GHI Management Guide

Spectrum Scale (GPFS) - HPSS Interface
Release 4.1.0.0.0, 21 August 2024

GHI Management Guide

Spectrum Scale (GPFS) - HPSS Interface Release 4.1.0.0.0, 21 August 2024

Table of Contents

List of Figures

Who Should Read This Book. The GHI Management 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. GHI basics

1.1. Introduction

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.

1.2. 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 the *GHI Installation Guide*, *Chapter 3: HPSS Basics* and *Chapter 4: HPSS Planning*.

Before performing the procedures described below, be certain that the appropriate system preparation steps have been performed. See the *GHI Installation Guide*, *Chapter 5: System Preparation* for more information. For a system created from scratch, be certain that the GHI installation and infrastructure configuration have been completed. See the *HPSS Installation Guide*, *Chapter 6: HPSS Installation and Infrastructure Configuration* for more information. To upgrade from a previous system, see the release notes for the upgraded version for specifics on the upgrade procedures.

1.2.1. 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
   ```

   Note

   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.

1.2.2. 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 the *GHI Installation Guide*.

1. Configure the GHI file system. The settings to be changed can be listed with **ghilsfs** and changed with **ghichfs**. 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.

   a. 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.

   b. Next, the minimum and maximum files per aggregate. These settings will depend on the cutoff size of aggregate versus nonaggregate files.

