GHI Management Guide

GPFS / HPSS Management Guide
Release 2.4

March 2014

Portions of this work were produced by Lawrence Livermore National Security, LLC, Lawrence Livermore National Laboratory (LLNL) under Contract No. DE-AC52-07NA27344 with the U.S. Department of Energy (DOE); by the University of California, Lawrence Berkeley National Laboratory (LBNL) under Contract No. DE-AC02-05CH11231 with DOE; by Los Alamos National Security, LLC, Los Alamos National Laboratory (LANL) under Contract No. DE-AC52-06NA25396 with DOE; by Sandia Corporation, Sandia National Laboratories (SNL) under Contract No. DE-AC04-94AL85000 with DOE; and by UT-Battelle, Oak Ridge National Laboratory (ORNL) under Contract No. DE-AC05-00OR22725 with DOE. The U.S. Government has certain reserved rights under its prime contracts with the Laboratories.

Printed in the United States of America.

GHI 2.4
March 2014

TABLE OF CONTENTS

LIST OF FIGURES

Preface

The GHI Management Guide is intended as a resource for HPSS and site administrators. This document provides chapters that contain the details for monitoring and managing a GHI system.

1. GHI BASICS

1.1. Introduction

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

1.2. GHI Capabilities

Both GPFS 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 a large number of files in a file system, and these may be any mixture of sizes from very small to very large. Both GPFS 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. The 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 GPFS. GHI uses and extends GPFS ILM capabilities, providing a cost-efficient integrated storage solution that is scalable to 100s of petabytes and billions of files. GHI enables GPFS file data transfers between GPFS high performance storage, usually high-speed disk, and HPSS cost-efficient storage, usually high capacity disk and tape. This movement between GPFS and HPSS occurs automatically under control of GPFS ILM rules and the DMAPI framework within GPFS, thus providing a complete and scalable HSM and backup solution that exploits HPSS parallel file system capabilities. In order to accomplish these goals, GHI is designed and implemented based on the concepts described in the following subsections.

1.2.1. Network-centered Architecture

The focus of the GHI feature of HPSS is the network. GPFS 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.

1.2.2. High Data Transfer Rate

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 I/O Manager 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.
For small GPFS file data transfers, GHI uses a modified GHI-specific version of the HTAR program, known as "GHI-HTAR". GHI-HTAR is used for aggregating a set of files from GPFS directly into HPSS. It uses a multi-threaded buffering scheme to write files directly into HPSS, thereby achieving a high rate of performance.

1.2.3. Standard Components

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 GPFS backups.
The GHI system is supported on IBM AIX and RedHat LINUX platforms.

1.2.4. DB2 Components

GHI uses DB2 to maintain a history of the GHI backups and to facilitate the implementation of GHI read-only file systems. The DB2 Server configured on HPSS is used to store the backup tables for GHI. There are two tables configured in the GHI database on the HPSS Core Server for each GHI managed GPFS file system to support GHI backups and one additional table used to store information about every GHI managed GPFS file system. The GHI Session nodes will be configured as DB2 clients to access the backup tables during GHI backup and restore operations. The file system information table is used to determine whether user data on a particular file system is to be full-access or read-only with respect to HPSS.

1.2.5. Terminology

Some of the following terms are overloaded, meaning GPFS and HPSS have different meanings for the same term. GHI uses the HPSS terminology.

- ***Backup*** - Backup refers to backing up a GPFS file system into HPSS. The information needed to restore a GPFS file system, including the restoration of the GPFS 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 the purpose of sharing resources and communicating with each other.
- ***Garbage Collection*** - This involves removing GHI files from HPSS that are no longer referenced by GPFS or a valid backup. (See Section *1.2.6.4 Garbage Collection and GPFS File Deletion)*
- ***Migration*** - Migration refers to the movement of file data from GPFS to

HPSS, while maintaining the file data in GPFS. There are two scenarios where migrations are performed. The first scenario is when the GPFS policy engine is run to transfer file copies from GPFS 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 a policy triggered HSM migration are copied to HPSS. The GPFS term is "pre-migration".

- ***Purge*** – Purge refers to freeing up data segments in the GPFS file to free up GPFS resources. A GPFS policy is used to trigger a threshold request. The data blocks for the selected files are freed, leaving a stub in GPFS. The GPFS term is "punching a hole".
- ***Recall*** - Recall refers to the movement of file data from HPSS to GPFS. The GPFS term is "pre-stage".
- ***Restore*** – Restore refers to the capability to restore either a GPFS file system or a GPFS cluster from a selected backup.
- ***Stage*** - Stage refers to the movement of file data from HPSS to GPFS. 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. A response is sent to the user when then operation is complete. This is sometimes referred to as "stage on-demand".
- ***Pin*** – Pin or pinning refers to flagging a file so that it can not be purged from the GPFS file system.
- ***GHI Metadata –*** Any data necessary to reconstruct the GPFS file system namespace. Metadata consists of GHI database, GHI backup files and aggregated index files.
- ***GHI User Data –*** User data consists of GPFS data files and GPFS data file attributes. The file attributes consist of UID, GHI, access time, modified time, DMAPI attributes, symbolic/hard link information.
- ***Full-Access -.*** File systems will normally be full-access, Read-Only (i.e. user files may be migrated into HPSS) a GHI backup may be taken, and migrated files may be deleted from GPFS to cause GHI to do garbage collection within HPSS. A read-only file system is one that is created and associated with a full-access FS and populated by restoring a GHI backup of the full-access FS. Restored files on a read-only FS may be recalled or staged from HPSS, purged from GHI, but they may not be modified or deleted. New files may be created, modified, and deleted but they may not be migrated into HPSS. In short, no file-related actions on a read-only FS will result in any addition of data to or deletion of data from HPSS. Read-only file systems were created to allow validation of GHI backups prior to being needed for disaster recovery. They may also be used to retrieve files from a backup without affecting the full-access FS or the status of backups.

1.2.6. HSM Concept of Operation

GHI uses the GPFS ILM policy driven storage management to efficiently provide

migration and staging of GPFS files through tiered storage. To copy file data from GPFS to HPSS storage, the GPFS policy engine is used. GHI supports the following ILM mechanisms to transfer data between GPFS and HPSS, as well as manage available space in the GPFS file system:

- Data Migration
- Data Recall
- File system limits
- Garbage Collection

GHI also uses the GPFS DMAPI interface to stage data back from HPSS on-demand when a user requests access to the GPFS file data (i.e. open, chksum, etc).

1.2.6.1. Transferring data into HPSS

Files are transferred (i.e. migrated) from GPFS into HPSS based on rules sent to the GPFS policy engine. A migration policy is used to provide a set of rules to determine which GPFS files are candidates to be transferred 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're transferred into HPSS. The other list contains non-aggregate files, which will be individually transferred to/from and maintained within HPSS. Next, the policy engine invokes the following GHI script:
***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 will receive 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 faction of "aggregate bulk size".
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: The template provides rules to split files to be migrated into two categories: aggregates and non-aggregates.
Any attempt to migrate files from a GHI read-only file system will be rejected. The rejection code will be -1 (Operation not permitted).

1.2.6.2. Transferring data from HPSS

Files are transferred from HPSS to GPFS based on two scenarios:
Recall Operations: Recall files from HPSS as a background or scheduled task. Either GPFS policy rules or a list of files/directories can be defined to retrieve the file data in advance of a user request to access those files. Policy rules instruct GPFS on how to create a list of files that are candidates that are eligible for recalling back from HPSS. GHI will sort the list of files (from either a policy run or a user-supplied list) based on location in HPSS. Files that reside on the same tape will be recalled together to minimize tape mounts. Files that reside in the same aggregate will be recalled together, using a single GHI-HTAR request.
Stage Operations: Stage files back from HPSS synchronously when file contents are accessed by a user or a program.

1.2.6.2.1 Recall Operations

The recall policy provides a set of rules that are used to determine which files are to be copied from HPSS to GPFS. The GPFS policy engine generates one list of files to be recalled and then invokes the following GHI script:

- ***ghi_recall***: One instance of the script is invoked by the policy engine to coordinate with the GHI scheduler to recall the files from HPSS. The scripts parses through the list and generates buckets 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.

GHI provides a recall template in the ***/var/hpss/ghi/policy*** directory as an example of how to generate a list of files to be recalled:

- ***recall.policy***: The template provides rules to split files to be migrated into two categories: aggregates and non-aggregates.

The ***ghi_stage*** command is the alternative to recalling using a policy. It accepts either a list of files/directories or a list of file lists. If the list is of files/directories, ***ghi_stage*** will recall the listed files and the [possibly recursive] contents of the listed directories. If the list is of file lists is provided, ***ghi_stage*** will recall the files named in the file list(s).

