Request Queue Size          10
--iomon    IOM Monitor Flag              * ON
--iomonf   IOM Monitor Frequency         * 60
--iomonp   IOM Monitor Output Path         /gpfs1/scratch/mon/mon_iom.out
--simaxc   SD IOM Max Connections          20
--sitps    SD IOM Thread Pool Size       * 5
--sirqs    SD IOM Request Queue Size       100
--sidm     SD IOM DMAPI Thread Share       6
--sibu     SD IOM Backup Thread Share      5
--simg     SD IOM Migrate Thread Share     4
--sirc     SD IOM Recall Thread Share      3
--sirs     SD IOM Restore Thread Share     2
--siad     SD IOM Admin Thread Share       1
--scmaxc   SD Client Max Connections       100
--sctps    SD Client Thread Pool Size    * 100
--scrqs    SD Client Request Queue Size    100
--sdport   SD Port Number                  8010
--sdmon    SD Monitor Flag               * ON
--sdmonf   SD Monitor Frequency          * 60
--sdmonp   SD Monitor Output Path          /gpfs1/scratch/mon/mon_sd.out
--fstype   FS Type                         full-access
--bustat   Backup Status                   last good backup 5
--afs      Associated FS                   cheyenne1 (read-only)
--aclstr   Associated FS Cluster           cheyenne.clearlake.ibm.com
--amp      Associated FS Mount Point       /RO_ch2
--astat    Associated Status               restored to backup 2
```

If both file system(s) and key(s) are specified, the configuration matching key(s) for each file system specified will be displayed.

If only a key(s) is specified, the configuration matching key(s) will be displayed for all configured file systems.

Configuration values which differ from the currently set default value will be indicated by a * before the value.

C.29.4. Options

-H

Display history of the key changes and the dates they were changed on.

-v

Verbose.

C.29.5. Parameters

<file_system>

The name of a file system — only the specified FS(s) will be displayed.

<key>

A key from the above list — only the specified key(s) will be displayed.

C.29.6. Notes

To determine the currently set default configuration values, run the ghilsfsdefaults command.

C.30. ghilsfsdefaults(7)

C.30.1. Name

ghilsfsdefaults - list the GHI default file system configuration

C.30.2. Syntax

ghilsfsdefaults [-Hv]

C.30.3. Description

Use the ghilsfsdefaults command to display the GHI default file system configuration. This is the configuration that any subsequently added file systems will have. Also, the ghilsfs command will place an * next to any value which differs from the currently set default value.

The configuration is displayed like in the following example:

```
Key        Description                      Value (# comment)
-------------------------------------------------------------------------------
File System Name                 ?FS_Name
--mp       Mount Point                      ?Mount_Point
--uid      Unique Identifier                ?UID
--junct    HPSS Junction                    /
--basep    HPSS Base Path                   /ghi
--bupath   HPSS Backup Path                 /ghi
--maxagg   Max Files Per Aggr               100000
--minagg   Min Files To Make Aggr           90000
--aggcos   Aggr Index COS                   1
--aggtps   Aggr Thread Pool Size            40
--bbc      Backup Bulk Count                50000
--bucos    Backup COS                       1
--pblock   Purged File Size                 0
--perf     Performance logging              NONE
--com      Checksum on Migrate              FALSE
--poiot    Purge Only If On Tape            FALSE
--edmaxc   ED Max Connections               5
--edtps    ED Thread Pool Size              50
--edrqs    ED Request Queue Size            10
--edport   ED Port Number                   ?ED_Port
--iomaxc   IOM Max Connections              5
--iotps    IOM Thread Pool Size             15
--iorqs    IOM Request Queue Size           10
--sidm     SD IOM DMAPI Thread Share        6
--sibu     SD IOM Backup Thread Share       5
--simg     SD IOM Migrate Thread Share      4
--sirc     SD IOM Recall Thread Share       3
--sirs     SD IOM Restore Thread Share      2
--siad     SD IOM Admin Thread Share        1
```

```
--iomon    IOM Monitor Flag                OFF
--iomonf   IOM Monitor Frequency           1800
--iomonp   IOM Monitor Output Path         ?Mount_Point/scratch/mon/mon_iom.out
--simaxc   SD IOM Max Connections          20
--sitps    SD IOM Thread Pool Size         350
--sirqs    SD IOM Request Queue Size       100
--scmaxc   SD Client Max Connections       100
--sctps    SD Client Thread Pool Size      100
--scrqs    SD Client Request Queue Size    100
--sdport   SD Port Number                  ?SD_Port
--sdmon    SD Monitor Flag                 OFF
--sdmonf   SD Monitor Frequency            1800
--sdmonp   SD Monitor Output Path          ?Mount_Point/scratch/mon/mon_sd.out
```

C.30.4. Options

-H

Display history of the key changes and the dates they were changed on.

Example:

```
--iotps    IOM Thread Pool Size           * 7
# 2012/09/13 = 15 <- The previous settings
```

-v

Verbose.

C.30.5. Notes

The *?XXX* in the default configuration is replaced by the associated parameter supplied to the ghiaddfs command used to add a file system to GHI. This is except for *?UID*, which is generated by ghiaddfs.

C.31. ghilsiom(7)

C.31.1. Name

ghilsiom - list the GHI IO Managers configuration

C.31.2. Syntax

ghilsiom [-Htv] <*file_system*> <*IOM*> ...

C.31.3. Description

Use the ghilsiom command to display the GHI IOM configuration. If an IOM is specified, only the information for that IOM will be displayed. Otherwise the information for all IOMs for the file system will be displayed.

The configuration is displayed like in the following example:

```
Key        Description               Value (# comment)
-------------------------------------------------------------------------------
---------
IOM Node:Port             testnode1:8012
--asn       Active Session Node       TRUE
--etr       Estimated Transfer Rate   4MB
--chunksize Migration Chunk Size      1GB
```

C.31.4. Options

-H

Display history of the key changes and the dates they were changed on. Example:

```
--asn      Active Session Node       TRUE
# 2012/09/13 = FALSE <- The previous settings
```

-t

Alternate format Example:

```
IOM                                  ASN     ETR    CHUNK
-------------------------------------------------------------
testnode1:8012                       TRUE    4MB    1TB
```

-v

Verbose.

C.31.5. Parameters

<file_system>

The file system for which IOM configuration is to be displayed

<IOM>

The IOM(s) for which configuration is to be displayed. Any IOM whose name, i.e., "*<nodename>:<port>*", contains *<IOM>* will be displayed. For example, if *<IOM>* is "est", or "8", or even ":", the "testnode1:8012" IOM shown in the above examples would be displayed, but not if *<IOM>* were "TEST".

C.32. ghilslog(7)

C.32.1. Name

ghilslog - Show log levels for GHI servers

C.32.2. Syntax

ghilslog [-Tv] [-f <*file system*>]

C.32.3. Description

Use the ghilslog command to list the log levels of the cluster as a whole or for a specific file system.

C.32.4. Options

-T

Test only — do all processing steps required but do not alter any files, or restart any processes.

-v

Verbose.

C.32.5. Parameters

-f <*file_system*>

Name of the GHI file system which will be modified is needed if Scheduler, Event Daemon, Policy or IOM settings are going to be changed.

C.32.6. Examples

List the log levels of the file system demofs.

ghilslog -f demofs

C.32.7. See Also

ghichlog

C.33. ghilsnodes(7)

C.33.1. Name

ghilsnodes - list the GHI nodes

C.33.2. Syntax

ghilsnodes [-cv] <*node*>

C.33.3. Description

Use the ghilsnodes command to display the nodes configured for GHI.

The configuration is displayed like in the following example:

```
#> ghilsnodes
Node                                        GPFS Type
------------------------------------------------------------
testnode1                                   quorum-manager
testnode2
usernode1
usernode2
usernode3
usernode4
```

C.33.4. Parameters

-c

Display the detailed configuration for the node(s) given by "<*node*>", i.e., for all nodes whose nodename contains <*node*>.

Example:

```
#> ghilsnodes -c testnode
Node                                        GPFS Type
------------------------------------------------------------
testnode1                                   quorum-manager
IOM for FS gpfs1
testnode2
IOM for FS gpfs1
```

-v

Verbose.

C.34. ghi_ls_state(7)

C.34.1. Name

ghi_ls_state - file status utility

C.34.2. Syntax

ghi_ls_state [-a] [-c <*config_file*>] [-C] [-f] [-i] [-l] [-n] [-R] [-u] <*file/directory*> ...

C.34.3. Description

This utility is used to status files in an HPSS GPFS managed file system. The utility tells the user if the file's data is in HPSS, in GPFS, or both.

If ghi_ls_state is run with a directory, then all the files in the directory will be displayed.

Because of how GPFS handles pathnames, GHI does not handle pathnames longer than 1024 characters. (GPFS limits individual components of a path to 256 chars.) Any pathname which exceeds this length will be displayed with "/.../" next to the final component to indicate that the total length of the pathname exceeds the GHI maximum and that one or more directories are missing from the displayed pathname. For example, if the max path length were 24 instead of 1024 characters, the file "/234567890/234567890/XYZ123/ABC/abcd1234" would be displayed as "/234567890/234567890/.../abcd1234". And, if a command-line argument specifies a path longer than the GHI max pathname limit, ghi_ls_state will display a message to indicate that it cannot process it. For example, if the max path length were 24 instead of 1024 characters, the command "ghi_ls_state /234567890/234567890/XYZ123/ABC" would fail with an indication that the supplied pathname is too long. You may *cd* into a parent directory and reissue the command, e.g., "cd /234567890/234567890;ghi_ls_state [-R] XYZ123".

C.34.4. Options

-a

Include hidden files/directories (names which begin with a .).

-c

Set the path to the GPFS Config File to <config_file>.

-C

Display in classic *ls* format, i.e., GHI file and directory names by level. The standard display format is that directories are not shown, while files are shown with the complete pathname from the root position, i.e., from either . or the applicable command-line argument of a directory to traverse. The classic format is as *ls* would show, i.e., directories will be displayed on a line by themselves with a *:* appended to the directory name, and subordinate files and directories will be listed underneath. See below under EXAMPLES.

-f

<*file/directory*> is a file containing a list of files to display. Filelists are output of *ls* or *ghi_stage*

command.

-i

Show inode number.

-l

Long format, i.e., include UNIX details similar to "ls -l".

-n

Like option *-l*, except that UserID and GroupID will be numeric.

-R

Recursively list subdirectories.

-u

Produce unsorted listing. Files and subdirectories are normally shown in alphabetical order. Unsorted listings require less time to produce. The time savings may become significant for directories that grow to thousands of files and directories.

C.34.5. Parameters

file/directory GPFS file or directory to check.

C.34.6. Examples

List files under the specified directory:

```
% ghi_ls_state /gpfs_test_directory
H   /gpfs_test_directory/file1
G   /gpfs_test_directory/file2
B   /gpfs_test_directory/file3
```

An example showing the difference the *-C* makes and how it compares with *ls*:

```
% ls -l * */*1
lrwxrwxrwx 1 root root    12 Feb 28 09:55 xxx -> /ch1/IT/root

root:
total 80
drwxr-xr-x 2 root root 16384 Mar  5 11:41 FUNC_01
drwxr-xr-x 2 root root 65536 Feb 27 16:53 FUNC_02

root/FUNC_01:
total 10048
-rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_0
-rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_1
-rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_2
-rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_3
```

```
       -rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_4
       -rw-r--r-- 1 root root     1024 Mar  4 18:03 func_01_file_5
       -rw-r--r-- 1 root root     1024 Mar  4 18:04 func_01_file_6
       -rw-r--r-- 1 root root 10240000 Mar  5 11:41 func_01_file_7

       xxx/FUNC_01:
       total 10048
       -rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_0
       -rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_1
       -rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_2
       -rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_3
       -rw-r--r-- 1 root root     1024 Feb 25 16:21 func_01_file_4
       -rw-r--r-- 1 root root     1024 Mar  4 18:03 func_01_file_5
       -rw-r--r-- 1 root root     1024 Mar  4 18:04 func_01_file_6
       -rw-r--r-- 1 root root 10240000 Mar  5 11:41 func_01_file_7

       % ghi_ls_state -l * */*1
       -  lrwxrwxrwx   1  root      root            12 Feb 28 09:55 xxx ->
/ch1/IT/root
       -  drwxr-xr-x   2  root      root         16384 Mar 05 11:41 root/FUNC_01
       -  drwxr-xr-x   2  root      root         65536 Feb 27 16:53 root/FUNC_02
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21
root/FUNC_01/func_01_file_0
       BP -rw-r--r--   1  root      root          1024 Feb 25 16:21
root/FUNC_01/func_01_file_1
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21
root/FUNC_01/func_01_file_2
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21
root/FUNC_01/func_01_file_3
       BP -rw-r--r--   1  root      root          1024 Feb 25 16:21
root/FUNC_01/func_01_file_4
       BP -rw-r--r--   1  root      root          1024 Mar 04 18:03
root/FUNC_01/func_01_file_5
       H  -rw-r--r--   1  root      root          1024 Mar 04 18:04
root/FUNC_01/func_01_file_6
       B  -rw-r--r--   1  root      root      10240000 Mar 05 11:41
root/FUNC_01/func_01_file_7
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21
xxx/FUNC_01/func_01_file_0
       BP -rw-r--r--   1  root      root          1024 Feb 25 16:21
xxx/FUNC_01/func_01_file_1
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21
xxx/FUNC_01/func_01_file_2
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21
xxx/FUNC_01/func_01_file_3
       BP -rw-r--r--   1  root      root          1024 Feb 25 16:21
xxx/FUNC_01/func_01_file_4
       BP -rw-r--r--   1  root      root          1024 Mar 04 18:03
xxx/FUNC_01/func_01_file_5
       H  -rw-r--r--   1  root      root          1024 Mar 04 18:04
xxx/FUNC_01/func_01_file_6
```