   c. 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** and **ghistartup**. 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.

1.2.3. 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 2. GHI concepts

2.1. 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** 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.

2.2. 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.

2.2.1. 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**.

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).

2.2.2. 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**. 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** command is the alternative to recalling files using a policy. **ghi_stage** can take as input either file lists or a specific files and directories to recall. See the **ghi_stage** man page for details.

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** 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** and passing in the **--dse** option.

2.2.3. 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.

2.2.4. 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.

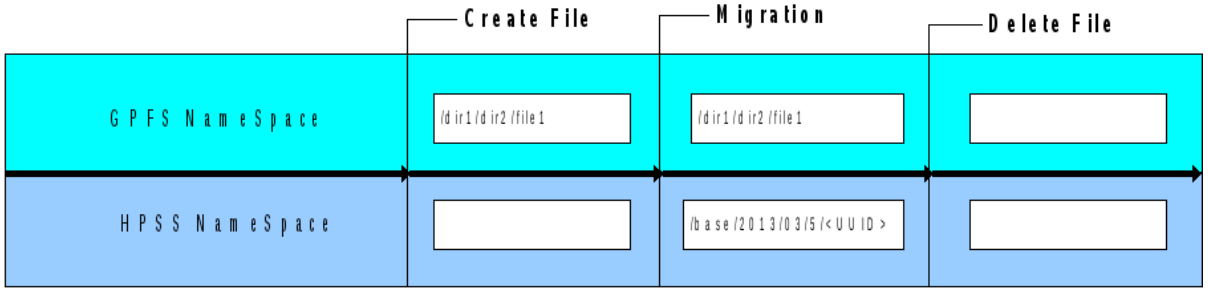
- If the file is part of a backup that is invalidated due to a restore of an earlier backup, the file will be deleted from HPSS.
- If the file is part of another backup, GHI will retain the notification in the GC table until all referencing backups are deleted.

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.

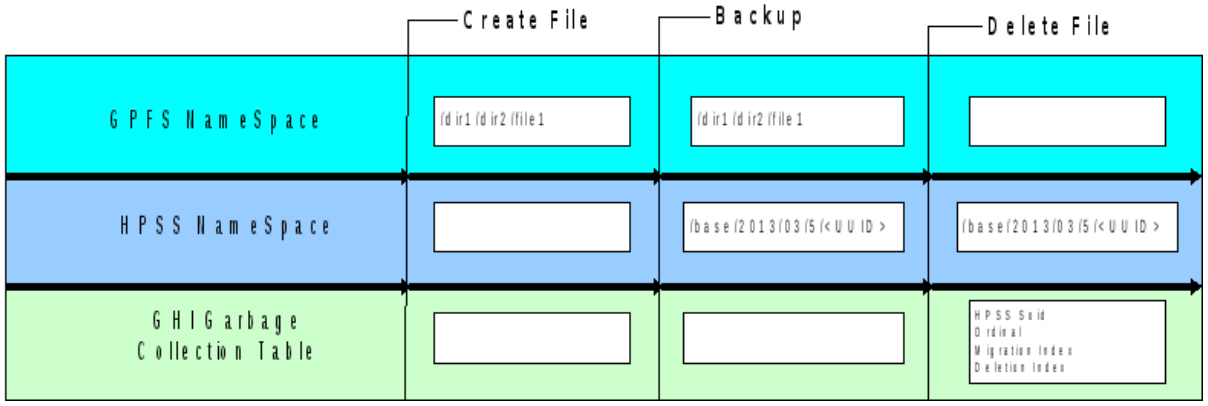
Figure 2.1. Namespace Mapping Table



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 2.2. 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.

2.2.5. 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.

2.3. 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. Chapter 5 of this guide provides more details on mapping.