1.2.6.2.2 Stage Operations

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 GPFS file is performed differently depending on where the data resides.
If a file resides in both GPFS and HPSS, a WRITE or TRUNCATE event is generated when the user updates the file. This does not cause the file to be staged, since it still reside 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 file in HPSS will become a candidate for garbage collection.
If a file only exists in HPSS, which means that no data resides in GPFS, a READ event is generated when the user opens the file in GPFS. This causes the file to be staged from HPSS and become dual resident. If the file is then modified, a WRITE or TRUNCATE event will be generated and processed as just stated.

1.2.6.3. **Managing Available Space**

To monitor the high and low water marks for a GPFS file system, the file system is enabled by attaching a set of policies rules to the file system. When the system triggers a high (NO_SPACE) or low (LOW_SPACE) event, the GPFS policy engine generates a list of files to be purged from the file system.
The following GHI provided script is invoked when the event is triggered:
ghi_migrate: When used with the '-P' option, the script sends the list of files to be purged to the GHI Scheduler Daemon.
GHI provides a threshold template in the ***/var/hpss/ghi/policy*** directory as an example of rules to generate a list of files to be purged:

threshold.policy: The template provides rules to generate purge candidates.

1.2.6.4. Garbage Collection and GPFS File Deletion

GHI Garbage collection is the removal of unreferenced GHI files from HPSS. Unreferenced files are GHI files in HPSS that no longer exist in GPFS and are not referenced by a backup of the GPFS file system. Files are not referenced when:

- They have not been migrated into HPSS.
- They are not part of a valid backup

GPFS will notify GHI when a GHI-managed file is deleted or updated. GHI will place each notification into a DB2 garbage collection (GC) table.
Whenever the GHI backup manager is executed to delete a backup, notifications 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 must retain the notification in the GC table until all referencing backup are deleted.

1.2.7. Concept of Operation for Backup

Starting with version 2.4, GHI includes support for the GPFS Scale-Out Backup and Restore (SOBAR) feature, a.k.a., "image backup". In order to allow restoration from backups taken with prior versions of GHI, support for restoration from that type of backup has been maintained. GHI will automatically determine the type of backup being restored.

The ability to backup a GPFS file system is accomplished using the GHI ***ghi_backup*** interface. The backup interface uses the GPFS 'mmimgbackup' command and the GPFS mmimgbackup command uses the ILM policy management interface. Each file system to be backed-up uses its own copy of each of the following backup policy templates that resides in the ***/var/hpss/ghi/policy*** directory:

- **Backup_migration.policy**: The backup migration policy contains the migration rules for the GPFS file system being backed up. The rules can migrate files as aggregate or non aggregates. The rules should select all the files to be backed up.
- **Backup_metadata.policy**: The backup metadata policy contains the rules that previous GHI releases backups need to capture the file system's metadata. The new image backup feature supported on this release does not require a metadata policy run for metadata backup. The metadata is contained in the image generated by GPFS as part of the image backup process.
- **Backup_error.policy**: The backup error policy contains the rules that will

be used to validate capture of the file system's metadata.

GHI backups use the GPFS snapshot feature to take a point-in-time image of the file system. When running the backup process, a snapshot of the GPFS 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 metadata will appear in the snapshot or the resulting backup. (If the file still exists at the next backup, the update(s) will appear in it.)

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

1.2.7.1. Restore Concept of Operation

GHI provides a restore utility, ***ghi_restore***, to rebuild a GPFS file system after a catastrophic failure. The GHI restore utility allows the administrator to display the full backups stored in HPSS, and if available for the selected backup, each of the incremental backups associated with a full backup. (Incremental backups are only applicable to pre-GHI 2.4 backups.) For restoring a file system, the restore process is broken into three phases:

- Restore of the namespace (directories, filenames, hard links, and symbolic links) and associated attributes (owner, permissions, etc). This is accomplished via the GPFS 'mmimgrestore' command.
- Mark future backups as invalid and mark files associated only with those backups as orphans which may be garbage-collected.
- Recall of file data resident on the file system when the backup was taken. This step is a separate procedure invoked by the administrator or will be performed on-demand on a per-file basis if a user should attempt to access a file.

Rebuilding the GPFS namespace and associated attributes is a process that is relatively fast. When complete, administrations should define recall rules to stage file data back from HPSS. If users are allowed onto the system prior to all file data being recalled from HPSS and they attempt to access a file(s), the file(s) will be synchronously recalled from HPSS via the GPFS DMAPI interface.

ghi_restore can also be used to restore a backup to a GHI read-only file system. This can be done to validate a backup or to retrieve files from a backup that have been deleted from GPFS or for comparison with the current version. Once the backup has been restored to a read-only FS, it can be compared with the associated full-access FS to ensure that the validity of the backup. Restored files can also be copied to the full-access FS to either restore deleted files or to restore previous versions of a file, either directly to the file or under another name.

1.2.8. Storage in HPSS of GPFS Files

The following subsections provide information on GHI object mappings and describe how GHI uses the mapping information.

1.2.8.1. Extended Attributes

To map GPFS file system objects to an HPSS object, mapping information is stored in the GPFS extended attributes. This information contains:
HPSS Identifier – An unique identifier to locate where the GPFS file contents are archived in HPSS.
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 GPFS 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 contents.
A separate DMAPI attribute contains the flag that indicates a file has been pinned.

1.2.8.2. Location of files in HPSS

1.2.8.2.1 Migrated files

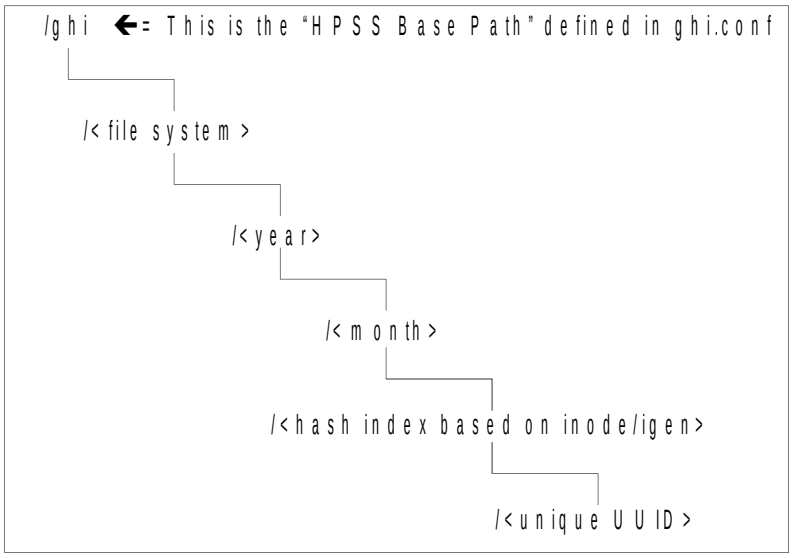

Figure 1 - Location of HSM Files in HPSS

Figure 1 - Location of HSM Files in HPSS shows the location of HSM files in HPSS. Hashing directories are created to store the GPFS files in HPSS. The directories are generated based on the following information:

- ***/ghi*** – This is the default ”root” directory in HPSS for storing the GPFS files. This value can be reconfigured by modifying the GHI configuration (command = ghi_ch_fs). However, this value must not be changed unless the existing directory is empty.
- ***file system*** – GPFS file system name.

- ***timestamping criteria*** - "*year/month*".
- ***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.

1.2.8.2.2 Backup files

Non-Image Backups

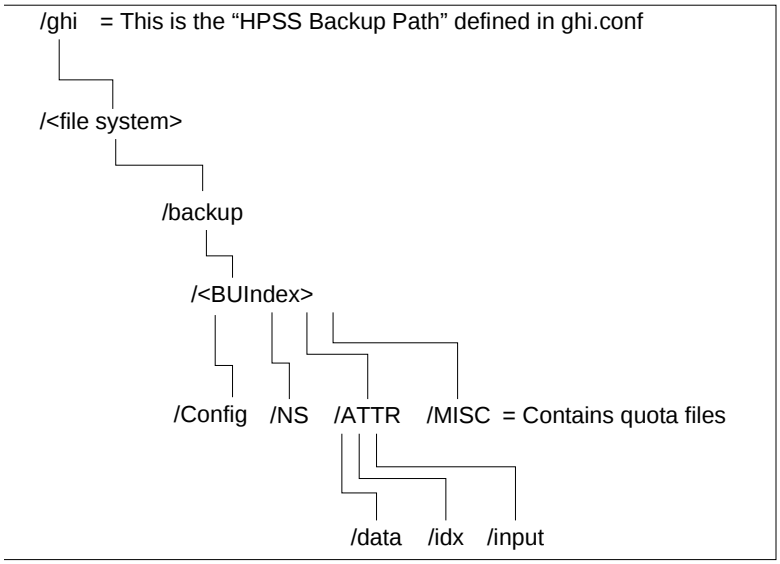

Figure 2 - Location of Backup Files in HPSS for non-image backups

Figure 2 - Location of Backup Files in HPSS for non-image backups shows the location of the backup files in HPSS. Backup files are stored in HPSS based on the snapshot ID at the time of the backup.