```
       B  -rw-r--r--   1  root      root      10240000 Mar 05 11:41
xxx/FUNC_01/func_01_file_7

       % ghi_ls_state -lC * */*1
       -  lrwxrwxrwx   1  root      root            12 Feb 28 09:55 xxx ->
/ch1/IT/root

       root:
       -  drwxr-xr-x   2  root      root         16384 Mar 05 11:41 FUNC_01
       -  drwxr-xr-x   2  root      root         65536 Feb 27 16:53 FUNC_02

       root/FUNC_01:
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_0
       BP -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_1
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_2
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_3
       BP -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_4
       BP -rw-r--r--   1  root      root          1024 Mar 04 18:03 func_01_file_5
       H  -rw-r--r--   1  root      root          1024 Mar 04 18:04 func_01_file_6
       B  -rw-r--r--   1  root      root      10240000 Mar 05 11:41 func_01_file_7

       xxx/FUNC_01:
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_0
       BP -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_1
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_2
       H  -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_3
       BP -rw-r--r--   1  root      root          1024 Feb 25 16:21 func_01_file_4
       BP -rw-r--r--   1  root      root          1024 Mar 04 18:03 func_01_file_5
       H  -rw-r--r--   1  root      root          1024 Mar 04 18:04 func_01_file_6
       B  -rw-r--r--   1  root      root      10240000 Mar 05 11:41 func_01_file_7
```

C.35. ghi_map(7)

C.35.1. Name

ghi_map - File status utility

C.35.2. Synopsis

ghi_map <*file/directory*>

C.35.3. Description

This utility is used to status files in an HPSS file system. The utility tells the user all of the file's information as recorded by the HPSS system (e.g., Stripe Width, Stripe Length, Virtual Volume ID, Read Count, FilesetType, etc.).

C.35.4. Parameters

cleanup

Displays all but the current session.

status <*file_name*>

Displays all active sessions.

exit, stop, quit

Exits ghi_map and deletes the session.

C.35.5. Examples

```
% ghi_map
> status root_080201082226_1024512_1
dm_path_to_handle succeeded for root_080201082226_1024512_1
ghi_get_dmattr succeeded
hpss_FileGetAttributesSOID succeeded
hpss_Open succeeded
hpss_FileGetXAttributes succeeded
Account                       : 309
RealmId                       : 10000
Comment                       : NONE
CompositePerms                : 355
CosId                         : 94
BitfileId: ServerId           : 8fc2f822-ca64-11da-a0d5-000d604d1ffa
BitfileId: ObjectId           : df92b056-d0dc-11dc-9db8-000d604d1ffa
DataLength                    : 50051.5 KB
DMDataStateFlags              : 0
     CORE_DMSTATE_SYNC_FILESET: OFF
     CORE_DMSTATE_HPSS_DATA_VA: OFF
     CORE_DMSTATE_CACHE_DATA_V: OFF
```

```
DMHandleLength                : 0
DontPurge                     : 0
EntryCount                    : 0
ExtendedACLs                  : 0
FamilyId                      : 0
FilesetHandle ObjId           : 0
FilesetHandle FileId          : 0
FilesetHandle Type            : 0
FilesetHandle Flags           : 0
FilesetHandle Generation      : 0
FilesetHandle CoreServerUUID  : 00000000-0000-0000-0000-000000000000
FilesetId                     : 3272623367.1692889324
FilesetRootId                 : 0.1
FilesetStateFlags             : 3
     CORE_FS_STATE_READ       : ON
     CORE_FS_STATE_WRITE      : ON
     CORE_FS_STATE_DESTROYED  : OFF
FilesetType                   : HPSS Only
GatewayUUID                   : 00000000-0000-0000-0000-000000000000
GID                           : 220
GroupPerms                    : 0
LinkCount                     : 1
MACSecLabel                   : 0
OpenCount                     : 0
OutherPerms                   : 0
ReadCount                     : 1
RegisterBitMap                : 00000000.00000000
SetGIDBit                     : 0
SetStickyBit                  : 0
SetUIDBit                     : 0
TimeCreated                   : Tue Nov 17 21:41:52 1970
TimeLastRead                  : Tue Nov 22 18:00:00 1970
TimeLastWritten               : Thu Mar  6 20:44:16 1970
TimeModified                  : Wed Oct 16 17:19:44 1970
Type                          : File
UID                           : 309
UserPerms                     : 141
WriteCount                    : 1
SubSystemId                   : 1
Bytes At Level                [0]: 0 bytes
Optimum Access Size           [0]: 4 MB
Flags                         [0]: 1 (DISK)
Stripe Width                  [0]: 1
StripeLength                  [0]: 16 MB
Bytes At Level                [1]: 50051.5 KB
Optimum Access Size           [1]: 4 MB
Flags                         [1]: 6 (TAPE)
Stripe Width                  [1]: 1
StripeLength                  [1]: 1 MB
Relative Tape Position        [1]: 639
Bytes On VV                   [0]: 50051.5 KB
```

```
Virtual Volume ID            [1]: Object ID: de12886a-8ef5-11dc-a2c3-000d604d1ffa,
Tape VV
                                  Server ID: 8fc2f822-ca64-11da-a0d5-000d604d1ffa,
COS change flag FALSE
                                  DB2 format:
x'DE12886A8EF511DCA2C3000D604D1FFA8FC2F822CA6411DAA0D5000000000201'
     PV Name                 [0]: 3TX52000
     PV Flags                [0]: 0 (IN LIBRARY)
```

C.36. ghi_migrate(7)

C.36.1. Name

ghi_migrate - Migrate file data between GPFS and HPSS in accordance with policy rules. Also, via policy rules, used to purge file data from the GPFS file system, and reset files that were incompletely migrated.

C.36.2. Syntax

ghi_migrate MIGRATE *Filelist* [-s *ScratchArea*] [-d | -D] [-a | -P | -R] [-c *COS*] [-f *FileFamily*]

C.36.3. Description

Use the ghi_migrate command to migrate aggregate and/or non-aggregate file data from the online GPFS file system to the external HPSS storage pool.

It may also be used to purge the contents of files from the GPFS file system or to reset files back to "not managed" if for some reason they are not completely migrated into HPSS, so that another migration may be attempted. ("ghi_ls -h" will indicate if a file has not been completely migrated into HPSS.)

The script requires a filelist as input. The filelist contains a list of the files to be migrated, purged, or reset. The filelist is required to exist in a path that can be accessed by all nodes running IOMs and/or the SD. The filelist must be a fully qualified path. The filelist is almost always generated by running the ghiapplypolicy command with a policy with an EXEC clause which names ghi_migrate and the FLAGS (see below) contained in the policy's OPTS clause.

Each line in the filelist describes one file. The lines are in the following format:

```
    Inode Igen Epoch -s FileSize  -- Path
```

where:

- Inode is the inode number associated with the file
- Igen is the generation number of the inode
- Epoch is the version identifier
- Path is the full path of the file

"-s FileSize" is not required when *-P* or *-R* used.

The script generates 4 output files:

- input_file
- input_file.list_of_files
- input_file_<X>.exc

- input_file_<X>.ok

<X> = *migrate*, *purge*, or *reset*.

C.36.4. Flags

-a

All files in the input list will be placed into one or more aggregate files within HPSS. Lack of *-a* means that each file will be separately migrated into HPSS.

-c *COS*

COS to be used in HPSS.

-d

Output files for the *.ok and *.exc files will not be deleted from the GPFS file system scratch area.

-D

No output files will be deleted from the GPFS file system scratch area.

-f *FileFamily*

File family to be used in HPSS.

-P

Purge file's data from GPFS, making the re-claimed disk space available for other files. Files which have not been migrated into HPSS will not be purged, nor will files which have been "pinned" or which HPSS has not yet written to tape, if the GHI "purge only if on tape" configuration option is in effect.

-R

Reset incompletely migrated files to "not managed". Note that all files named in the input file are assumed to be incompletely migrated; if they have in fact been migrated into HPSS, the HPSS data will be orphaned.

-s *ScratchArea*

Sets the scratch area. Default is "<mount point>/scratch/.ghi".

C.36.5. Notes

- -a, -P, and -R are mutually exclusive.
- -d and -D are mutually exclusive.
- -c and -f not applicable, and ignored with *-P* or *-R*.

Files may only be purged from a GHI read-only file system. Any attempt to migrate (either aggregate or non-aggregate) or reset files on a read-only file system will be terminated with an error code of -1 (operation not permitted).

C.37. ghi_mitigate_recall_storm(7)

C.37.1. Name

ghi_mitigate_recall_storm - Enable or disable GPFS recall storm mitigation

C.37.2. Syntax

ghi_mitigate_recall_storm *filesystem -l|on|off*

C.37.3. Description

The **ghi_mitigate_recall_storm** command enables or disables GPFS recall storm mitigation for a specified filesystem.

Recall storm mitigation helps reduce the impact of delete events within snapshots.

C.37.4. Options

filesystem>
: The name of the GHI filesystem to modify.

-l
: Provide the current value of the mitigate-recall-storm option (off/on)

on|off
: Toggle to enable (`on`) or disable (`off`) recall storm mitigation.

C.37.5. Examples

List the recall storm mitigation on filesystem `fs1`:

```
ghi_mitigate_recall_storm fs1 -l
```

Enable recall storm mitigation on filesystem `fs1`:

```
ghi_mitigate_recall_storm fs1 on
```

Disable recall storm mitigation on filesystem `fs1`:

```
ghi_mitigate_recall_storm fs1 off
```

C.38. ghi_mount(7)

C.38.1. Name

ghi_mount - Mounts GHI file system(s) on cluster manager then mounts on other nodes

C.38.2. Syntax

ghi_mount.ksh {Device | DefaultMountPoint | DefaultDriveLetter | all | all_local | all_remote | {-F *DeviceFileName*}} [-o *MountOptions*] [-a | -N {Node[,Node...] | NodeFile | NodeClass}]

or

ghi_mount.ksh *Device* {MountPoint | DriveLetter} [-o *MountOptions*] [-a | -N {Node[,Node...] | NodeFile | NodeClass}]

C.38.3. Description

This tool is used to mount GHI file systems first on the cluster manager and then mount the file system on the other nodes. This tool uses the same parameters and options as mmmount.

C.38.4. Parameters

***Device* | *DefaultMountPoint* | *DefaultDriveLetter* | all | all_local | all_remote | {-F *DeviceFileName*}**

Indicates the file system or file systems to be mounted.

Device

The device name of the file system to be mounted. File system names need not be fully-qualified. fs0 is just as acceptable as /dev/fs0.

DefaultMountPoint

The mount point associated with the file system as a result of the mmcrfs, mmchfs, or mmremotefs commands.

DefaultDriveLetter

The Windows drive letter associated with the file system as a result of the mmcrfs or mmchfs command.

all

Indicates all file systems known to this cluster.

all_local

Indicates all file systems owned by this cluster.

all_remote

Indicates all files systems owned by another cluster to which this cluster has access.

-F *DeviceFileName*

Specifies a file containing the device names, one per line, of the file systems to be mounted. This must be the first parameter.

DriveLetter

The location where the file system is to be mounted. If not specified, the file system is mounted at its default drive letter. This option can be used to mount a file system at a drive letter other than its default one or to mount a file system that does not have an established default drive letter.

MountPoint

The location where the file system is to be mounted. If not specified, the file system is mounted at its default mount point. This option can be used to mount a file system at a mount point other than its default mount point.

C.38.5. Options

-a

Mount the file system on all nodes in the GPFS cluster.

-N {Node[,Node...] | NodeFile | NodeClass}

Specifies the nodes on which the file system is to be mounted. For general information on how to specify node names, see the following IBM Spectrum Scale: Administration and Programming Reference topic: Specifying nodes as input to GPFS commands. This command does not support a NodeClass of mount.

-o *MountOptions*

Specifies the mount options to pass to the mount command when mounting the file system. For a detailed description of the available mount options, see the following IBM Spectrum Scale: Administration and Programming Reference topic: GPFS-specific mount options.

C.38.6. Exit Status

- 0 - Successful completion.
- nonzero - A failure has occurred.

C.38.7. Security

You must have root authority to run the ghi_mount command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. See the following IBM Spectrum Scale: Administration and Programming Reference topic: Requirements for administering a GPFS file system.

C.38.8. Examples

To mount all GPFS file systems on all of the nodes in the cluster, issue this command:

```
ghi_mount.ksh all -a
```

To mount file system fs2 read-only on the local node, issue this command:

```
ghi_mount.ksh fs2 -o ro
```

To mount file system fs1 on all NSD server nodes, issue this command:

```
ghi_mount.ksh fs1 -N nsdsnodes
```

C.39. ghi_pin(7)

C.39.1. Name

ghi_pin - Marks GPFS files for exclusion from GHI threshold processing.

C.39.2. Synopsis

ghi_pin [-v] [-u] [-p] {file | wildcard | -f <*filelist*>}

C.39.3. Purpose

The purpose of this tool is to be able to mark files that are not desired to be purged during a GPFS policy run. The tool accomplishes this by setting a special extended attribute on a GPFS file. The key to the extended attribute that can be used in GPFS policies is _GHI_PIN. This attribute is an integer which indicates the time (in seconds since epoch) when ghi_pin was run, or 2147483647 (max epoch time) if "-p" was passed on the command line. For a list of files pinned by ghi_pin, they will all receive the same pin time based on when the ghi_pin binary started pinning files.

The GPFS policies may have an exclude rule like the following to exclude the files that have been marked.