2.3.1. 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.

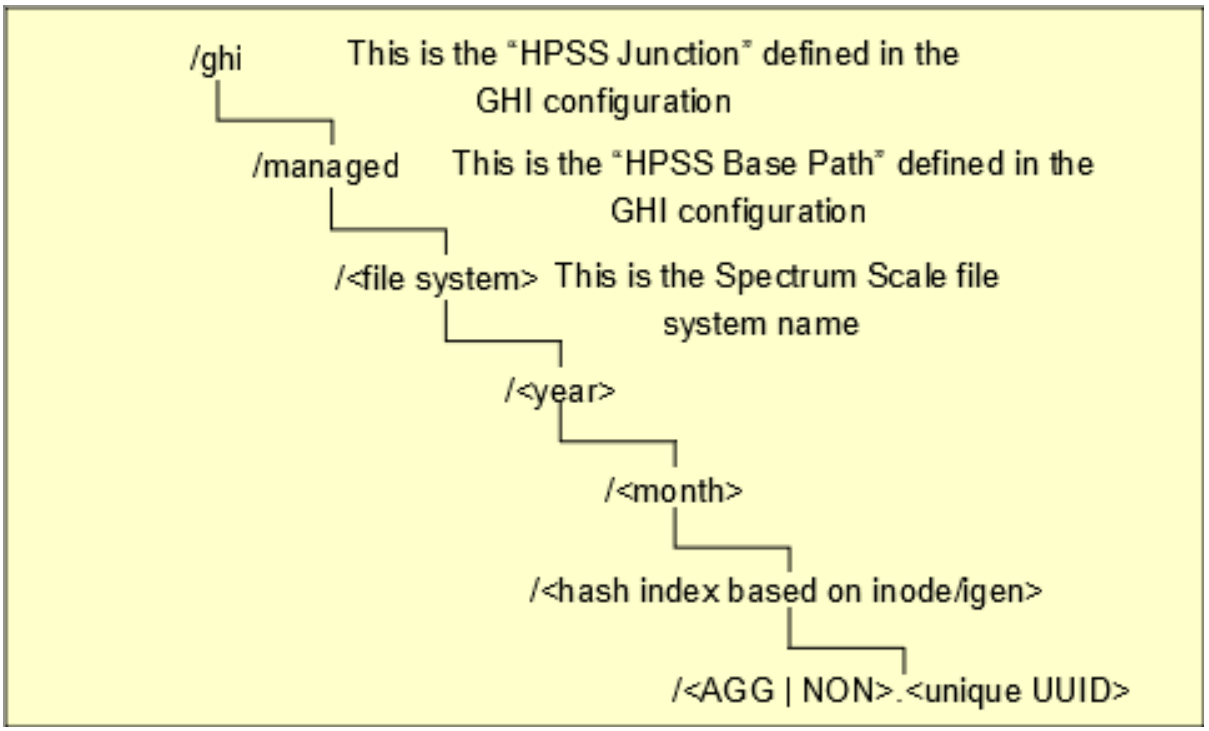
Note

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.

2.3.2. Location of Spectrum Scale Files in HPSS

Figure 2.3. 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 2.4. Location of image backup files in HPSS

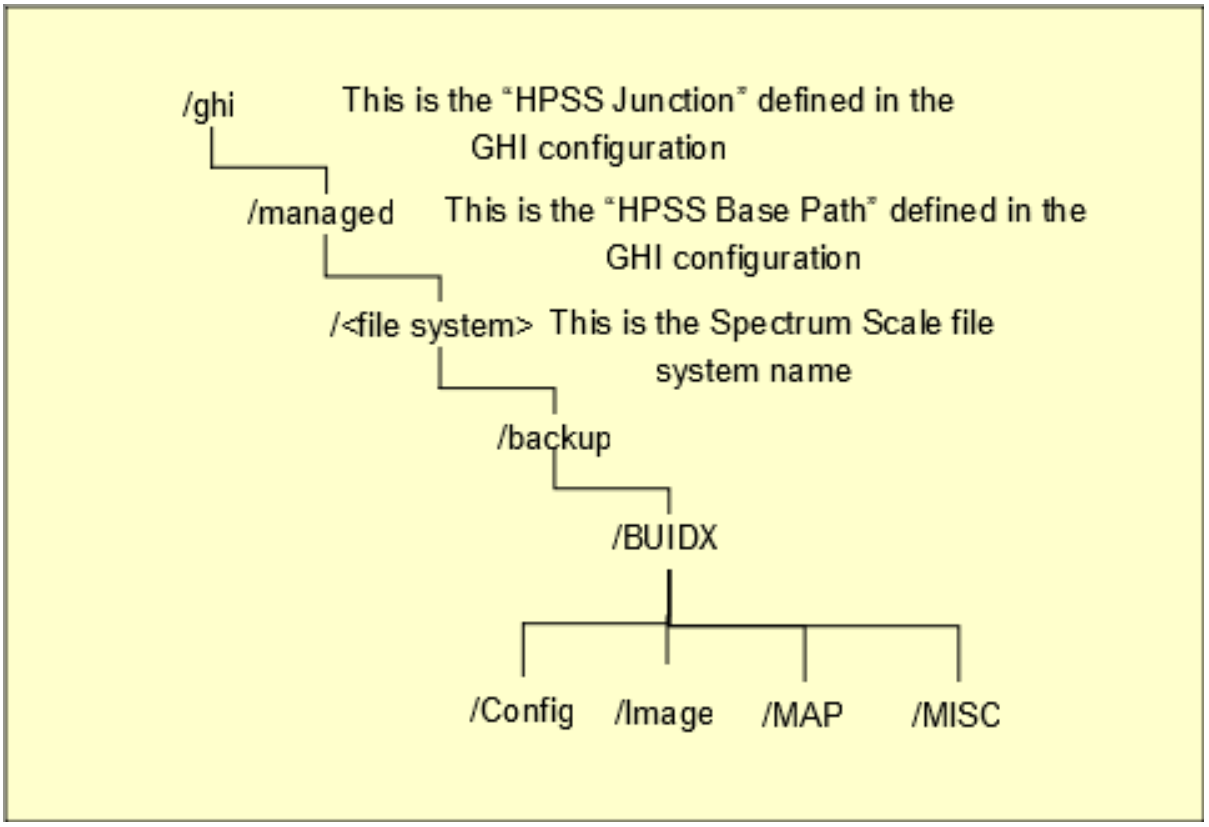


Image backup files are stored in HPSS based on the backup index of the backup.

- **/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.

2.3.3. 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 Management 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 Management Guide* for in-depth information on file families, classes of service, and storage subsystems.

Chapter 3. GHI internals

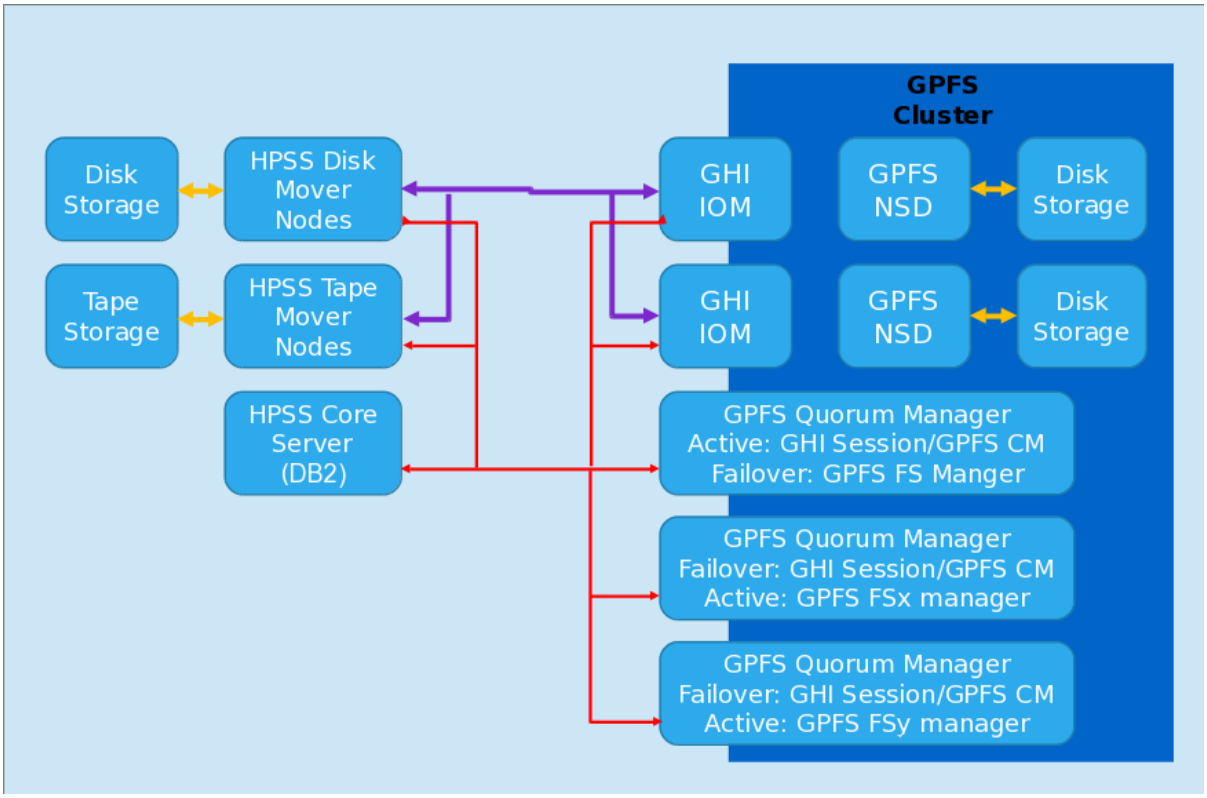
3.1. Servers and processes

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 3.1. 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** 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 3.2. GHI session node

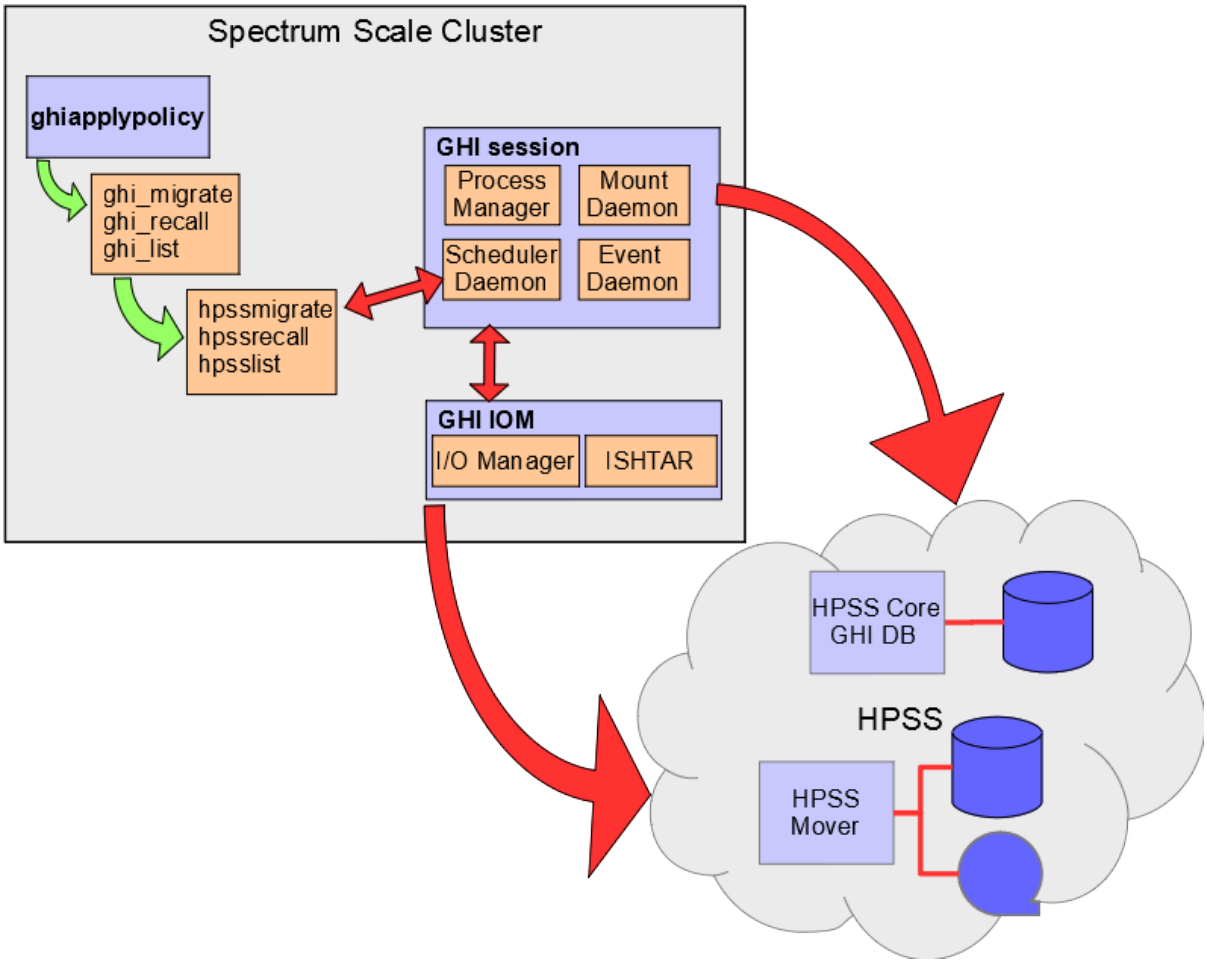
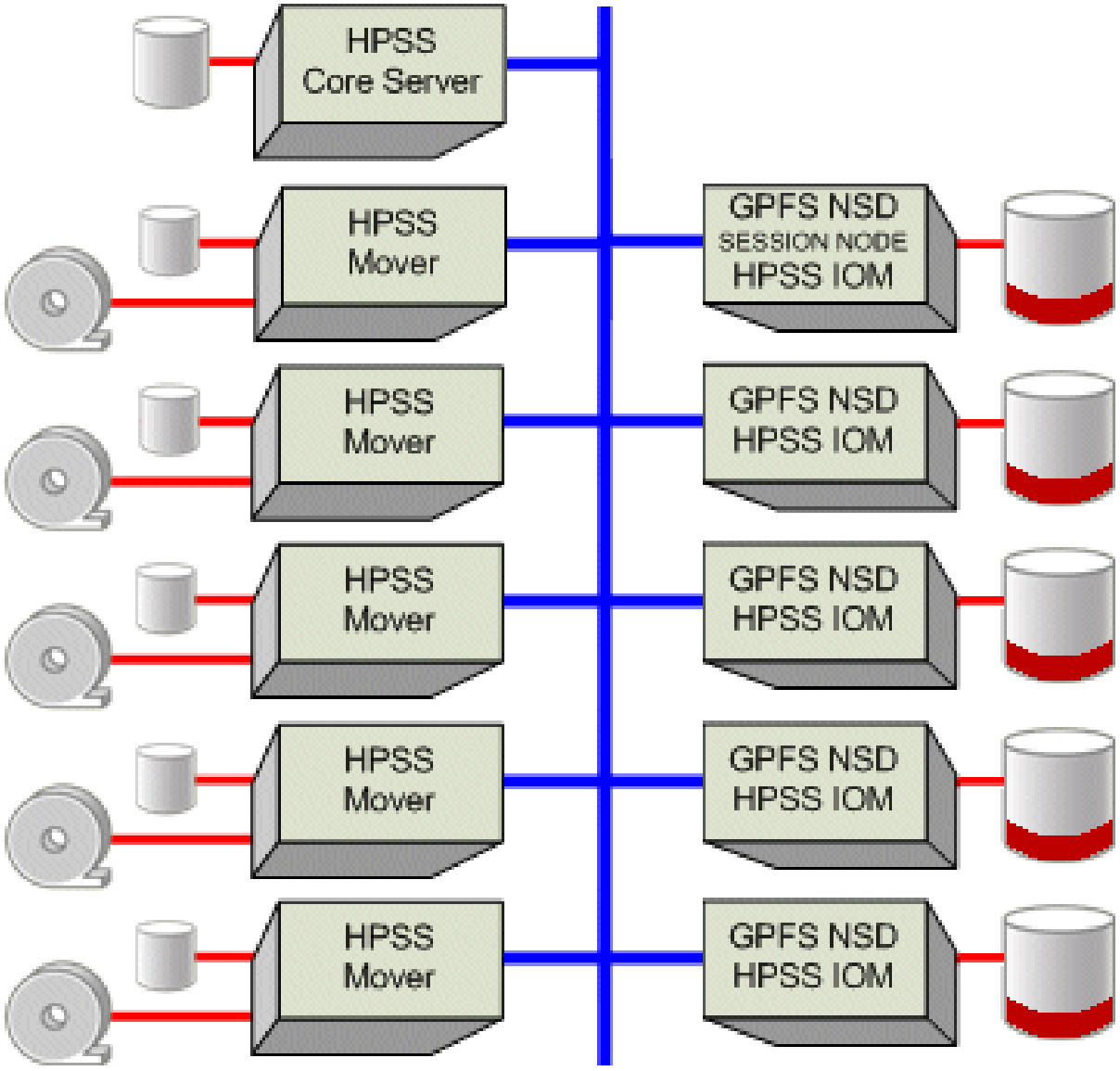


Figure 3.3. GHI I/O manager



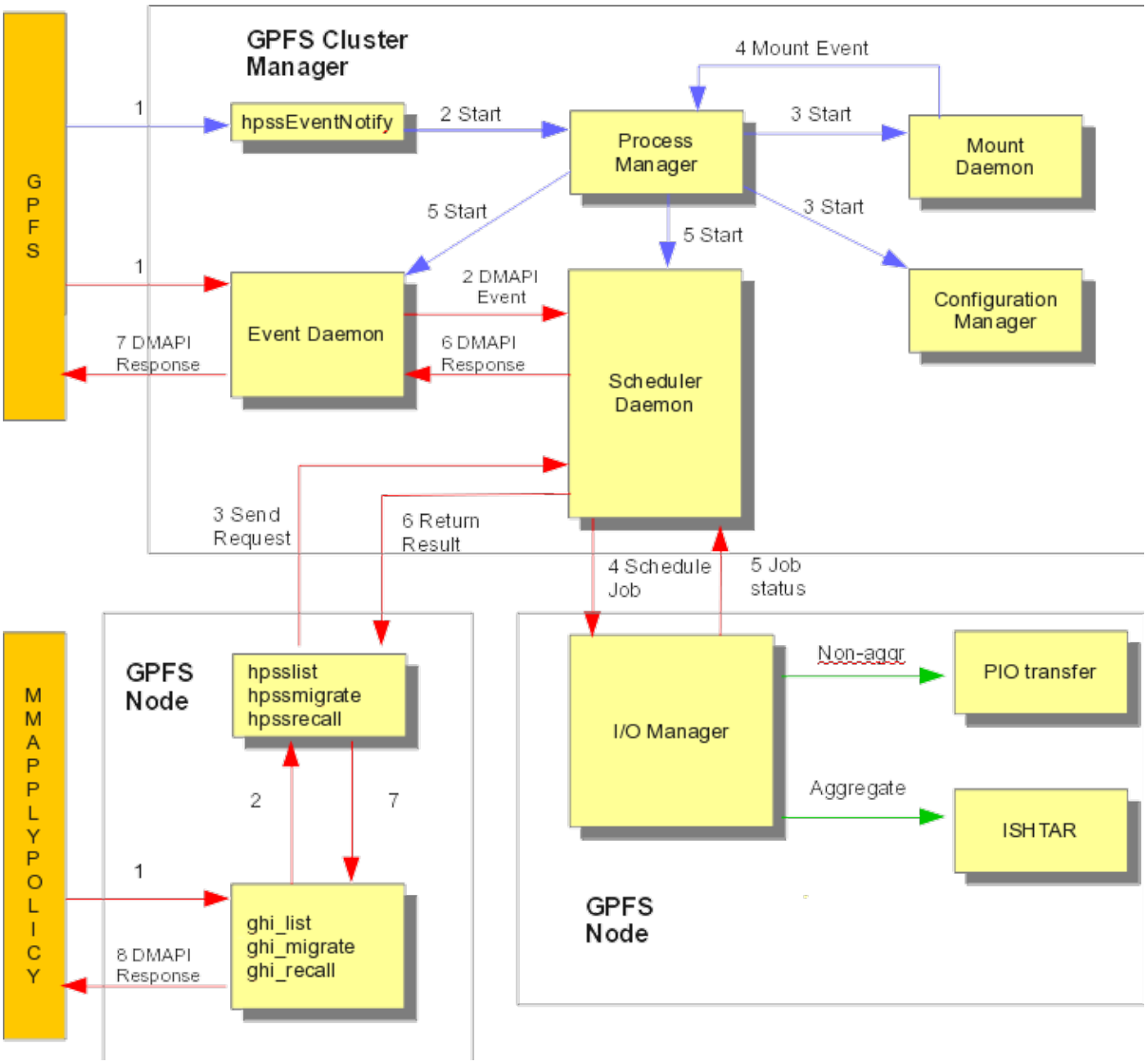
ISHTAR. Software that aggregates files before migrating them into HPSS. ISHTAR clients are installed on each I/O manager node. The ISHTAR process is started via a script, `htar.ksh`, that resides in the `/var/hpss/hsi/bin/` directory. An ISHTAR process starts when an IOM requests processing of an aggregate. 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.

3.2. 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 3.4. 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**. 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).

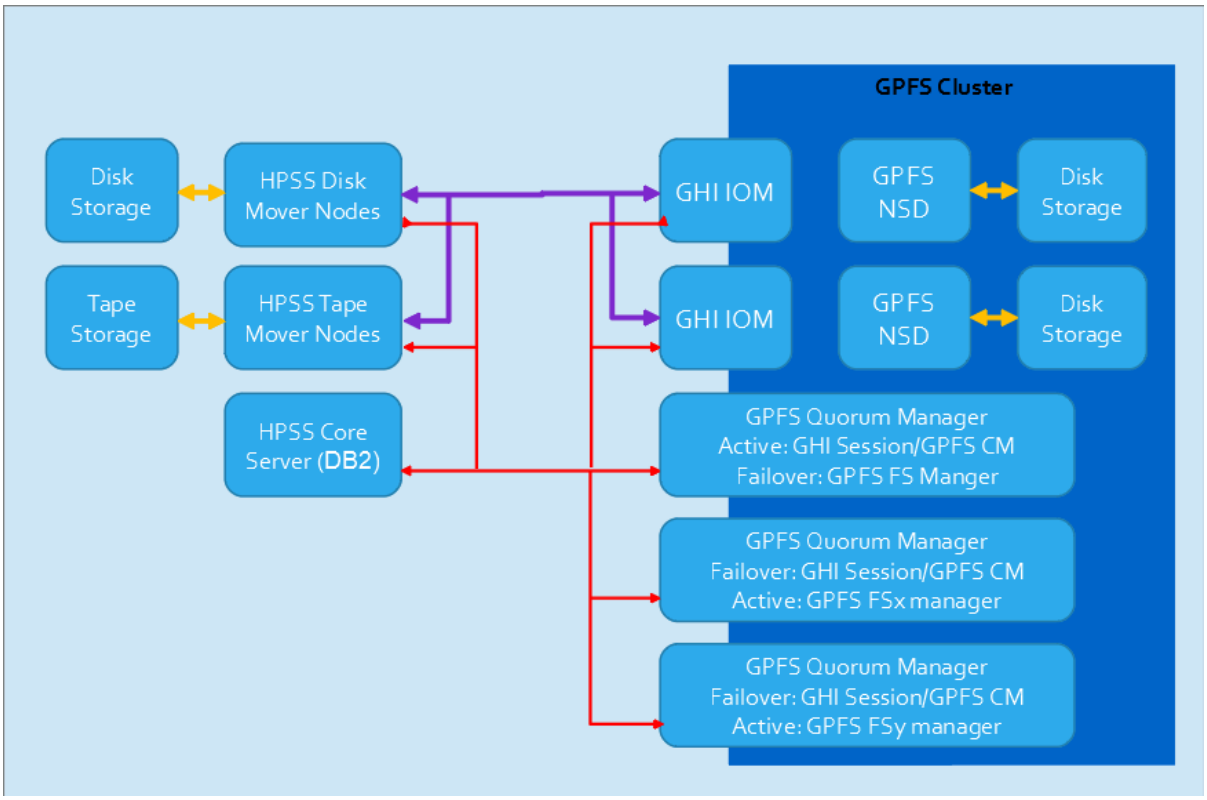
3.3. Network