- ***/ghi*** – This is the default "root" directory in HPSS for storing the GPFS backup files. This value can be reconfigured by running ghi_ch_fs. However, this value must not be changed unless the existing directory is not empty.
- ***File system*** – GPFS file system name.
- ***BU Index*** – DB2 index associated with the backup.
- ***Config, NS, ATTR and MISC directories*** – Based on type of files being backed up.
    - ***Config*** – contains the GPFS cluster and file system configuration information.
    - ***NS*** – contains the GPFS namespace file attribute information.
    - ***ATTR*** – contains aggregate file information
    - ***MISC*** – contains the GPFS quota files and the version file.

Image Backups

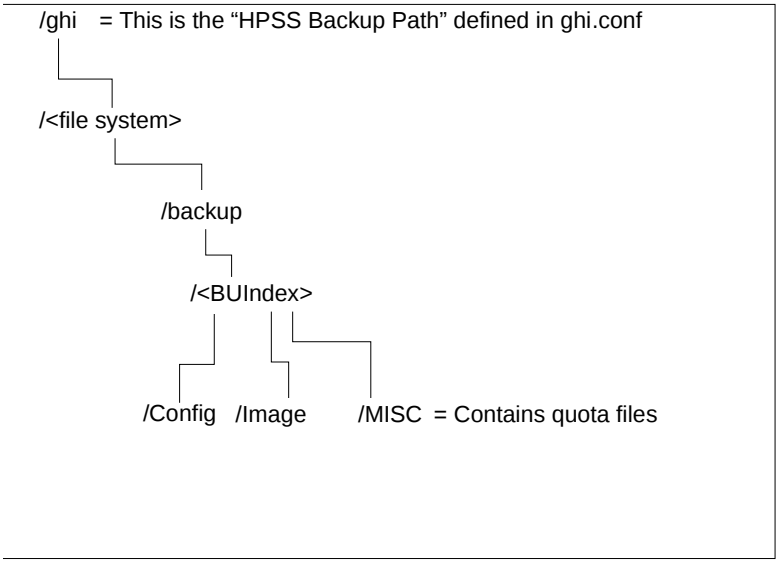

Figure 3 - Location of Backup Files in HPSS for image backups

Figure 3 - Location of Backup Files in HPSS for image backups shows the location of the image backup files in HPSS. Image backup files are stored in HPSS based on the snapshot ID at the time of the backup.

- ***/ghi*** – This is the default ”root” directory in HPSS for storing the GPFS backup files. This value can be reconfigured by running ghi_ch_fs. However, this value must not be changed unless the existing directory is not empty.
- ***File system*** – GPFS file system name.
- ***BU Index*** – DB2 index associated with the backup.
- ***Config, Image and MISC directories*** – Based on type of files being backed up.
    - ***Config*** – contains the GPFS cluster and file system configuration information.
    - ***Image*** – contains the GPFS image backup files
    - ***MISC –*** contains the GPFS quota files and the version file

1.2.8.3. HPSS Class of Service (COS)

Each file in HPSS has an attribute called Class Of Service (COS). The COS defines a set of parameters associated with operations and performance characteristics of a file. The COS results in the file being stored in a storage hierarchy suitable for its anticipated actual size and usage characteristics.
The following rules are defined for COS selection:
Data files (aggregate and non-aggregate) – Selected based on Maximum File Size Hints.
Aggregate index files – Selected based on the GHI configuration.
Backup Files – Selected based on the GHI configuration.
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 file candidates to

specific Classes of Service.

1.2.8.4. HPSS File Families

HPSS files can be grouped into families. HPSS supports grouping files per file family on tape volumes only. All files in a given family are stored on a set of tapes assigned to the family. When one of these files is migrated from disk to tape, it is stored on a tape with other files in the same family. If no tape volume associated with the family is available, a blank tape is reassigned from the default family. The family affiliation is preserved when tapes are repacked. File Families can be specified for each rule in a GHI policy run.

1.2.8.5. HPSS Storage Subsystems and Scalability

Storage Subsystems can be used to separate HPSS resources. GPFS HSM files can be placed on their own resources based on the HPSS Storage Subsystem. GHI currently supports one subsystem per GPFS file system.

1.3. GHI Components

The GHI components (see Figure 4 - GHI Components) consist of GHI servers that provide management for the DMAPI enabled, GPFS file systems. The servers process DMAPI events for mounting and unmounting file systems, as well as process I/O (READ, WRITE and TRUNCATE) events.

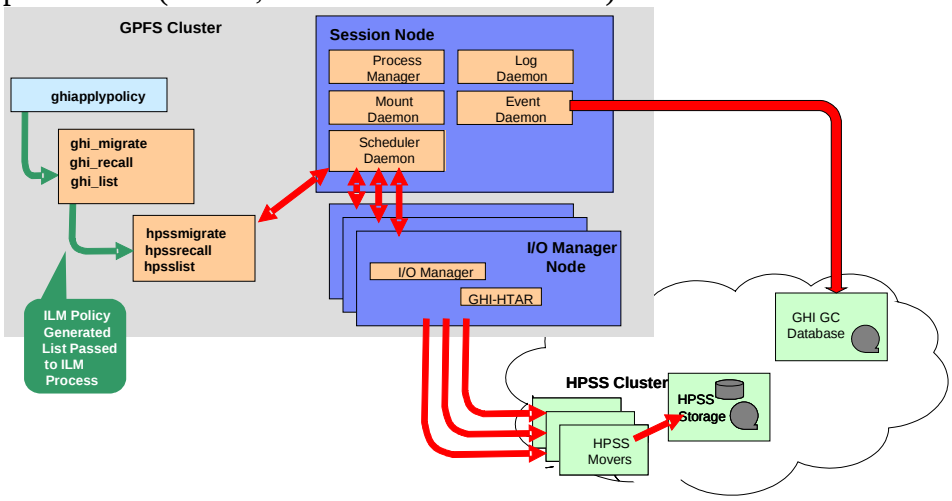

Figure 4 - GHI Components

1.3.1. GHI Servers

GHI consists of the following processes:

- ***Process Manager (PM):*** The Process Manager runs on the GHI Session Node. It is responsible for the following activities:
  1. Starts and stops the other GHI processes.
     a. On startup, the Process Manager starts the Mount Daemon, Log Daemon, and GHI Configuration Manager as child processes.

b. The Process Manager, at the request of the Mount Daemon, starts/stops an Event Daemon and a Scheduler when the file system is mounted/unmounted on the Session node.

2. Ensures the Mount Daemon, Log Daemon, Configuration Manager, Event Daemon(s), and Scheduler Daemon(s) stay functional and that they are able to perform work and minimizes system hangs from occurring.
3. In the event of the death of one of the child processes, the Process Manager receives a SIGCHLD signal. After receiving the signal, the Process Manager restarts that process.

The Process Manager is started by GPFS as part of the GPFS heartbeat mechanism. It is started as part of the GPFS cluster configuration manager node start up process.

- ***Event Daemon (ED):*** The Event Daemon runs on the Session node. It is responsible for the following activities:
    1. Registers for DMAPI I/O (DESTROY, REMOVE, RENAME, READ, WRITE and TRUNCATE) events.
    2. 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 FS, processing of WRITE and TRUNCATE events will be to instruct GPFS to abort the write or truncate operation (instead of passing the request to the GHI Scheduler Daemon).
    3. Receives DESTORY events for files from the DMAPI session queue and performs garbage collection logic on the file.
    4. Receives responses from the GHI Scheduler Daemon and responds to the user request with the result.
    5. On a read-only file system, receives RENAME and REMOVE events for files from the DMAPI session queue and determines whether or not the files are being managed by GHI (i.e., they originated from a GHI backup of the associated full-access FS). If the files are managed by GHI, GPFS is instructed to abort the rename or remove operation. As a result, GPFS will not generate the usual subsequent DESTROY event for that file.
- ***Log Daemon (LD):*** The Log Daemon runs on the Session Node. It is responsible for the following activities:
    1. Maintains two rotating central log files (similar to the HPSS log files).
    2. Logs Event, Minor, Major and Critical log messages from all GHI components.
    3. Archives full log files to HPSS.
- ***Mount Daemon (MD):*** The Mount Daemon runs on the Session Node. It is responsible for the following activities:
    1. Captures MOUNT and UNMOUNT events for DMAPI enabled file systems and instruct the Process Manager to start/stop the

associated Event Daemon and Scheduler.
2. Processes remote mounts for DMAPI enabled file systems.