```
RULE 'exclude3' EXCLUDE FROM POOL 'system' WHERE XATTR_INTEGER('dmapi._GHI_PIN')
```

C.39.4. Options

-v

Verbose. This mode tells the application to provide information messages about the file. For example, if a file is HPSS-resident only a message will be printed that the file needs to be staged. When not specified only errors are reported.

-u

Deletes the extended attribute.

-p

Pins the file "permanently" which sets the pin time to 2147483647 (max epoch time).

-f <*filelist*>

Specifies a file name that contains on each line the full path names of the files to process.

C.40. ghi_recall(7)

C.40.1. Name

ghi_recall - Recall file data between GPFS and HPSS in accordance with policy rules.

C.40.2. Syntax

ghi_recall RECALL *Filelist* -s *ScratchArea* [-d | -D]

C.40.3. Description

This script is invoked to recall aggregate and/or non-aggregate file data from the external HPSS storage pool to the online GPFS file system.

The script requires a filelist as input. The filelist contains a list of the files to be migrated, purged, or reset. The filelist is required to exist in a path that can be accessed by all nodes running IOMs and/or the SD. The filelist must be a fully qualified path. The filelist is almost always generated by running the ghiapplypolicy command with a policy with an EXEC clause which names ghi_recall and the FLAGS (see below) contained in the policy's OPTS clause.

Each line in the filelist describes one file. The lines are in the following format:

```
Inode IGen Epoch -s FileSize  -- Path
```

where:

- Inode is the inode number associated with the file
- Igen is the generation number of the inode
- Epoch is the version identifier
- Path is the full path of the file

The script generates output files and directories/subdirectories:

- recall.<pid>
- recall.<pid>/*

C.40.4. Flags

-d

Output files for the *.ok and *.exc files will not be deleted from the GPFS file system scratch area.

-D

No output files will be deleted from the GPFS file system scratch area.

-s *ScratchArea*

Scratch flag that sets the scratch area to <mount point>/scratch/.ghi by default.

C.41. ghirepackfs(7)

C.41.1. Name

ghirepackfs - Repack sparsely used aggregates

C.41.2. Syntax

ghirepackfs <*filesystem*> -r <*repack ratio*> [-T] [-v]

ghirepackfs <*filesystem*> -f <*repack file*> [-T] [-v]

C.41.3. Description

Use the ghirepackfs utility to repack sparsely used aggregates. This will free up previously unavailable disk space. This command finds sparse aggregates and compares the percent used to the repack ratio specified in the tool. If the percent used is equal to or above the repack ratio the tool will repack that aggregate. The tool can also repack specific aggregates by using the force option.

C.41.4. Options

-r

Use this option to have ghirepackfs scan the file system and find the percent used and compares it to the repack ratio specified.

-f

Force Option — Use this option to force the repack of specific aggregates.

-T

Test Only — generate a list of files that will be repacked.

-v

Verbose option.

C.41.5. Parameters

<*filesystem*>

Name of the GPFS file system you would like to repack.

<*repack ratio*>

Sets the ratio for repack selection. This is the percent of the aggregate that is being used [0-100].

<*repack file*>

This is the file that contains the list of aggregates to repack.

C.42. ghi_restore(7)

C.42.1. Name

ghi_restore - Restore a GHI-managed file system from a backup

C.42.2. Syntax

ghi_restore [-v] [-t] [-w *secs*] [-c *config_file*] [-R <*BUIdx*>] [-x] -g <*temporary space*> <*file system*>

C.42.3. Description

ghi_restore is a tool that will restore a GHI-managed file system from a previous successful backup. The tool is interactive and will prompt the user to select which backup index to restore from.

C.42.4. Parameters

-f

Force restore of a backup without verifying backup image.

-v

Verbose.

-t

Allows target file system to be smaller than the original file system.

-w <*secs*>

Delay between command retry attempts.

-c <*config_path*>

Specifies an alternate configuration path.

-R <*BUIdx*>

Resume restoration of specified backup.

-x

Cancel incomplete restore.

-g <*temporary space*>

A separate GPFS-mounted file system temporary space in addition to the restoring file system where to place temporary backup files. <*file system*> - File system (string).

C.42.5. Note

ghi_restore runs a restore_error.policy after the namespace has completed successfully. This error policy verifies all files have valid DMATTRs. All errors found when running this policy are for informational purpose. We will not treat the restore as a failure if we find files that don't have valid DMATTRs.

C.42.6. Example

```
[root@rock ~]# ghi_restore ghi-rockfs1 -g /rockfs2
************************************************
* GHI FILESYSTEM RESTORE Main Menu
************************************************
Menu options are
1. Select a backup for restore.
Type Quit to exit
> 1

************************************************
* List of Available Backups
************************************************

Selection Index Type        Status      Timestamp
1         227   Image       Completed   Mon Mar 21 12:31:51 2016
2         230   Image       Completed   Mon Mar 21 13:04:51 2016
3         234   Image       Completed   Mon Mar 28 10:15:23 2016
4         235   Image       Completed   Mon Mar 28 15:33:50 2016
5         236   Image       Completed   Mon Mar 28 15:35:54 2016
6         239   Image       Completed   Mon Apr 18 11:29:12 2016

************************************************
* Input the selection to list all the associated backups.
* Type List to display the entries again.
* Type Back to return to the main menu.
* Type Quit to exit.
************************************************
> 6

********************************************
* List of all backups associated with selection
* Dated Mon Apr 18 11:29:12 2016
********************************************

Selection Index Type        Status      Timestamp
1         239   Image       Completed   Mon Apr 18 11:29:12 2016
************************************************
Input the selection to choose a backup to restore
Type List to return to display the entries again
Type Back to return to the main menu
Type Quit to exit
************************************************
> 1
************************************************
* Backup to restore
************************************************