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 3.5. 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.

3.4. 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**, 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.

Note

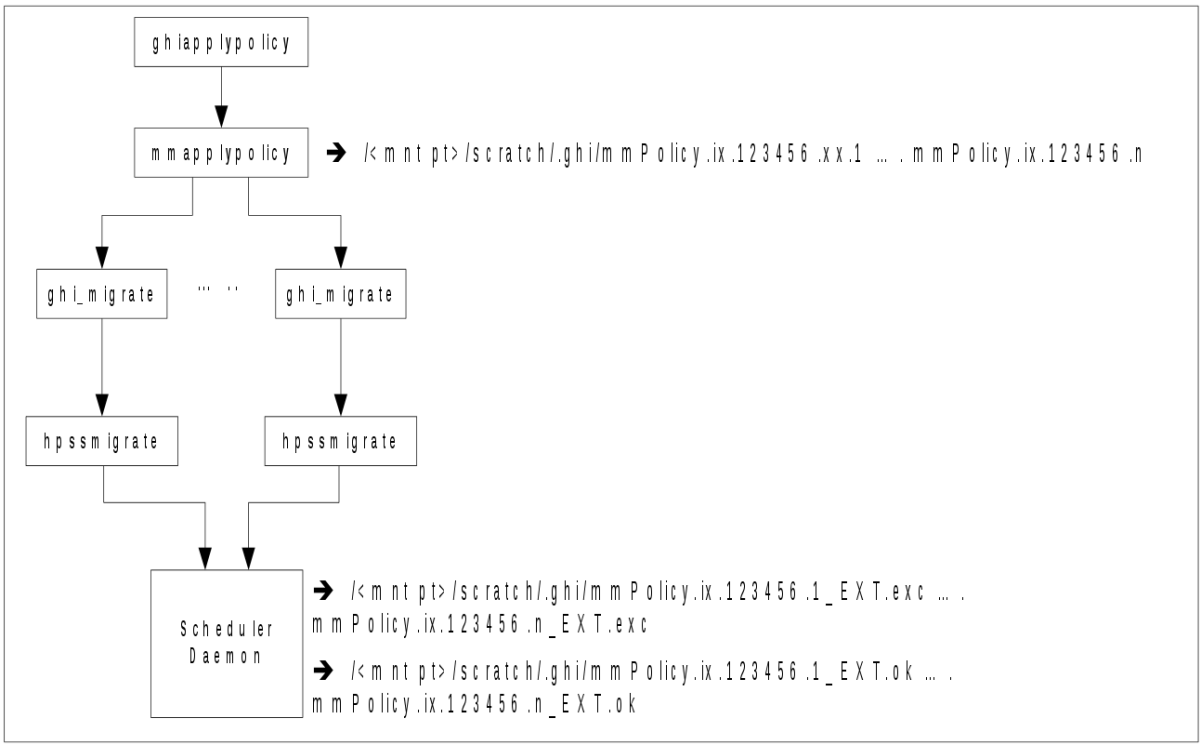
It is important to monitor the ILM policies to ensure they complete without errors. Some errors in the migration process can cause problems, such as the file system filling up or backups not completing.

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** is called to start a migration.

Figure 3.6. 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 4. 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.

4.1. 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.

/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.

> **Note**
>
> 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.

4.2. 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 5. GHI features

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

5.1. 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 restore a file from a previous backup. 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.

To create the mapping table:

1. Create the tablespace - see the *GHI 3.3 Installation Guide* for instructions.

2. Use **ghiCreateMapping** to make a base table for the file system. This is a one-time command to make the base table.

3. Use **ghiLoadMapping** to load all the backups that are GHI 2.5 and greater:

   ```
   % /opt/ghi/bin/ghiLoadMapping <file system> <backup index>
   ```

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**. See the man page for detailed usage information.

After a GHI backup is deleted, the administrator needs to unload the deleted backup from the mapping table with **ghiUnloadMapping**.