- ***Scheduler Daemon (SD):*** The Scheduler Daemon runs on the Session Node. It is responsible for the following activities:
  1. Accept data transfer requests from the ED and ILM clients. On a read-only file system, immediately reject requests to transfer data to HPSS.
  2. Communicates with the I/O Managers to transfer data.
  3. Provides a mechanism to pass back transfer results to the ED and ILM clients.
  4. Provides file system's full-access/read-only status, and load balancing to the IOMs.
  5. Processes purge requests for threshold processing.
  6. Filters out duplicate file requests.
- ***I/O Manager (IOM)*****:** The IOM runs on one more more nodes in the GPFS file system. The IOM is responsible for the following activities:
  1. Spawns GHI-HTAR to perform aggregate data transfers.
  2. Coordinate with HPSS to perform non-aggregate data transfers.
  3. Receive GPFS metadata and namespace information from GPFS and transfers it to HPSS when the backup type is the old-style GHI backup (i.e., not a GPFS/image backup). An "old style" backup may still be taken, but it requires action by GHI support personal to accomplish.
  4. Retrieve GPFS metadata and namespace information from HPSS and forward to GPFS to effect file system restoration when the backup type is the old-style GHI backup (i.e., not a GPFS/image backup).
- ***GHI-HTAR/HSIGWD*****:** Provides an HPSS interface for aggregating and retrieving small files. GHI-HTAR clients resides on each node that the I/O Manager resides on. The HSIGWD server resides on HPSS nodes that are assigned by the HPSS administrator.

1.3.2. GHI Infrastructure

The GHI infrastructure items are those components and services used by the various GHI servers. The RPC communication between each of the GHI processes are shown in Figure 5 - Intra-process Communication. The GHI infrastructure components common among servers are discussed below.

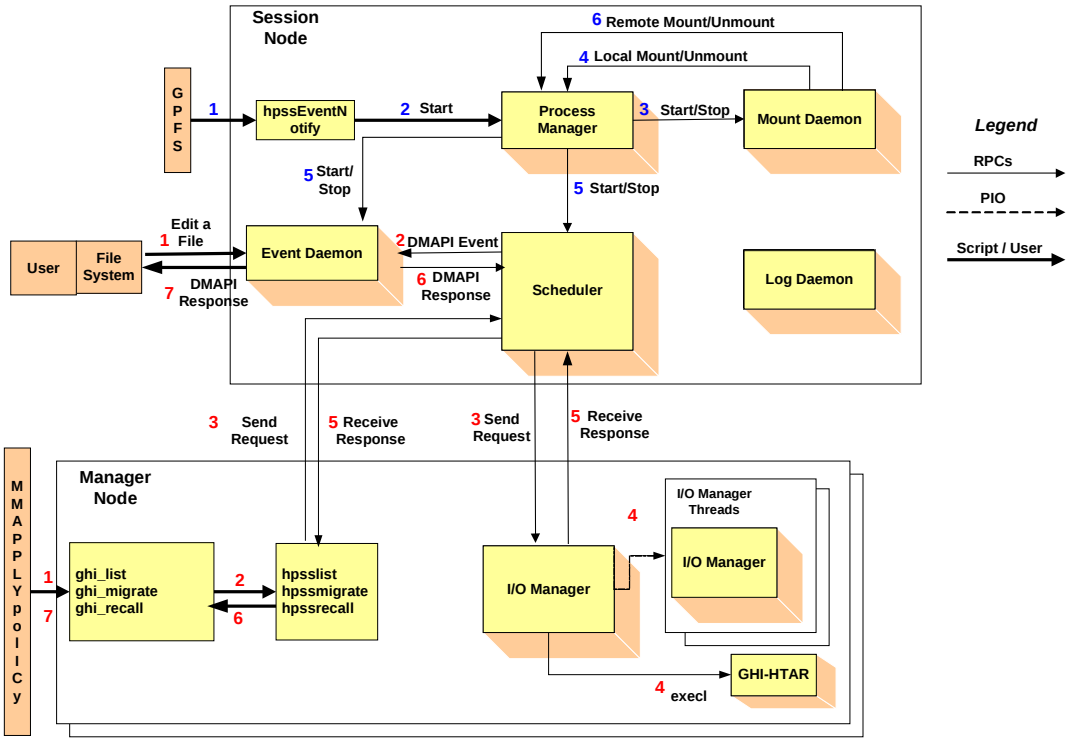

Figure 5 - Intra-process Communication

1.3.2.1. Remote Procedure Calls (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.

1.3.2.2. Thread Services

GHI uses a threads package for multitasking. The threads package enables GHI to serve large numbers of concurrent users and to enable multiprocessing of its servers.

1.3.2.3. Security

GHI uses HPSS security software to allow GHI components to communicate in an authenticated manner, to authorize access to HPSS objects, to enforce access control on HPSS objects and to issue log records for security-related events. The security components of HPSS provide authentication, authorization, enforcement, and audit capabilities for the HPSS components.

Authentication: responsible for guaranteeing that the GHI principal user, ***hpssdmg***, 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.

1.3.2.4. Logging

A logging infrastructure component in GHI provides an audit trail of server events. Logged data includes alarms, events, requests, migration and backup information. The GHI Log Daemon, maintains a central log. When the central log fills, messages are sent to a secondary log file. When the log file rolls over, the full log file is sent to HPSS to be archived.
GHI-HTAR and the HSIGWD provide separate logs that contain audit trail events and file transfer information. Currently, these logs must be manually maintained. SYSLOG logging is also available as a configuration option, and the standard SYSLOGD mechanisms may be used to control the number and size of the audit trail logs.

1.3.3. GHI User Interfaces

GHI provides the user with a transfer interface, ***ghiapplypolicy***. The interface is a wrapper used to control the location where output files from the policy run are placed. It also controls the number of entries, i.e. bulk rates, of each of the output files.

1.3.4. ILM Policies

There are a number of aspects of storage management that will differ at each GHI site. For instance, sites typically have their own guidelines or policies covering how they want to implement data migration/recalls. In order to accommodate site-specific policies, GHI has implemented a set of policy templates to be used as guidelines to allow a site administrator the freedom to tailor management operations to meet their particular needs. Refer to the *GPFS Advanced Administration Guide* for implementation of the ILM policies.

1.3.4.1. ghi_backup

This utility allows a site administrator to backup GPFS metadata into HPSS. The ILM policy management interface is used to generate lists of files that need to be migrated, the file system namespace information, and a list of the GPFS files to be used to gather the file attributes. Figure 6 GHI Backup Functionality shows the steps that are taken to complete a backup.

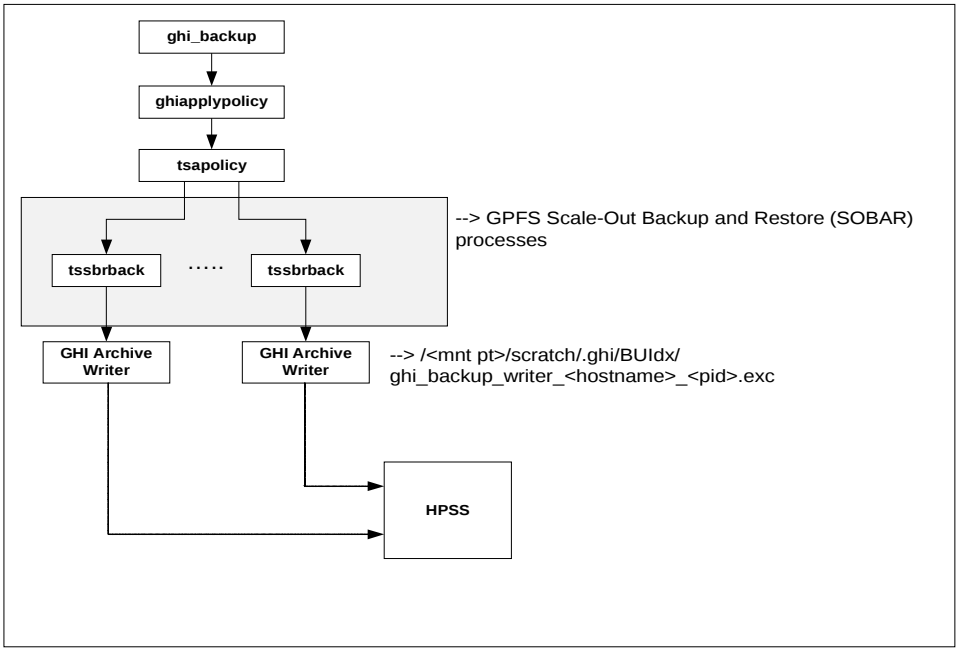

Figure 6 GHI Backup Functionality

Only full backups are supported in this release of GHI, although incremental backups from prior releases may still be restored. Full backups gather the attributes for all the GPFS files in the file system. Incremental backups only gather the attributes for the files that have changed since the previous snapshot.
GPFS snapshots are used to capture a point in time snapshot for the file system. The file system is quiesced before capturing the namespace and attributes for the files being backed up.