Backup Index 239         Timestamp Mon Apr 18 11:29:12 2016
```

```
The following processing will be performed:
Files to be orphaned in HPSS: 0
No backups will be invalidated
************************************************
* Please check the above information and
* confirm that it is correct.
************************************************
Type Restore to start the restore process
Type Back to return to the main menu
Type Quit to exit
> Restore
Start Restore
******************
Restore the GPFS filesystem configuration.
Warning: "Network Options" section not found in HPSS.conf file
Unmounting the filesystem
Tue Apr 19 14:15:45 CDT 2016: mmunmount: Unmounting file systems ...
Restoring FS config...
.
.
.
Restore completed successfully
```

C.43. ghiSearchMapping(7)

C.43.1. Name

ghiSearchMapping - Search the mapping table for a file

C.43.2. Syntax

ghiSearchMapping -f <*filesystem*> {-h | -H | -g | -i} <*value*> [-e]

C.43.3. Description

Use the ghiSearchMapping utility to determine the GPFS name for an HPSS file without GPFS. The utility uses a database mapping table to search for the given backup indexes. The user can specify an HPSS file name and the ghiSearchMapping utility will return the corresponding GPFS file if found. If the user passes in the GPFS file name, the utility will return the HPSS file name if found. The user can also pass in the inode number and get the corresponding file name.

C.43.4. Options

-f

Filesystem — This required option is used for specifying the GPFS filesystem.

-h

HPSS filename — Use this option when passing in an HPSS filename. By default, this will use a pattern matched search that can include wildcards using %. See wildcards section below.

-H

HPSS filename — Use this option when passing in an HPSS filename to return the number of records that match. By default, this will use a pattern matched search that can include wildcards using %. See wildcards section below.

-g

GPFS filename — Use this option when passing in a GPFS filename. By default, this will use a pattern matched search that can include wildcards using %. See wildcards section below.

-i

Inode — Use this option when passing in an inode number.

-e

Use this option to disable pattern matching and do an exact search for the filename specified. If not specified, an SQL LIKE match will be done for the filename specified which can take longer to complete. This argument has no effect when used with the -i argument.

C.43.5. Wildcards

ghiSearchMapping supports using *and % as wildcard characters. The* character is for a single character wildcard while % is for zero or more characters.

For example -h "%.txt" would match any file that ends in .txt, while -h "%._if" would match files that end in .gif, .tif, or .jif, but not .txif.

To find all the files in the directory "/my/directory" you would use "-h /my/directory/%".

The wildcards can be placed anywhere in the path and can be escaped by using a backslash (e.g. -h "\%.txt" will match %.txt). However, for exact matches, it is better to use the -e option.

C.43.6. Parameters

<filesystem>

Name of the GPFS file system to search the mapping for.

<value>

This is value passed to the mapping utility after specifying -h, -H, -g, or -i. It can be either a GPFS filename, an HPSS filename, or an inode number.

C.44. ghishutdown(7)

C.44.1. Name

ghishutdown - Stop GHI and/or GPFS

C.44.2. Syntax

ghishutdown [-f] -g

ghishutdown [-f] -G

ghishutdown [-f] -i { ALL | <*FS_name*> } [<*node*> ...]

C.44.3. Description

Use the ghishutdown command to stop the GHI servers, restart the GHI IOMs, or stop GPFS. Stopping GPFS will first stop the GHI servers.

Note that a normal shutdown will force kill any remaining servers, except for the SD, after *GHI_SHUTDOWN_WAIT_SECS*. This behavior can be disabled by setting *GHI_SHUTDOWN_WAIT_SECS* to 0. The default time is 5 minutes.

C.44.4. Options

-g
: Stop GHI only.

-G
: Stop both GHI and GPFS.

-i
: Restart IOMs for all or selected FS on all or selected nodes.

-f
: Force.

C.44.5. Parameters

ALL
: Restart IOMs for all file systems.

<*FS_name*>
: Restart IOMs only for the specified FS.

<*node*>
: Restart IOMs only on the specified nodes.

C.44.6. Notes

In a normal shutdown, shutdown of GHI or an IOM will be delayed until all on-going file transfers have been completed. GHI processes will terminate as they complete their work. If the -g/-G option is specified, all new file transfer requests will be rejected by the GHI servers as they come down.

In a forced shutdown, the affected GHI processes will be immediately terminated and any on-going file transfers will be aborted — which will leave affected files in an unpredictable state.

C.45. ghi_stage(7)

C.45.1. Name

ghi_stage - File staging utility

C.45.2. Syntax

ghi_stage [<options>] <*file/directory*> [<*file/directory*> ...]

ghi_stage [<options>] -f <*filelist*> [<*filelist*> ...]

C.45.3. Description

This utility is used to stage files which have been purged from GPFS.

ghi_stage will execute an order, sort, and recall step to translate the input GPFS file names into files in HPSS to stage, organize those files and remove duplicates, and then recall them in an optimal manner.

For files which reside on tape, ghi_stage will provide a *tape unit* reference. This is merely a unique identifier and not a reference to a specific tape volume.

ghi_stage provides a callout functionality (-C) to allow the user to specify an action to occur for each completed stage. The user must specify an executable that can be forked. When specified, that executable will be called for each file staged with the path and the return code of the stage operation. This can be used, for example, to pin staged files, or to integrate with other workflows.

ghi_stage also supports a dry run (-r). The dry run option will follow the same steps as the non-dry run mode, except for the actual staging. That will simply calculate the bytes and files required by the inputs and display them to the user. Note that the dry run mode does not support setting a timeout (-t) or providing a callout (-C).

C.45.4. Options

-R

Recursively stage files in subdirectories.

-t <*n*>

Wait *n* seconds for stages to complete. *n* = 0 means do not wait. Default is wait forever.

-C <*exec*>

Provide a executable that can handle being called with arguments <*stage-path*> <*return-code*> An return code of 0 indicates success; anything else represents failure.

-v

Show verbose output.

-r

Dry run mode. Only shows the number of files and bytes that will be staged outside of dry run.

C.45.5. Parameters

file/directory

GPFS file or directory to be staged.

filelist

Pathname of filelist naming files to be staged.

C.45.6. Examples

To stage files within a directory without waiting:

```
% ghi_stage -t 0 /gpfs_test_directory/stuff
```

Run a dry run stage within a directory:

```
% ghi_stage -r /gpfs_test_directory/stuff
```

C.46. ghistartup(7)

C.46.1. Name

ghistartup - start GHI and/or GPFS

C.46.2. Syntax

ghistartup -g

ghistartup -G [-a | {-N <*nodes*>}] [-E <*env_var*>=<*value*>] ...

C.46.3. Description

Use the ghistartup command to start either the GHI daemons on the current GPFS cluster manage or the GPFS daemons on one or more nodes. If using option *-G* to start GPFS, then GPFS will start GHI.

C.46.4. Options

For more information on options [-a], [-N], and [-E] please see documentation for mmapplypolicy

-g
: Start GHI (assumes GPFS already running).

-G
: Start GPFS (GPFS will start GHI).

-a
: Start GPFS on all nodes (default).

-N <*nodes*>
: Start GPFS on <*nodes*> where <*nodes*> is <*node*>*[,*<*node*>*,...]* or <*node_file*> or <*node_class*>.

-E <*env_var*>=<*value*>
: Pass an environment variable to GPFS.

C.47. ghi_state(7)

C.47.1. Name

ghi_state - Get the state of the GHI system

C.47.2. Syntax

ghi_state <*file system*>

C.47.3. Description

This tool tells you the state of the GHI system. It will give the GPFS cluster status, the GPFS Disk Status, the GPFS Mount Status, the GPFS manager Node and the GHI Session Node.

C.47.4. Example

```
[root@rock ghi_state]# ghi_state ghi-rockfs1
------ GPFS Cluster Status -----

 Summary information
---------------------
Number of nodes defined in the cluster:            2
Number of local nodes active in the cluster:       2
Number of remote nodes joined in this cluster:     2
Number of quorum nodes defined in the cluster:     2
Number of quorum nodes active in the cluster:      2
Quorum = 2, Quorum achieved

------- GPFS Disk Status -------
All disks up and ready

------- GPFS Mount Status ------
File system ghi-rockfs1 is mounted on 4 nodes.