```
% /opt/ghi/bin/ghiUnloadMapping <file system> <buidx>
```

The GHI mapping tables will take up space in the GHI database. Plan for the additional space. You can reduce the amount of space the mapping table uses by decreasing the number of valid GHI backups. Load the mapping information only when needed.

5.2. 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** 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**. 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** command for GHI to repack these aggregates. To use this method, the administrator must first create a GHI *xref: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.

5.3. 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.

> **Note**
>
> Prior to running ghi_verify.py for the first time, the HPSS subsystem database must be catalogued. See the GHI Installation Guide for information on cataloging the subsystem database.

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.

5.4. 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
```

5.5. 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.

5.6. ISHTAR

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

5.7. Memcached

Memcached is an open-source, high-performance, distributed memory object caching system intended for use in improving the HPSS Name Server (NS) performance. Some GHI operations require looking up many NS objects, sometimes the same NS object repeatedly in the same operation. **Memcached** will increase the performance of GHI's NS operations while also decreasing the load on the NS itself. The GHI operations include an image backup verification step and **ghi_ls** on a large set of files. Refer to the *GHI Installation Guide* for information on how to install and configure **Memcached**.

5.8. 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** tool is provided for administrators to pin or unpin GHI files. See its man page (`ghi_pin.7`) for details on its use.

It is important to note that running **ghi_pin** 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 6. 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.

6.1. Node failures

6.1.1. 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.

6.1.2. 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.

6.1.3. 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.

6.2. Single process failures

6.2.1. ILM client

The processes *hpssmigrate*, *hpssrecall*, and *hpsslist* are started from the corresponding **ghi_migrate**, **ghi_recall**, 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.

6.2.2. 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.

6.2.3. 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 Guide* and review the **mmlscallback** output.

6.2.4. 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.

6.2.5. 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.

6.2.6. 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.

6.2.7. 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.

6.3. Multiple process failures

6.3.1. 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.

6.3.2. 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.

6.3.3. 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.

6.4. 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**. 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 7. Backup and recovery

7.1. 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** command. The **ghi_backup** 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**, execute:

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

Note

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.

7.2. 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.

The **ghi_restore** utility displays the image backups stored in HPSS.

7.2.1. 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** 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> <…>
   ```

   Note

   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]
   ```

Note

*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.

7.2.2. 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>`

Note

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** or **ghi_stage** 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.

Note

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.

7.3. Managing GHI backups

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

Note

The **ghi_backup_manager** utility should only be run from a GHI session node.

7.3.1. Deleting a backup

To delete a backup:

1. Run the **ghi_backup_manager**:
   `/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.

7.3.2. Resuming deletion of a backup

To resume deletion of a backup:

1. Run the **ghi_backup_manager**:
   `/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.

7.3.3. Validating the checksums of a backup

To validate the checksums of a backup:

1. Run the **ghi_backup_manager**:
   `/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 8. 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

/var/hpss/ghi/etc/memcached.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 Chapter 9 for details on each command.

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.

ghilscluster
List GHI cluster.

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** 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**. The IOM must be deleted and re-added via **ghideliom** and **ghiaddiom**.

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.

8.1. 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)

8.2. 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.

8.3. 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 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.

8.4. 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** and changed with **ghichfs**. 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** 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 --iommaxc)
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 --iomtps)
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 --iomrqs)
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 set list. • **order_TS**. Used by **ghi_stage**; ordering the tape set lists to minimize tape mounts. • **TS_recall**. Used by **ghi_stage**; recalling the sorted 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 2017 *********************
Active Q: 0 W: 0 M: 0/0 R: 2 S: 0 P: 0 D: 0 B: 10 I: 1/1
           F: 0 F: 0/0 F: 0 F: 0 F: 0 F: 0 F:    0

********************* Thu Sep 28 12:50:42 2017 *********************
Active Q: 0 W: 0 M: 0/0 R: 2 S: 0 P: 0 D: 0 B: 10 I: 1/1
           F: 0 F: 0/0 F: 0 F: 0 F: 0 F: 0 F:    0
```

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 (Aggregates/Non-aggregates).
R	The number of staged files from HPSS.
S	The number of recalled files from HPSS.
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 — Under 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.

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.

8.5. 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**. If these settings need to be changed, the entire IOM must be deleted and re-created via **ghideliom** and **ghiaddiom**.

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 9. GHI configuration commands

This chapter provides details of each GHI Configuration Utility command.

9.1. ghilscluster

Lists the GHI cluster-wide configuration.

Usage:

```
ghilscluster [-Hv]
```

Where:

-H
History of the key changes and the dates they were changed.

-v
Verbose.

Example:

```
# ghilscluster

Key        Description             Value (# comment)
 ----------------------------------------------------
--
           GHI Version             3.3
```

9.2. ghichluster

Modifies the cluster-wide configuration. Restart GHI or run **ghi_admin** to reinitialize for the changes to take effect. This command is deprecated starting with GHI 2.5.

There should not be any reason to change the compile code for a GHI cluster.

9.3. ghilsnodes

Lists all GHI nodes. Node[name] and Type are returned by the Spectrum Scale **mmlscluster** command.

Usage:

```
ghilsnodes [-v] [-c <node>]
```

Where:

-c
Display the detailed configuration for the node(s) given by "**<node>**"; that is, for all nodes whose node nodename contains <node>.

-v
Verbose.

Example:

```
# ghilsnodes -c davis

Node                        GPFS Type
 ----------------------------------------
davis.clearlake.ibm.com     quorum-manager
IOM for FS davis-1
```

9.4. ghiaddnode

Add a node to the GHI configuration. The node to be added must already be configured in Spectrum Scale. If you add a node to the Spectrum Scale after the GHI cluster has been created and you want to use that node as a session or IOM node, it must first be added to the GHI cluster with **ghiaddnode**.

Usage:

```
ghiaddnode [-Tv] <node>
```

Where:

-T
Test only; shows all processing steps but will not alter any files or restart any processes.

-v
Verbose.

9.5. ghidelnode

Delete a node from the GHI configuration. A confirmation of the delete request will be presented unless the **-f** option is specified, in which case the deletion will proceed without further user intervention. The node to be deleted must not be configured as an IOM for any file system. Use **ghideliom** to remove the configuration of any configured IOMs. The current GHI session node cannot be deleted.

Usage:

```
ghidelnode [-fTv] <node>
```

Where:

-f
Force; do not output the "Are you sure?" prompt for confirmation.

-T
Test only; do all processing steps but do not alter any files or restart any processes

-v
Verbose.

The following example shows an attempt to delete a node that is still being used by an IOM:

```
% ghidelnode -v fresno
Are you sure you wish to delete this node? (y/n) y
GPFS cluster manager is node1.clearlake.ibm.com
Retrieving from GHI session node -- /var/hpss/ghi/etc/ghi.conf
  scp node1.clearlake.ibm.com:/var/hpss/ghi/etc/ghi.conf /tmp/ghi.conf
Retrieving from GHI session node -- /var/hpss/ghi/etc/ghi_gpfs3fs.conf
  scp node1.clearlake.ibm.com:/var/hpss/ghi/etc/ghi_gpfs3fs.conf
/tmp/ghi_gpfs3fs.conf
node2.clearlake.ibm.com is still referenced within configuration for gpfs3fs!
!!! ABORTING -- UNDOING CHANGES !!!
  TMP_FILE: -- /tmp/ghi_gpfs3fs.conf
  TMP_FILE: -- /tmp/ghi.conf
```

9.6. ghilsfsdefaults

Displays the default configuration for a GHI file system. Once a file system is added to GHI, **ghichfs** can be used to modify the file system configuration. Any of these default configurations can contain a "?xxx" to signify that it serves as a template to be set by **ghiaddfs**. As **ghiaddfs** reads values from the default configuration, it searches the fetched text and replaces every occurrence of "?xxx" with the run-time replacement. "?xxx" is one of:

?FS_Name
The **ghiaddfs** command-line parameter <*FS_name*>.

?Mount_Point
The **ghiaddfs** command-line parameter <*mount_point*>.

?ED_Port
The port number assigned to the SD.

?SD_Port
The port number assigned to the ED.

For example, given a value of "/logs/SD_?FS_Name.log" for configuration item "SD Monitor Output Path", and <FS_name> supplied to **ghiaddfs** as gpfs3fs, the configured "SD Monitor Output Path" for the FS would be /logs/SD_ gpfs3fs.log.

9.7. ghichfsdefaults

Changes the default settings used by GHI when a file system is added to GHI. You can change any of these default configuration items that contain a "?xxx", but the "?xxx" must still be there since it will be populated by the **ghiaddfs** when defaults are used.

Usage:

```
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>] [--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>]
```

Where:

-c
A comment to be associated with the file system.

--junct
The HPSS junction under which all system and user data reside.

--basep
The path in HPSS under which all user data reside.

--bupath
The path in HPSS under which all system data reside.

--maxagg
The maximum number of Spectrum Scale files which GHI will place into a single aggregated HPSS file.

--minagg
The minimum number of Spectrum Scale files which GHI will place into a single aggregated HPSS file.

--aggcos
The HPSS Class of Service into which the associated index file for aggregated files will be placed.

--aggtps
The aggregate thread pool size.

--bbc
The maximum number of Spectrum Scale files which GHI will place into a single GHI backup file.

--bucos
HPSS Class of Service into which the files used to store GHI backups will be placed.

--pblock
Number of Spectrum Scale data blocks to be left resident when a file is purged.

--perf
Performance logging is one or more of the following: ED, ILM, SD, IOM, HTAR, PIO.

--poiot
Purge only if on tape.

--sod
Stage on demand.

--dse
DMAPI stage error number.

--edmaxc
ED maximum connections.

--edtps
ED thread pool size.

--edrqs
ED request queue size.

--edport
Port used by the ED for communicating with its associated SD.

--iomaxc
IOM maximum connections.

--iotps
The IOM thread pool size.

--iorqs
The IOM request queue size.

--iomon
Enable or disable output to the IOM monitor file.

--iomonf
The interval, in seconds, at which the SD writes to the IOM monitor file.

--iomonp
The pathname of the IOM monitor file.

--simaxc
The SD IOM maximum connections.

--sitps
The SD IOM thread pool size.

--sirqs
The SD IOM request queue size.

--sidm
The SD IOM DMAPI thread share.

--sibu
The SD IOM backup thread share.

--simg
The SD IOM migrate thread share.

--sirc
The SD IOM recall thread share.

--sirs
The SD IOM restore thread share.

--siad
The SD IOM admin thread share.

--scmaxc
The SD Client maximum connections.

--sctps
The SD client thread pool size.

--scrqs
The SD client request queue size.

--sdport
The port used in communicating with the SD to request and monitor file transfers.

--sdmon
Enable or disable output to the SD monitor file.

--sdmonf
The interval, in seconds, at which the SD writes to the SD monitor file.

--sdmonp
The pathname of the SD monitor file.

For example, given a value of "/logs/SD_?FS_Name.log" for configuration item "SD Monitor Output Path" and <FS_name> supplied to ghiaddfs as "gpfs3fs", the configured "SD Monitor Output Path" for the FS would be "/logs/SD_ gpfs3fs.log". The "# <comment>" is comment text which will be applied to the "?FS_Name" whenever the **ghiaddfs** command is executed.

9.8. ghichlog

Changes log levels for GHI processes.

Usage:

```
ghichlog -l add|change|delete {loglevel1[,loglevel2,...,loglevelN]|ALL}
         [-T] [-v] [-a] [-p] [-m] [-c] [-t]
         [-f filesystem] [-s] [-e] [-q] [-i iom name] [-x]
```

Where:

-l
This required argument is used to specify an action followed by one or more comma-separated log levels. The actions are:

- *add* - adds the log levels specified
- *change* - changes the log levels to what is specified

- *delete* - removes the log levels specified.

See Logging levels for a list and description of the different log levels. The log levels should be specified without spaces and may be in any order and are case-insensitive. The default log levels are ALERT, CRIT, and NOTICE. Note that the special log level *ALL* should be specified by itself and has the following behaviors for the specified processes:

- *ALL* with an action of *delete* changes the log levels to ALERT and CRIT only.
- *ALL* with an action of *add* or *change* adds all log levels.

Note

The log levels of ALERT and CRIT may not be removed.

-T
Test only; do all processing steps required, but do not alter any log levels.

-v
Verbose.

-a
Apply the specified log levels to all processes.

-p
Change the process manager's log settings.

-m
Change the mount daemons's log settings.

-c
Change the configuration manager's log settings.

-t
Change the tool's log settings.

-f
The name of the GHI <*file system*> to be modified must be specified if the Scheduler, Event Daemon, Policy, or IOM settings are going to be changed.

-s
Change the scheduler's log settings. The file system must be specified with **-f**.

-e
Change the event daemon's log settings. The file system must be specified with **-f**.

-q
Change the policy's log settings. The file system must be specified with **-f**.

-i
Change the <*iom name*> IOM's log settings. The IOM 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.

Examples:

Adds DEBUG, THREAD, and INFO log levels to the scheduler daemon process for file system demofs:

```
# ghichlog -l add DEBUG,INFO,THREAD -f demofs -s
```

Deletes DEBUG, QUEUE, and WARN from the log levels for processes process manager, mount daemon, configuration manager, and tools:

```
# ghichlog -l delete DEBUG,QUEUE,WARN -p -m -c -t
```

Changes the log levels for all processes to be just ALERT and CRIT:

```
# ghichlog -l delete all -a
```

Adds all log levels to all processes:

```
# ghichlog -l add all -a
```

Changes the log levels to ALERT, CRIT, and NOTICE for the IOM named demofs1 on port 8012 for file system demofs:

```
# ghichlog -l change ALERT,CRIT,NOTICE -f demofs -i demofs1:8012
```

Adds TXN log level to the Scheduler, Event Daemon, Policy, and IOM processes for the file system demofs.

```
# ghichlog -f demofs -x
```

9.9. ghilsfs

Displays the configuration for a specific file system. If **ghilsfs** is run with no arguments, then it will return a complete list of GHI-managed file systems. If a key value is given, **ghilsfs** will only list the single key.

Usage:

```
ghilsfs [-Hv] [<file_system> ...] [<key> ...]
```

Where:

-H
The history of the key changes and the dates they were changed.

-v
Verbose.

<file_system>
The name of a file system; only the specified FS(s) will be displayed.

<key>
A key from the list of keys below; only the specified key(s) will be displayed.

Example:

```
# ghilsfs davis-1

Key          Description                   Value (# comment)
 ------------------------------------------------------------------------
--
             File System Name              davis-1
--mp         Mount Point                   /davis-1
--uid        Unique Identifier             1
--junct      HPSS Junction                 /
--basep      HPSS Base Path                /ghi
--bupath     HPSS Backup Path              /ghi
--maxagg     Max Files Per Aggr          * 5000
--minagg     Min Files To Make Aggr      * 200
--aggcos     Aggr Index COS                1
--aggtps     Aggr Thread Pool Size         40
--bbc        Backup Bulk Count           * 10000
--bucos      Backup COS                    1
--pblock     Purged File Size              0
--perf       Performance logging           NONE
--poiot      Purge Only If On Tape       * FALSE
--sod        Stage On Demand               ON
--dse        DMAPI Stage Errno             7500
--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           5
--iotps      IOM Thread Pool Size        * 15
--iorqs      IOM Request Queue Size        10
--iomon      IOM Monitor Flag            * ON
--iomonf     IOM Monitor Frequency       * 30
--iomonp     IOM Monitor Output Path       /davis-1/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     500
--sctps      SD Client Thread Pool Size    100
--scrqs      SD Client Request Queue Size  400
--sdport     SD Port Number                8010
--sdmon      SD Monitor Flag             * ON
--sdmonf     SD Monitor Frequency        * 30
--sdmonp     SD Monitor Output Path        /davis-1/scratch/mon/mon_sd.out
--fstype     FS Type                       full-access
--bustat     Backup Status                 last good backup 10
--afs        Associated FS                 none
```

9.10. ghiaddfs

Adds a file system to the GHI configuration. The file system name and mount point must be unique among all GHI clusters connected to the HPSS system. The default SD and ED ports are 80x0 for the SD and 80x1 for the ED, where "x" is the order in which file systems were configured. For example, 8010 and 8011 are for the first configured file system, 8020 and 8021 for the second configured file system, and so on. The actual configured port numbers will be the first available ports starting with the default. The file system to be added must be known to Spectrum Scale and not mounted. GHI assumes the file system is DMAPI-enabled. Without the **-r** argument, the file system will be full-access. To configure a read-only file system, include the **-r** argument. The read_only_target must have already been configured and may reside on the same or a different GHI cluster.

Usage:

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

Where:

-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 the steps required, but do not alter any files or Db2 tables, or restart any processes.

-c
Comment.

-v
Verbose.

-r
Create the file system as a read-only version of the full-access file system, which must already be configured and not itself be a read-only file system. The read-only file system may reside on the same or a separate cluster as the full-access file system.

<file_system>
The name of the Spectrum Scale file system to add to the GHI configuration. This must match the Spectrum Scale configuration and be unique among all GHI clusters which connect to the same HPSS.

<mount_point>
The mount point for the Spectrum Scale file system. This *must* match the Spectrum Scale configuration and be unique among all GHI clusters which connect to the same HPSS.

<SD_port>
The port the **ghi_sd** should use for the file system. GHI will assign a port if one is not specified.

<ED_port>
The port the **ghi_ed** should use for the file system. GHI will assign a port if one is not specified.

Example:

```
# ghiaddfs -k davis-1 /davis-1
Updating DB2 table GHI_FS_LIST...
Using existing DB2 BU/GC tables as-is...
Distributing updated GHI config to all GHI nodes...
Distributing updated GHI config to all GHI nodes...
Creating FS policy files on all GHI nodes...
Alerting the GHI PM and MD to the new FS...
Done.