1.4. GHI Hardware Platforms

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 components. The Secondary Session Nodes can act as I/O Manager Nodes until they are told to take over as the Primary Session Node. The following diagram in Figure 7 - GHI Hardware Platforms provides an example of a typical GHI system configuration.

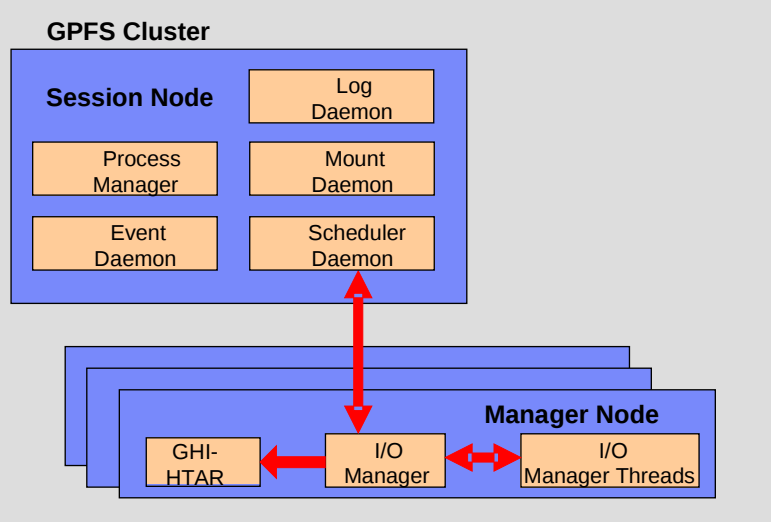

Figure 7 - GHI Hardware Platforms

1.4.1. Session Node Platforms

The Session Node is a machine where a DMAPI session has been instantiated and it has registered to receive DMAPI events. This node also functions as the GPFS Cluster Manager Node. The following GHI servers run on the Session Node:

- GHI Configuration Manager (one instance per cluster).
- Event Daemon (one instance per file system).
- Log Daemon (one instance per cluster).
- Mount Daemon (one instance per cluster).
- Process Manager (one instance per cluster).
- Scheduler Daemon (one per file system).

1.4.2. I/O Manager Platforms

I/O Managers are used to control the logical network attachment of storage devices and are configured to run on one or more nodes. IOMs are supported on both AIX and Linux platforms.
This node performs data transfers between GPFS and HPSS. The following GHI processes run on the IOM node:

- I/O Manager – performs non-aggregate transfers to/from HPSS and coordinates with GHI/HTAR.
- GHI-HTAR -- performs aggregate transfers to/from HPSS.

1.5. Process/Node Failover/Recovery

GHI provides a feature to recover from any failed GHI process (either restart the process on the same node, restart the process on a new node, or have another ‘like’

process take over the work load). The current implementation takes advantage of the GPFS "user exit" mechanism that provides the capability to determine failure of the GHI Session node, and fail-over of the GHI processes to a Secondary Session node.

GHI handles several types of failover scenarios:

- Session node failure or loss of quorum.
- GHI Session node process failure.
- IOM failure.

1.6. Disaster Recovery Plan

GHI metadata disaster recovery requires full consideration by the administrator and the HPSS service team during the "Planning Process". The degree to which the customer wishes to protect GHI metadata and user data, and provision for the protection and recovery of GHI metadata and user data will be documented by the customer and reviewed by IBM. Refer to the *HPSS Disaster Recovery Guide* for more information.

2. GHI PLANNING

2.1. Overview

This chapter provides GHI planning guidelines and considerations to help the administrator effectively plan and make key decisions for utilizing an HPSS GHI system.
Careful planning is required to fully consider how the resulting system will operate in an efficient manner and best meet site requirements.
The following sections describe the preparation steps for the GHI installation, configuration, and operational phases.

2.1.1. GHI System Architecture

Figure 8 - GHI System Architecture, shows the basic architecture of an HPSS GHI system and their relationship to HPSS server nodes, and HPSS Mover nodes.

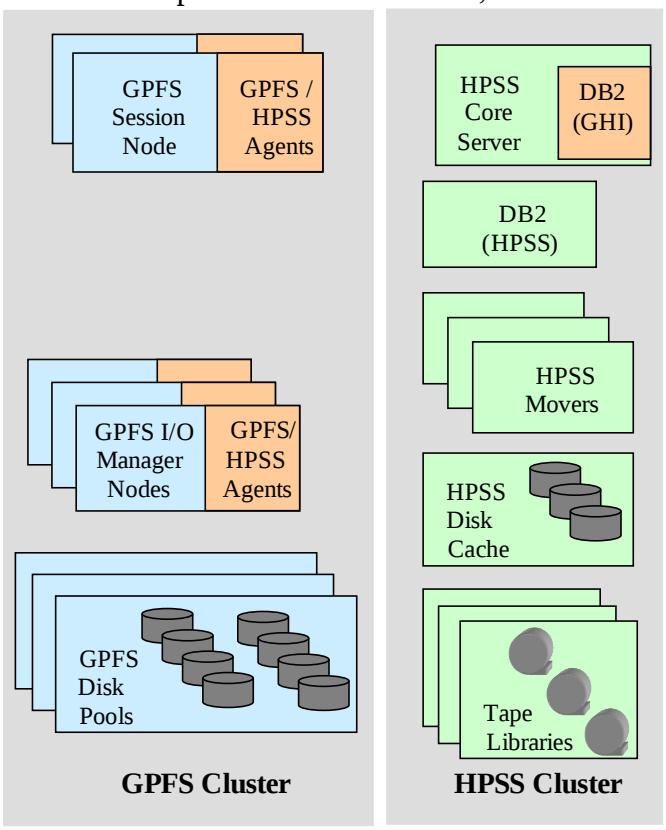

Figure 8 - GHI System Architecture

Specifics of this architecture for a given site are developed during the proposal and initial project planning stages of a deployment. Often the disk and tape data resources for GPFS and HPSS are dictated by equipment already available and budgetary constraints on what can be purchased. Specific quantities and sizing of

these data resources are beyond the scope of this planning document. For this document, it is assumed these parameters were already defined. Contact your HPSS service provider with questions about and for assistance with GHI resource sizing and configuration. See section *2.4 GHI Sizing Considerations* for sizing considerations for the metadata resources as well as the local disk resource requirements.

2.1.2. GHI Configuration Planning

When planning to provide HSM space management and backup and restore services for GPFS using HPSS, it is very important to review the following from the initial proposal and planning of the system:

- The user's or HPC bandwidth and capacity requirement on GPFS.
- The additional capacity requirements being imposed on the GPFS file system to hold the extended attributes used by GHI.
- The additional bandwidth needed to support the HSM activity between GPFS and HPSS.
- The bandwidth and capacity requirements of HPSS for HSM file data and any backup and restore requirements of the solution.

When looking at the GPFS requirements, the following items should be considered important and should be reviewed by the customer:

- Verify the site's storage requirements and policies, such as the initial storage system size, anticipated growth, usage trends, average file size, expected throughput, backup policy, and availability.
- Review the architecture of the entire Storage Subsystem to ensure that it satisfies the above requirements. Confirm with the GPFS architects that the GPFS solution is properly sized to account for HSM activity. The review should include the following for GPFS:
    - Verify the total number of files (both scratch files and HSM space managed files) being stored in GPFS.
    - Verify the types of GPFS files. Obtaining a file size distribution histogram is often helpful.
    - Verify the data pathways (the network) of the GPFS cluster. GPFS clusters are often used with HPC activity. It is important to understand the HPC bandwidth requirements for scratch files, and to understand how the raw data and eventually the product data will be moved and stored.
    - Verify the GPFS file system block size.
    - Verify how the GPFS storage is configured into the cluster. Capture the amount of data and metadata resources.
        - Identify the amount of metadata disk resources needed for the DMAPI attributes for all HSM space managed files. When a GPFS file is copied to HPSS, GHI creates an extended attributes outside of the i-node. GHI stores the HPSS reference material in the GPFS extended attribute.
        - GPFS files that are one block or smaller are not purged from

the file system. Identify the amount of space needed to hold all of those small files.
        - Verify the amount of inode space that is needed.
    - Verify how often to backup the GPFS file system and identify any backup and restore requirements.
- Review the architecture of the entire HPSS system to satisfy the HSM portion and the Backup/Restore portion of the requirements. The review should include the following for HPSS:
    - Verify the amount of disk cache needed for the aggregate index files if aggregation is to be used.
    - Verify the location of the HPSS Mover nodes vs. IOM nodes. IOM nodes can be either GPFS NSD nodes or GPFS client nodes. HPSS Mover nodes may be inside the GPFS cluster or outside of the GPFS cluster.
    - Verify COS selection.
        - ***Non-Aggregate Files***: Consider a disk-to-tape COS with purge, or a tape only COS.
        - ***Aggregate Data Files***: Consider a disk-to-tape COS with purge, or a tape only COS.
        - ***Aggregate Index Files***: Consider a disk-to-tape COS, no purge.
        - ***Backup Files***: Consider a disk-to-tape COS with purge.
    - Verify File Family selection. Consider using file families to guarantee associated files are placed on the same set of tapes.
    - Verify the HPSS subsystems to be configured and how resources will be allocated among them.
    - Verify the amount of network capacity that is required to provide data to the tape drives at rates sufficient to keep them running at their rated speed.
- Confirm the list of prerequisite software that needs to be obtained in order to satisfy the target GHI architecture. Refer to *Section 2.2.5 - Prerequisite Software Overview* for more information on the HPSS prerequisite software requirements.
- Confirm the space requirements needed for temporary policy output files.
- Confirm the resources required to handle the work loads to be imposed on the GHI nodes. Refer to *Section 2.3 - Considerations* for more discussions on the system resource requirements.