------- GPFS Manager Node ------
Cluster manager node: fc00::220:52 (rock)

------- GHI Session Node -------
rock.clearlake.ibm.com
# SD_START_TIME = 1460996627
[root@rock ghi_state]#
```

C.48. ghi_undelete_file(7)

C.48.1. Name

ghi_undelete_file - Undelete a file in GPFS that is still in HPSS

C.48.2. Synopsis

ghi_undelete_file <*filesystem*> [-v] < -d *restore_dir* > [-f *filelist* | *file...*]

C.48.3. Description

Use the ghi_undelete_file to restore files which were deleted from GPFS which still reside in HPSS. The tool can be used to recover a set of files provided as arguments, or a list of files in a file list.

This tool is limited to files which are part of a backup. Files which are migrated but not backed up cannot be restored using this tool.

By default, the files are restored to their original location in GPFS, including creating any necessary directories. However, using the -d option, the files can be restored to an alternate directory. This will put the GPFS file name in the specified GPFS directory rather than the original location.

C.48.4. Arguments

filename

GPFS file name to restore from HPSS.

-v

Verbose output.

-d *restore_dir*

Optional alternative directory to restore.

-f *file*

File containing GPFS file names to restore.

C.48.5. Examples

```
# Restore a file
% ghi_undelete_file myfs my_deleted_file

# Restore a list of files
% ghi_undelete_file myfs -f my_list_of_deleted_files
```

C.49. ghiUnloadMapping(7)

C.49.1. Name

ghiUnloadMapping - Unload the mapping information for a backup index

C.49.2. Syntax

ghiUnloadMapping <*filesystem*> <*buidx*>

C.49.3. Description

Use the ghiUnloadMapping utility to unload the mapping information for the given backup index.

It can also be used to remove all loaded mapping information from the mapping table.

C.49.4. Parameters

<*filesystem*>

Name of the GPFS file system you would like to unload the mapping information for.

<*buidx* | *all*>

The backup index that you would like to unload from the mapping table.

If *all* is specified, then all loaded backups will be removed.

C.50. ghiupdate(7)

C.50.1. Name

ghiupdate - bring the GHI cluster configuration into compatibility with the current version of GHI

C.50.2. Syntax

ghiupdate [-dTv] {--aix|--ppc64le|--ppc_64|--x86_64 <*node*>} ...<*GHI_node*>

C.50.3. Description

Use the ghiupdate command to bring the configuration of an existing GHI installation into compatibility with the newly installed update. Must be executed from the GHI source node and GHI must not be running.

C.50.4. Options

-d

Delete unknown nodes.

-T

Test only — do all processing steps required but do not alter any files, or restart any processes.

-v

Verbose.

C.50.5. Parameters

--aix <*node*>

Use this node as the source node for the GHI AIX binaries and libraries.

--ppc64le <*node*>

Use this node as the source node for the GHI RHEL PowerLE PC binaries and libraries.

--ppc_64 <*node*>

Use this node as the source node for the GHI RHEL Power PC binaries and libraries.

--x86_64 <*node*>

Use this node as the source node for the GHI RHEL X86 binaries and libraries.

<*GHI_node*>

GHI will copy over the GHI binaries to this node(s).

C.50.6. Notes

The updated binaries must be installed on at least one node of each type of hardware/OS combination if the cluster is heterogeneous, this node is called the source node. The ghiupdate

command will then transfer binaries from these to all other nodes listed in the <*GHI_node*> parameter.

Each <*node*> parameter, and each node listed in the to-be-updated GHI nodelist file, "/var/hpss/ghi/etc/ghinode.conf", must match an Admin node as listed in the output of a GPFS *mmlscluster* command. If necessary, edit "/var/hpss/etc/ghinode.conf" to change node names as required, or they will be seen as no longer existing (see next paragraph). It is highly recommended that ghiupdate be first executed in test mode (option *T*) to make certain that all nodes listed in the GHI nodelist file are handled correctly.

The behavior of when a node is listed in the GHI configuration but no longer exists in the GPFS configuration for the cluster depends on whether or not the *-d* option has been specified. If it was specified, the unknown node is deleted from the GHI configuration (with an explanatory message). Else, it is considered an error and the ghiupdate command will be aborted. A non-Admin node is considered to not exist.

When executed to upgrade from a pre-2.4 installation of GHI, an entry for each configured file system (run the ghilsfs command on 2.3x systems, look in file "/var/hpss/etc/ghi.conf" on 2.2x systems) will be added to a GHI DB2 table that holds a list of all GHI file systems which [will] have data stored in the HPSS to which this installation of GHI connects. It will be an error if the name or mount point of any configured FS is already configured in any other GHI installation which also connects to this GHI's HPSS.

C.51. ghiverifyaggr(7)

C.51.1. Name

ghiverifyaggr - Verifies the Spectrum Scale ordinal to HPSS aggregate mapping in a file system.

C.51.2. Syntax

ghiverifyaggr <*file system*> <*file containing list of files or policy output*> <*temp dir*> <*step to run*> [-v]

C.51.3. Description

Use the ghiverifyaggr utility once a year or after a suspected discrepancy between Spectrum Scale and HPSS to identify and fix incorrect ordinal values in the metadata.

C.51.4. Options

-v

Verbose

C.51.5. Parameters

<***file system***>

Name of the GPFS file system to process.

<***file containing list of files or policy output***>

List of files you want to verify aggregate metadata on.

<***temp dir***>

Temporary directory for use by the tool.

<***step to run***>

The particular step in the ghiverifyaggr process to run. Running the ghiverifyaggr utility is a multi-step process and requires steps to be run in a certain order. The soid step is a standalone step and is not a required step. The sort, order, and index steps should be run in that order as each step depends on the results of the previous step. After the index step you can run the compare or the compare_size steps, where these steps just print the results of the compare. Or the comparefix step will perform the compare and fix any incorrect ordinal values.

--soid

Lists the SOIDs for the input file list.

--sort

Sort the files into tape groups and by aggregate.

--order

Order the list of aggregate files created by the sort step.

--index

Create an index file for each aggregate file in the ordered file list created in the order step.

--compare

For each local index file created in the index step, compare each entry in a local index file with each entry in the aggregate file to ensure the inode and ordinal values match. This step will just print the results of the compares.

--compare_size