Key        Description                Value (# comment)
 ------------------------------------------------------------------------
--
           File System Name           davis-1
```

9.11. ghichfs

Changes the configuration of a file system. Neither the file system's name nor its mount point can be modified. To change the file system name or mount point, the file system name must be deleted and re-created via **ghidelfs** and **ghiaddfs**. All other configuration items can be modified.

Note

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 in section GHI file system configuration.

Usage:

```
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>]
[--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>]
```

Where:

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

-v
Verbose mode.

-c
A user comment.

--junct
The HPSS junction name under which all system and user data will reside.

--basep
The path in HPSS under which all user data will reside.

--bupath
The path in HPSS under which all system data will reside.

--maxagg
The maximum number of Spectrum Scale files which GHI will place into a single aggregated HPSS file.

--minagg
The minimum number of Spectrum Scale files which GHI will place into a single aggregated HPSS file.

--aggcos
The HPSS class of service into which the associated index file for aggregated files will be placed.

--aggtps
The aggregate thread pool size.

--bbc
The maximum number of Spectrum Scale files which GHI will place into a single GHI backup file.

--bucos
The HPSS class of service into which the files used to store GHI backups will be placed.

--pblock
The number of Spectrum Scale data blocks to be left GPFS-resident when a file is purged.

--perf
Performance logging is one or more of the following: ED, ILM, SD, IOM, HTAR, or PIO.

--poiot
Purge Only If on Tape.

--sod
Stage on Demand.

--dse
The DMAPI Stage Error number.

--edmaxc
The ED max connections.

--edtps
The ED thread pool size.

--edrqs
The ED request queue size.

--edport
The port used by the ED for communicating with its associated SD.

--iomaxc
The IOM max connections.

--iotps
The IOM thread pool size.

--iorqs
The IOM request queue size.

--iomon
Enable/disable output to the IOM monitor file.

--iomonf
The interval, in seconds, at which the SD writes to the IOM monitor file.

--iomonp
The pathname of the IOM monitor file.

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

--simaxc
The SD IOM max connections.

--sitps
The SD IOM thread pool size.

--sirqs
The SD IOM request queue size.

--sidm
The SD IOM DMAPI thread share.

--sibu
The SD IOM backup thread share.

--simg
The SD IOM migrate thread share.

--sirc
The SD IOM recall thread share.

--sirs
The SD IOM restore thread share.

--siad
The SD IOM admin thread share.

--scmaxc
The SD client max connections.

--sctps
The SD client thread pool size.

--scrqs
The SD client request queue size.

--sdport
The port used in communicating with the SD to request and monitor file transfers.

--sdmon
Enable/disable output to the SD monitor file.

--sdmonf
The interval, in seconds, at which the SD writes to the SD monitor file.

--sdmonp
The path name of the SD monitor file.

Example:

```
# ghichfs davis-1 --sdmonf 60
    Key         Description                 Value (# comment)
    ----------------------------------------------------------------------
    --sdmonf    SD Monitor Frequency        60
    Distributing updated GHI config to all GHI nodes...
Done.
```

9.12. ghidelfs

Deletes a file system from the GHI configuration. A confirmation of the delete request is presented unless the **-f** option is specified, in which case the deletion will proceed without further user intervention. The file system to be deleted must be unmounted and not have any IOMs configured for it. Use **ghideliom** to remove the configuration of any configured IOMs for the file system.

Usage:

```
ghidelfs [-fTv] <file_system>
```

Where:

-f
Force; do not ask "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 mode.

Example:

```
# ghidelfs davis-1
Are you sure you wish to delete this FS? (y/n) y
Updating DB2 tables...
Distributing updated GHI config to all GHI nodes...
Distributing updated GHI config to all GHI nodes...
Deleting associated GHI FS config data from all GHI nodes...
Alerting the GHI PM and MD to the deleted FS...
Done.

Looking for FS davis-1 in GHI configuration...

FS davis-1 not found in GHI configuration
```

9.13. ghilsiom

Lists the complete configuration of an individual IOM or all IOMs for a file system.

Usage:

```
ghilsiom [-Htv] <file_system> [<IOM> ...]
```

Where:

-H
Display the history of the key changes and the dates they were changed.

-t
Alternate format.

-v
Verbose mode.

Example:

```
# ghilsiom davis-1

Key        Description               Value (# comment)
 ------------------------------------------------------------------------
           IOM Node:Port             davis.clearlake.ibm.com:8012
--asn      Active Session Node       TRUE
--etr      Estimated Transfer Rate   10GB
[root@davis ~]#
```

9.14. ghiaddiom

Adds an IOM to the GHI configuration. The scheduler daemon will also be restarted to make use of the updated IOM configuration.

Usage:

```
ghiaddiom [-RTv] <file_system> [-c "# <comment>"]
<IOM_node>[:<port>] <Active_on_session_node> <Est_XFER_rate>
```

Where:

-R
Push configuration changes to all GHI nodes and restart the GHI SD. If this 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.

-c
"# comment" – 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? Possible 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".

Example:

```
# ghiaddiom davis-1 davis:8012 True 1GB
No IOMs currently configured for this FS.
Configuring IOM system services for davis.clearlake.ibm.com:8012
Distributing updated GHI FS config to all GHI nodes...
Distributing updated GHI config to all GHI nodes...
Not restarting the SD.
The new IOM will not be used until the SD gets restarted.
Done.

Key      Description               Value (# comment)
 ------------------------------------------------------------------------
         IOM Node:Port             davis.clearlake.ibm.com:8012
--asn    Active Session Node       TRUE
--etr    Estimated Transfer Rate   1GB
```

9.15. ghichiom

Updates the configuration for an IOM. To change the IOM's node or port, the IOM must be deleted and re-created via **ghideliom** and **ghiaddiom**. All other configuration items can be modified.

Usage:

```
ghichiom [-Tv] <file_system> <IOM_node>:<port> [-c "# <comment>"]
[--asn {TRUE|FALSE}] [--etr <rate>] [--chunksize <bytes>] [--sort]
```

Where:

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

-v
Verbose.

<file_system>
The GHI managed file system name.

<IOM_node>:<port>
The IOM to be updated.

-c "# <comment>"
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
Maximum number of bytes to transfer per non-aggregate HPSS I/O request. Append "KB", "MB", "GB", or "TB" as a units multiplier.

--sort
Sort the list of IOMs alphabetically for the file system.

Example:

```
# ghichiom davis-1 davis:8012 --asn FALSE
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       FALSE
--etr       Estimated Transfer Rate   10GB
--chunksize Transfer Chunk Size       1TB
```

9.16. ghideliom

Deletes an IOM from the GHI configuration. A confirmation of the delete request will be presented unless the **-f** option is specified, in which case the deletion will proceed without further user intervention. The scheduler daemon will need to be restarted to make use of the updated IOM configuration unless the **-R** option is used.

Usage:

```
ghideliom [-RfTv] <file_system> <IOM_node>:<port>
```

Where:

-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; 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 "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 mode.

<file_system>
The file system associated with the IOM to be deleted.

<IOM_node>:<port>
The IOM to be deleted.

Example:

```
# ghideliom davis-1 davis:8012
Are you sure you wish to delete this IOM? (y/n) y
De-configuring IOM system services for davis.clearlake.ibm.com:8012
Deleting associated GHI FS config data from all GHI nodes...
Distributing updated GHI config to all GHI nodes...
Not restarting the SD.
The SD will attempt to use the IOM until the SD gets restarted.
Done.