2.1.3. Software Planning

Refer to Section 2.2.5 - Prerequisite Software Overview for more information on the required software that is needed to run GHI.

2.1.4. Operations Planning

The following section outlines key aspects of the operations planning process which

is reviewed by the HPSS system engineering and deployment team with the customer. This information is first obtained during the proposal or opportunity assessment phase. It is then reviewed during the implementation and pre-production phases since requirements and/or assumptions will often need to be updated from the initial understanding of the system. The customer is responsible for providing operations planning information to the HPSS systems engineering and deployment team and for communicating changes and updates. The HPSS systems engineering and deployment team will use the information to review the configuration with the customer to provide an assessment of the system's readiness for operational use. The following planning steps must be carefully considered and reviewed by the customer for the GHI operational phase:

- Verify the user's operations scenarios, including, but not limited to:
  - ***GPFS file systems***: Will the customer have multiple GPFS file systems? Are the file systems to be segregated? Are individual directories to be segregated into their own file families?
  - ***Data ingest rates***: What amount of data are the users going to write into the GPFS file system?
    - Which of the users' files will be archived?
    - Will there be a scratch area which does not need to be archived?
    - Will GPFS files be pinned, are there GPFS files which are required to stay in the GPFS file system, but still need to be archived/backup?
  - ***Data archive rates***: How much of the user data will make its way into the archive (HPSS) via GHI?
  - ***Data retrieval rates***: How much of the user data will be read from the archive via GHI?
  - ***Aggregation or not***: Will GHI aggregation be used? What are the GPFS file sizes to be included in the GHI aggregate? How many files per aggregate?
  - ***Filesystem backups***: How often will the GPFS file system be backed up? How many backups will the customer require to be online? How many backups will the customer require to be offline? How long does the customer keep backups?
- Confirm which NSD quorum nodes will be used for potential Session nodes.
- Confirm if the GHI IOMs will be co-resident with the GPFS NSD nodes or be resident on GPFS client (Non-NSD) nodes.
- Determine the number of IOMs required to achieve the data transfer rates.

2.1.5. GHI Deployment Planning

The following planning steps must be carefully considered for the GHI deployment phase:

- Once the items in Section *2.1.2 - GHI Configuration Planning* have been analyzed, it is important to confirm that the GPFS and HPSS architecture

and resources allocated to GHI are adequate to meet expected data transfer rates and storage capacity. The HPSS team members are not GPFS solution architects, but will participate in the analysis of the overall GPFS + HPSS coupled solution, along with the customer and the GPFS architects.

- Before the GHI deployment begins, the prerequisite foundations must be installed and operationally verified by the customer and the HPSS team
- The HPSS deployment team will be available to work with the customer and GPFS deployment team to determine a customer test plan that demonstrates that the GPFS file system can manage the aggregate load that the customer is expecting. It is important to understand and plan for the transfer and transaction rates of the system.
- The GPFS and HPSS deployment team will work with the customer to help them determine if the GPFS + HPSS coupled solution is ready for production use.

2.2. Requirements and Intended Uses for GHI

This section provides some guidance for the administrator to identify the site's requirements and expectations of GHI. Issues such as the amount of storage needed, access speed and data transfer speed, typical usage, security, expected growth, data backup, and conversion from an old system must be factored into the planning of a new GHI system.

2.2.1. Storage System Capacity

The amount of GHI user data storage space the administrator must plan for and take into account includes the following considerations:

- The amount of metadata storage space required to support storage management activities such as policy runs, aggregation, backups, DMAPI. The extended attributes are stored in blocks of data outside the inode metadata. Each extended attribute uses additional GPFS disk resources of at least 16K, or a GPFS sub-block, which is 1/32 of a GPFS data block.
- The amount of user data storage space required to support a specified number of files to remain in disk cache.
- The amount of disk cache required for HPSS should include the space required for Aggregate Index files as well as for backup files.

2.2.2. Required Throughput

Determine the required or expected throughput for the various types of data transfers that users will perform. Some users want quick access to small amounts of data. Other users have huge amounts of data they want to transfer quickly, but are willing to wait for tape mounts, etc. In all cases, plan for peak loads that can occur during certain time periods. These findings must be used to determine the type of storage devices and network to be used with HPSS to provide the needed throughput.

Site planners should consider file system functionality which can cause scalability issues.

Policy scan overhead and time to complete increases as the GPFS file system grows. Sites should consider including multiple rule definitions in a single policy run rather than fewer rules, and more policy runs. Also, if sites are generating both aggregate and non-aggregate rules, the rules for non-aggregates should be placed before the aggregate rules. Aggregate lists must be constructed, whereas non-aggregate requests are processed immediately.

POSIX command line interfaces such as "*file* *", can cause staging of GPFS file data from GPFS. Consider keeping the first block of each file's data on GPFS disk. This is a candidate for a future GHI release to configure the file system such that the first block of the file contents can be kept on the GPFS file system. Currently, releasing the GPFS resources frees up all data blocks from the GPFS file contents.

The time to create and destroy snapshots is relative to the size of the GPFS file system, as well as file modifications to the GPFS file data during the snapshot operations.

2.2.3. Load Characterization

Understand the kind of load users are putting on an existing file storage system provides input that can be used to configure and schedule the GPFS system. What is the distribution of file sizes? How many files and what amount of file contents are transferred in each category? How does the load vary with time (e.g., over a day, week, month)? Are any of the data transfer paths saturated?
Having this file system load information helps to properly size both GPFS and HPSS so that they can meet the peak demands. Also based on this information, maintenance activities such as purge and backups can be scheduled during times when the system is less busy.

2.2.4. Security

Authentication and authorization between GHI servers is done through use of either UNIX or Kerberos security tools for authentication and UNIX for authorization services.
GHI should be configured to use the same authentication/authorization that HPSS is configured with. It is basically a client to HPSS. GHI uses the ***hpssdmg*** principal for all authentication.
If aggregation is used, the same ***hpssdmg*** principal applies.

2.2.5. Prerequisite Software Overview

This section defines the prerequisite requirements for GHI. Some products must be obtained separately from GHI and installed prior to the GHI installation and

configuration.

2.2.5.1. DB2

GHI uses the DB2 Universal Database Enterprise Server Edition by IBM Corporation to manage all GHI metadata for backup/restores. DB2 software and a limited-use license is included in the HPSS distribution. Refer to ***db2_install*** to install the DB2 client.

2.2.5.2. GPFS

Please refer to the *GPFS Advanced Administration Guide*, when installing and configuring GPFS. There are many online resources that are available. IBM Redbooks may also be considered as a valuable resource.

2.2.5.3. HPSS Client API

Please refer to the *Section 5.4.2.1 - Construct the HPSS Mover/Client/Other HPSS Server*
Source Tree in the *HPSS Installation Guide* to install/build the Client API on the GHI Session nodes.

2.2.5.4. DCE,Kerberos

GHI uses Massachusetts Institute of Technology (MIT) Kerberos to implement Kerberos authentication. MIT Kerberos is a network authentication protocol designed to provide authentication for client/server applications by using secret-key cryptography. A free implementation of this protocol can be downloaded from the MIT's website (http://web.mit.edu/kerberos/). Refer to *Section 5.2.2 - Install MIT Kerberos* in the *HPSS Installation Guide* for more information.
For Linux, Kerberos is installed as part of the operating system.
If UNIX authentication will be used, this product is not required.

2.2.5.5. GHI-HTAR

GHI uses GHI-HTAR to implement file aggregation when migrating files into HPSS. GHI-HTAR bundles small files on a platform into large files in storage. Temporary storage is used to build the index files associated with an aggregate. The temporary files are removed once the file is written to HPSS. IBM will supply a version of GHI-HTAR that is compatible with the version of GHI that is being installed.

GHI aggregated data relies upon GHI-HTAR index files to always be available to the system. If an index is inaccessible (missing, damaged, or delayed for an extended period of time), 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 and/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.

If aggregation will not be used, this product is not required.