For each local index file created in the index step, compare each entry in a local index file with each entry in the aggregate file to ensure the file sizes match. This step will just print the results of the compares.

--comparefix

Performs the compare step (by inode) above, but will also fix any incorrect ordinal values.

C.51.6. Notes

The policy output file that can be used as input to this tool can be generated during a policy run by adding a "-d" or "-D" in the policy's OPTS field. The generated okay file (file extension .ok) will be located in the /<*file system*>/scratch/.ghi directory.

The format for an entry in the input file is:

<*INODE*> <*IGEN*> 0 0 — <*filepath*>

Example line:

089024 1690824023 0 0 — /rfs1/agg/file_e.aakw

C.52. ghiverifyfs(7)

C.52.1. Name

ghiverifyfs - This tool finds files that have been orphaned in HPSS or files that have not been migrated.

C.52.2. Syntax

ghiverifyfs <*file system*> -l *outputfile* < -f | -o | -u > < -n *num-threads* >

C.52.3. Description

Use the ghiverifyfs tool to verify if a file system has orphaned files in HPSS or verify if files in the file system have not been migrated.

C.52.4. Options

-f

File system verification

-o

Check for orphan files

-u

Check that the file UDAs are correct

-n <*num threads*>

Number of concurrent threads to use (default=40)

C.52.5. Parameters

<file system>

Name of the GPFS file system to verify.

outputfile

Name of the output file to write to.

C.52.6. Notes

Tuning the file system verification.

ghiverifyfs -f runs a policy which uses backup verification to identify files which have not been migrated. Performance of this scan requires tuning GHI and the Spectrum Scale policy. The number of IOMs, the number of available IOM threads, the restore thread share count, the Backup Bulk Count for the FS, and policy scan parallelism all come into play.

The policy scan parallelism by default is 40, but can be increased via the -n option. For best

performance, the SD must have enough IOM threads (simaxc, sitps) across all IOMs for the number of threads, the IOMs must have enough connections and thread pool size (iomaxc, iotps), the scheduling of the SD must allow for the backup verification threads to run (--sirs), and the Backup Bulk Count (--bbc) should be set to a value of at least 50,000. Bottlenecks will occur if the Backup Bulk Count is too low, if the default scan parallelism is too low, if the SD or IOM thread or request limits have been hit, or if the restore thread share causes the verification requests to be held off for other work. These values can be tuned with ghichfs.

C.53. ghi_verify.py(7)

C.53.1. Name

ghi_verify.py - This tool finds files that have been orphaned in HPSS.

C.53.2. Syntax

ghi_verify.py < -h | -d1 -d2 -g -s1 -t1 -s2 -f -o -l -L -D >

C.53.3. Description

Use the ghi_verify.py tool to verify if a GPFS file system has orphaned files in HPSS.

C.53.4. Options

-h | --help

Help.

-d1 | --ghi_db <*database name*>

HPSS GHI database name.

-d2 | --subsys_db <*database name*>

HPSS subsystem database name.

-g | --ghi_mount <*ghi mount point*>

GHI mount point name in HPSS. Default *ghi*.

-s1 | --ghi_schema <*schema name*>

Schema name to use in HPSS GHI database. Default *HPSS*.

-s2 | --subsys_schema <*schema name*>

Schema name to use in HPSS subsystem database. Default *HPSS*.

-t1 | --ghi_table <*table name*>

Table name to use for orphan candidates in HPSS GHI database. Default *ORPHAN_CANDIDATES*.

-f | --fs_name <*file system name*>

Name of GPFS file system defined in HPSS GHI database.

-o | --output_file <*file name*>

The base output file path/name for query results identifying orphans. Identified orphans information will be written in JSON format to <file name>_<levela>, where <*levela*> is *01 - 12*.

-l | --log_file <*log file*>

Console messages are also written to the specified log file.

-L | --level_a <*level a value*>

Specific levela value to process, valid values: 1 thru 12. Default is to process all levels.

-D | --debug

Turns on debug mode. SQL statements executed will be included in the console and log file output.

C.53.5. Notes

Identifying HPSS orphans.

ghi_verify.py loads orphan candidate file information associated with the specified GPFS file system from the HPSS subsystem database into the HPSS GHI database. Once loaded, it is compared to the associated GHI garbage collection and GHI mapping tables. If the orphan candidate file information is not referenced by both of the above tables it is written in JSON format to the output file for the specific level a value.

C.54. ghi_verify_userattrs.py(7)

C.54.1. Name

ghi_verify_userattrs.py - This tool verifies the HPSS user attributes of a file.

C.54.2. Syntax

ghi_verify_userattrs.py [-h] -f *FS_NAME* [-d1 *GHI_DB*] [-d2 *SUBSYS_DB*] [-g *GHI_MOUNT*] [-s1 *GHI_SCHEMA*] [-t1 *GHI_TABLE*] [-s2 *SUBSYS_SCHEMA*] [-o *OUTPUT_FILE*] [-l *LOG_FILE*] [-D {0,1,2,3,4}]

C.54.3. Description

Use the ghi_verify_userattrs.py tool to verify HPSS file attributes.

C.54.4. Options

-h | --help

Help.

-f | --fs-name

The GPFS file system name to verify.

-d1 | --ghi_db

GHI database name. Default *HGHI*.

-d2 | --subsys_db

HPSS subsystem database name. Default *HSUBSYS1*.

-g | --ghi_mount

GHI mount point name. Default *ghi*.

-s1 | --ghi_schema

Schema name of GHI tables in GHI database. Default *HPSS*.

-t1 | --ghi_table

Table name to use for load target table in GHI database. Default.

-s2 | --subsys_schema

Schema name for NSOBJECT and USERATTRS table in the subsystems database. Default *HPSS*.

-o | --output_file

The base output file path/name for query results. Validate attributes output will be written to validate_userattrs_<fs_name>_<levela> where <levela> is *01* = *12*. Default "/tmp/validate_userattrs_<fs_name>_<levela>".

-l | --log_file

Log messages are also written to a specified log file.

-D | --debug

Enables debug mode.

C.55. lsghi(7)

C.55.1. Name

lsghi - capture the GHI configuration for HPSS support

C.55.2. Syntax

lsghi

C.55.3. Description

Use the lsghi command to capture the GHI file system configuration. This is usually done after the site has completed their acceptance test. The command runs a combination of the GHI administrative commands that list the configuration.

The command outputs the configuration to stdout.

C.55.4. Parameters

There are no parameters.

C.56. lsgpfs(7)

C.56.1. Name

lsgpfs - capture the GPFS configuration for HPSS support

C.56.2. Syntax

lsgpfs

C.56.3. Description

Use the lsgpfs command to capture the GPFS file system configuration. This is usually done after the site has completed their acceptance test. The command runs a combination of the GPFS administrative commands that list the configuration.

The command outputs the configuration to stdout.

C.56.4. Parameters

There are no parameters.