Looking for IOM davis.clearlake.ibm.com:8012 in FS configuration...
No IOMs are configured for this FS.
```

9.17. 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. This tool should be run as root.

Usage:

```
ghi_dump_fs_config_logging [-l logrotate_dir] [-m max_age]
[-o output_dir] [-r rotate] [-h] file_system
```

Where:

-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>/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

file_system

The name of the file system that will be dumped with the **ghi_dump_fs** tool. This must be the last parameter.

Example:

```
# ghi_dump_fs_config_logging rockfs5
Copying the log file template
/var/hpss/ghi/config/templates/ghi_dump_fs_log_config.template to
/etc/logrotate.d/ghi_dump_fs_rockfs5.
Updating the logrotate configuration file with the output directory location of
the ghi_dump_fs logs.
Updating the logrotate configuration file with the rotate value.
```

Chapter 10. GHI tools

10.1. ghi_dump_fs

Allows you to list the contents of a GHI managed file system for inspection or future disaster recovery purposes. The tool will create two output files: `ghi_dump_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.

> **Note**
>
> 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 Installation Guide* for more information.

See the **ghi_dump_fs** man page for information on tuning the performance.

Usage:

```
ghi_dump_fs.py [-h] [-a] [-c GPFS_CONFIG_FILE] [-E] [-H]
[-o OUTPUT_DIR] [-p START_PATH] [-i] [-n] [-t THREAD_COUNT]
[-v] [-x] file_system
```

Where:

-h
Show the help message and exit.

-a
Include hidden files/directories (names which begin with a .).

-c
Set the path to the GPFS Config File to <GPFS_CONFIG_FILE>.

-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.

-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
The directory to store the output files from this tool. (Default: `/<file_system_mount_point>/scratch/.ghi`)

-p
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
Specifies the <THREAD_COUNT> indicating the maximum number of threads to run. (Default: `10`)

-v
Displays the SOID version for files which reside in HPSS.

-x
During the listing process also convert SOIDs to latest version for files which reside in HPSS.

file_system
The name of the file system.

Example:

```
ghi_dump_fs.py rockfs1
```

The output below shows a sample record in the output file when the tool is executed with no optional arguments.

```
B  -rw-r--r--   1  root      root          1024 Sep 19 13:50 /rockfs1/test/fil
e.aaeg MIDX:3 COS:3 FF:0 L0-DISK:KE000100:102913536  Ord:N/A //ghi/rockfs1/202
3/09/46241/AGG.b2f2798e-9a61-4848-a8e8-c8bd84a52b02
```

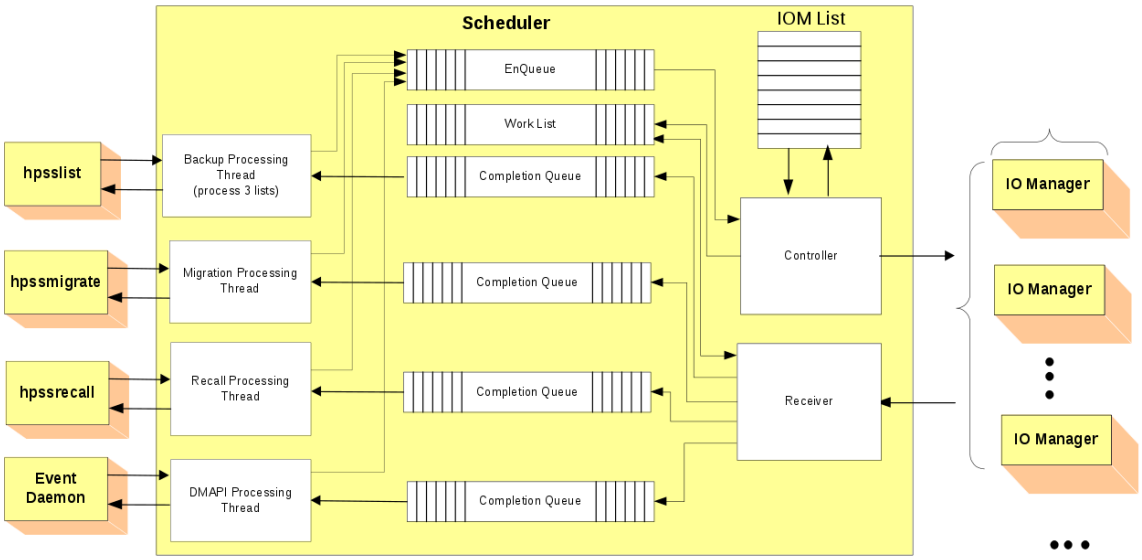
Chapter 11. System monitoring

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.

11.1. Scheduler

The figure depicts the internals of the scheduler.

Figure 11.1. 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.

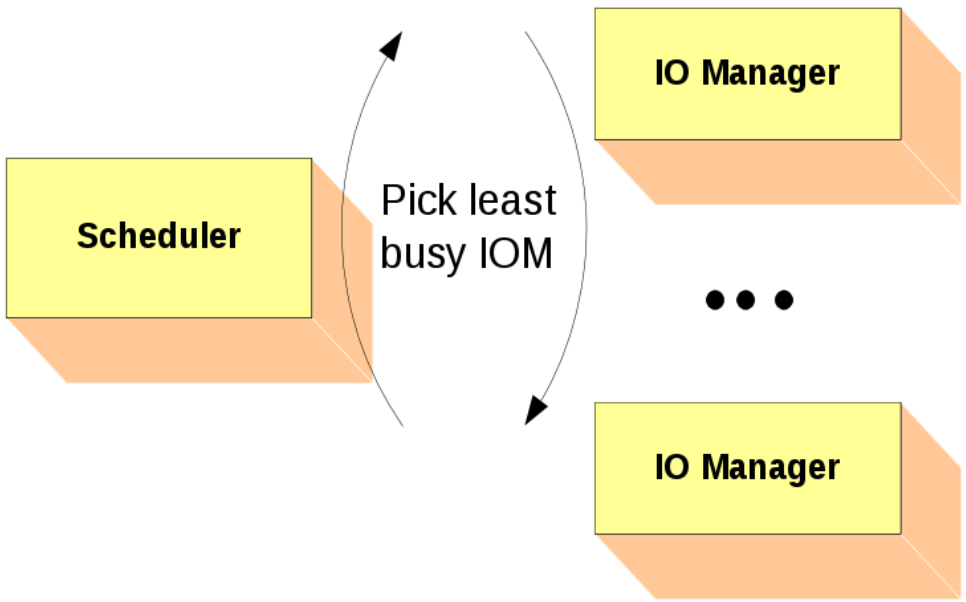
Failed
Total number of migrations, recalls, stages, or purges that have failed.

IOM(A/T)
Total number of IOMs that are active or total number of IOMs configured.

11.2. I/O manager

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

Figure 11.2. 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.

11.3. 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** 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** 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.

11.3.1. 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**) 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.

11.3.2. 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 12. 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.

12.1. Common infrastructure (communication) problems

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

12.1.1. 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.

12.1.2. 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.

12.1.3. 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.

12.1.4. 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.

12.2. Server problems

12.2.1. 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`.

12.2.2. 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.

12.2.3. 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.

12.2.4. 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.

12.2.5. 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.

12.2.6. 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.

12.2.7. 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.

12.2.8. 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.

12.2.9. 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.

12.2.10. 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.

12.2.11. 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.

12.2.12. 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.

12.2.13. 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.

12.2.14. 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.

12.2.15. IOM is in STANDBY mode

Diagnosis: IOM loses connection to the SD.

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

12.2.16. 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.

12.2.17. 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.

12.2.18. 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.

12.2.19. 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.

12.2.20. 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.

12.2.21. 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*.

12.2.22. 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.

12.2.23. 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*.

12.2.24. 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.

12.2.25. 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.

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

Diagnosis: The xinetd was configured incorrectly.

Resolution: See the IOM problems above.

12.2.27. 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**. Also, verify the file system is configured with the **ghilsfs** 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**. Also, verify the file system is configured by executing the **ghilsfs** 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
```

12.2.28. 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%'
```

12.2.29. 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**).

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.

12.2.30. 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.

12.2.31. 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.

12.2.32. 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.

12.2.33. 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.

12.2.34. 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.

12.2.35. 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.

12.2.36. 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.

12.2.37. 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.

12.2.38. 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.

12.2.39. 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.

12.2.40. 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 *GHI Installation Guide* for how to set up the **ulimit** settings.

12.2.41. 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 Installation Guide, Release 7.5.3**
2. **HPSS Management Guide Release 7.5.3**
3. **HPSS Error Manual Release 7.5.3**
4. **HPSS User's Guide Release 7.5.3**
5. **Spectrum Scale Data Management API Guide**
6. **Spectrum Scale Administration and Programming Reference**
7. **Spectrum Scale Advanced Administration**
8. **POSIX 1003.1-1990 Tar Standard**