2.2.6. Prerequisite Summary Based on Node Type

The Session nodes and IOM nodes all require the following prerequisite software:

- GPFS 3.5 PTF 16
- HPSS 7.4.2 (Mover/Client Interface)

This section provides a summary list of prerequisite software required for HPSS. It also lists the software versions which have been verified with HPSS.

2.2.6.1. HPSS Core Server

The HPSS Core Server node requires the following to support aggregation:
HSI 5.0.0
Openssl version 0.9.8g or later.

2.2.6.2. GHI Session Nodes

The active GHI Session node contain the following processes: Process Manager, Log Daemon, Mount Daemon, GHI Configuration Manger, and multiple Scheduler and Event Daemons. An IOM can also be configured on this node, and can either be active, or remain dormant when the node is the active Session node.

2.2.6.2.1 AIX Requirements

Each AIX Session node must have the following installed:

- IBM RS/6000 (eServer pSeries) with a minimum of 8 cores and 16 GB RAM.
- AIX 6.1 (6100-08_AIX_ML which consists of Technology Level 8 and Service Pack 3).
- DB2 UDB V10.5 Enterprise Server Edition (ESE) for AIX.
- MIT Kerberos 1.6.3 (if planning to use Kerberos authentication).
- C compiler for AIX, version 10.1.0.0 (if planning to recompile the HPSS Client API and GHI code on this node).

2.2.6.2.2 Linux Requirements

Each Linux Session node must have the following installed:

- Linux machine (eServer xSeries) with a minimum of 8 cores and 16 GB RAM.
- Red Hat Enterprise Linux EL release 6.4 (Santiago).
- DB2 UDB V9.10.5 Enterprise Server Edition (ESE) for LINUX.
- MIT Kerberos 1.6.3 (if planning to use Kerberos authentication).
- C compiler for Linux: gcc-4.1.2 (if planning to recompile the HPSS Client API and GHI code on this node).
- Openssl version 0.9.8g or later.

2.2.6.3. GHI I/O Manager Nodes

An I/O Manager node will execute an I/O Manager administrative process for each managed file system for which it is configured, and GHI-HTAR processes will be spawned by an IOM to perform data transfers.

Each IOM node must have the same prerequisites as a Session node of the same hardware/OS combination (i.e, AIX or Linux) except:

- The C compiler and DB2 are not required.
- GHI-HTAR 5.0.0 is required

2.3. Considerations

This section describes the infrastructure needed to operate GHI and includes considerations about infrastructure installation and operation that may impact GHI.

2.3.1. Network Considerations

Because of its distributed nature and high-performance requirements, a GHI system is highly dependent on the networks to provide connectivity among the GHI servers, HPSS, and GPFS.
For control communications (i.e., all communications except the actual transfer of data) among the GHI servers, GHI requires TCP/IP services. Since control requests and replies are relatively small in size, a low-latency network usually is well suited to handle the control path.
The data path is logically separate from the control path and 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 supporting large data transfers, the latency of the network may not impact overall data throughput.

2.3.1.1. Network Considerations

GHI supports using IPV6 addresses through the HPSS environment. The HPSS_NET_FAMILY can be set to ipv4_only, ipv6, or ipv6_only. This value must be set in both the /var/hpss/etc/env.conf and /opt/hpss/bin/htar.ksh. Failure to correctly set this value will result in connection and data transfer errors.

2.3.2. General ILM Policy Considerations

The GPFS ILM migration policy provides the capability for GHI to copy (migrate) files from GPFS to HPSS. The GPFS ILM 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 simply copies them from GPFS to HPSS. Larger files are usually copied straight to HPSS, while the smaller files are usually aggregated into larger HPSS objects.

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

The GPFS ILM recall policy provides the capability for GHI to stage files in bulk from HPSS back to GPFS. The GPFS administrator will need to author an ILM policy rule to stage a given set of files back from HPSS. These requests will be optimized so that files located on the same tape will be recalled together to minimize tape mounts.

The site administrator will need to monitor the GPFS ILM policies to ensure that they are completed without errors. Errors in the migration process may lead to undesired file system behavior – file system may fill up, backups may be incomplete, etc. The GHI version 2 software release plan includes a summation of the number of files that were transferred successfully, as well as the number of files that were unsuccessful.

The GPFS ILM backup policy provides the capability for GHI to make a point-in-time backup of the GPFS file system. The lists of files are processed by GHI. Some lists are copied directly to HPSS, while other lists are used to gather file attribute information to generate files containing the metadata. Those files will also be copied to HPSS. The result of the GHI processing is a point-in-time backup of the GPFS file system.

Administrators should experiment to determine the parameter settings that will fit the needs of their site. If a site has a large amount of disk file write activity, the administrator may want to have more free space and more frequent purge runs. However, if a site has a large amount of file read activity, the administrator may want to have smaller disk free space and less frequent purge runs, and allow files to stay on disk for a longer time.

The policy generates multiple ****.exc*** and a ****.ok*** files. The exception files (*.exc) contains all the files that failed during the policy run. The okay (*.ok) files contain all the files that were successfully transferred. The exception files are displayed as a

result of the policy run. 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: This is the "dirty flag" option keeps not only the exception and okay files, it also keeps 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.

2.3.2.1. HSM Policy Considerations

Figure 9 - HSM Policy Output shows an example of the output files for a policy run. The reference to "EXT" reflects the operation being performed: "migrate", "recall" or "purge".

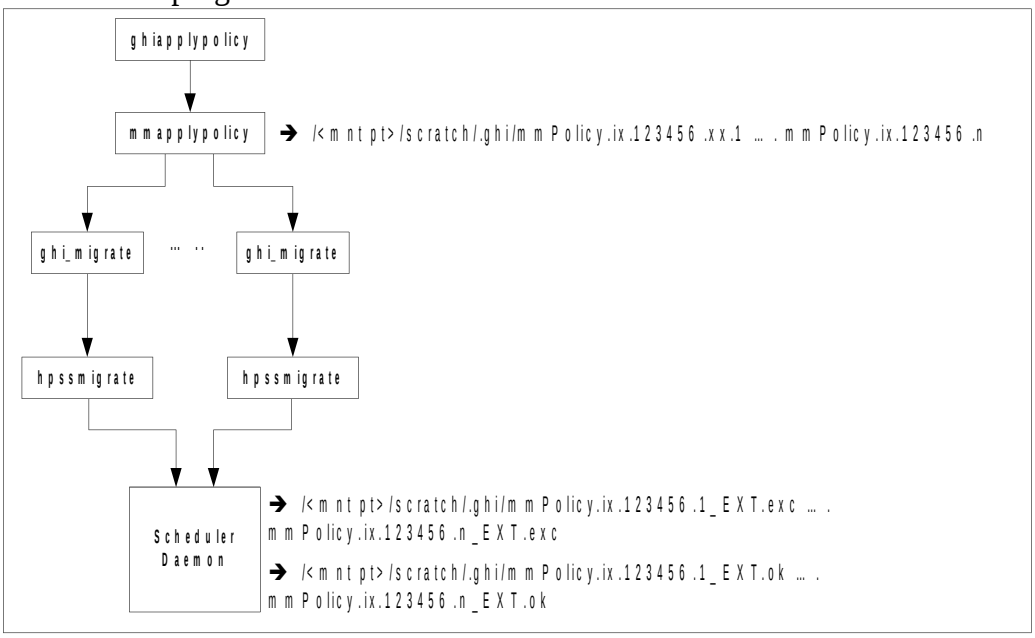

Figure 9 - HSM Policy Output

2.3.2.1.1 migrate.policy

The directory ***/<file system>/scratch/.ghi*** is where GHI places the temporary files for migrations, so that directory tree should be omitted from being migrated to HPSS. For optimization, two additional rules should be defined to exclude files that are already co-managed or are of length zero. All three of the preceding rules are included in the template migration policy supplied with GHI. The migration uses a "Max Aggregate Files" value managed via GHI configuration commands **ghilsfs** and **ghichfs**. This value is used to generate the policy output files in bulk sizes based on this amount. So, for example, if the bulk size is 100, and 500 files are output candidates from the policy run, then 5 output files will be generated from the policy engine. If all 500 files are selected for aggregation, then 5 100-file aggregated (HTAR) 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 less than 5 HTARs may be created in HPSS.
Any attempt to migrate files from a GHI read-only file system will be rejected. The rejection code will be -1 (Operation not permitted).

2.3.2.1.2 recall.policy

The directory ***/<file system>/scratch/.ghi*** is where GHI places the temporary files, so that directory tree should be omitted from being recalled from HPSS. For optimization, an additional rule should be defined to exclude files that are not co-managed. Both of the preceding rules are included in the template migration policy supplied with GHI.
The recall policy does not use a bulk size. The policy generates one list. That list is parsed into aggregates and non-aggregates. Non-aggregates are sorted by the HPSS tape on which they reside and position within the tape. and the results sent to the Scheduler on a per-tape basis. Aggregates are sorted based on which aggregates they are located in. and the results sent to the Scheduler on a per-aggregate basis. This allows GHI to make a single request to GHI-HTAR to retrieve all files requested to be retrieved from that aggregate.

2.3.2.2. Backup Policy Considerations

Figure 10 - Backup Policy Output shows an example of the output files for a policy run. The figure only shows the output from the ‘mmimgbackup’ portion of the policy run. The migration output will be similar to the output shown in Figure 9 - HSM Policy Output. There will be sets of output files generated from the IOMs: “EXT” will be both “backup_ns” and “backup_attr” for both namespace and attribute file generation and backup into HPSS. The “backup” files generated by the SD will contain general backup failures for backup of input files and GPFS quota files into HPSS.

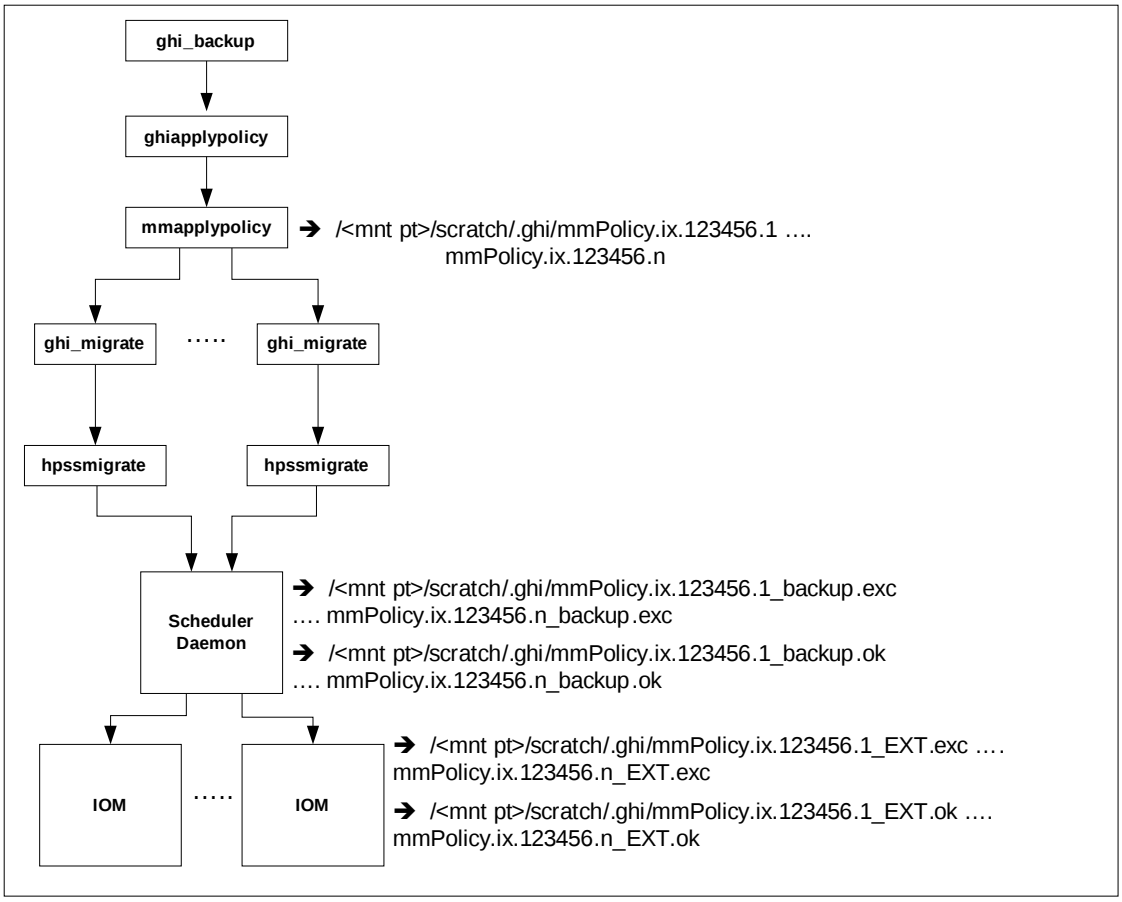

Figure 10 - Backup Policy Output

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, and works as described in *Section 2.3.2.1.1 - Migration Policy*. Backing up the namespace and attribute files is performed via the GPFS mmimgbackup command. Any attempt to take a [GHI] backup of a GHI read-only file system will be rejected.

2.3.2.3. Scratch Area

GHI uses a temporary directory in the GPFS file system to store temporary files. The following file types are stored here:
Policy output files: The output files generated from the GPFS policy.
***GHI-HTAR index files*:** GHI-HTAR stores the temporary index files here before writing the file to HPSS.
Session node configuration file*:** This configuration file, ***sn_host.conf, is used by each of the nodes to determine which node is the active session node. This file is used to locate the Scheduler Daemon.
Backup configuration file*:** The backup configuration, ***backup.conf, is generated by the ghi_backup script and is used by the I/O Managers to determine the location in HPSS for storing the files generated by the backup process.

2.3.2.4. Thresholds

A threshold policy will be defined to manage the available space. It will define the high (NO_SPACE) and low (LOW_SPACE) water marks. It will be associated with one or more file systems. Each file system can have its own threshold policy. When the file system reaches the high-water mark, the policy will be invoked to free up GPFS disk resources. We refer to this as "purging data". See *Section 2.3.2.4.1- Purging Data* for more information.
Candidates can be considered based upon file age, file size, etc. The GPFS ILM policy allows the candidates to be weighted, so you can specify which files to be considered first. The list of files generated will be enough to free up resources until the low water mark is reached.
To configure the system to react to these events, add the threshold policy and update the GPFS configuration to enable threshold processing. When activated, and one of the 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 what files to "punch holes" into.

2.3.2.4.1 Purging Data

Freeing GPFS disk resources is referred to as purging or "punching a hole". Files to be purged must have already been migrated into HPSS and have within HPSS been migrated to tape. The migrated-to-tape requirement can be disabled via a GHI configuration setting, but doing so increases the opportunity for data loss.
Policy rules are used to weigh or exclude files that will not be considered as purge candidates. You can specify a clause like:

```
WHERE FILE_SIZE >= 262144     # where 262144 = block size of the
                              # file system
```

The GHI configuration command-line utilities **ghilsfs** and **ghichfs** allow a site to define how many bytes of data will remain in the GPFS file system after the file is purged.

2.3.3. High Availability Considerations

When a High Availability(HA) configuration is used it is critical to ensure that all components that are configured on the primary Core Server are also configured on the backup HA Server. Please reference the 2.4 GHI Install Guide for what to install on the HA server.

2.4. GHI Sizing Considerations

There are four types of storage space that must be planned for:
GHI Infrastructure Storage Space.
GPFS File Systems.
GHI Metadata Space.
System Memory and Disk Space.

2.4.1. GHI File Systems

The following sections describe the various file systems used by GHI. When GHI is deployed into the GPFS cluster, these directories will be needed. Each GPFS node selected for GHI use will require a trivial amount of file system disk space – approximately 15 MB.

2.4.1.1. /opt/hpss

This directory holds HPSS/GHI binaries, source code, include files, libraries, and utilities. The GHI software is installed in the ***/opt/hpss/src/ghi*** directory.

2.4.1.2. /var/hpss

The ***/var/hpss*** directory tree is the default location of a number 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 the /***var/hpss*** file system the following sub-directories exist:
The ***/var/hpss/cred*** is the default directory where some additional UNIX configuration files are placed. These files are typically very small.
The ***/var/hpss/etc*** is the default directory where some additional UNIX configuration files are placed. These files are typically very small.
The ***/var/hpss/ghi*** is the default directory where several GHI files are maintained. There are five sub-directories required: config, etc, log, policy, and temp. HPSS environment variable $HPSS_GHI_ETC_PATH contains the actual base path to the GHI "etc/" directory. This is the directory portion which appears before the "/etc" (e.g. /var/hpss/ghi) .

The ***/var/hpss/ghi/tmp*** is the default directory where the Process Manager creates a lock file for each of the GHI process it started in the node. GHI may also write diagnostic log files and performance files here as well. The lock files are very small, but the log files may get into the MB range while the performance files can easily grow into GB or even larger. Performance monitoring, which results in creation of performance files, is normally disabled precisely because of the demands it places on disk resources.

The ***/var/hpss/ghi/log*** is the default directory where the Log Daemon creates two central log files. The size of these log files is specified in the Log Daemon specific configuration. By default, these files are 10 MBs each. Each of the GHI servers write its server-specific log files to this directory as well. They write two rotating log files that are based on the amount of logging that is turned on (also each capped at 10MB).
The ***/var/hpss/ndapi*** is the default directory where the log and performance files are found for HSIGWD. This directory resides on the HPSS Core Server node.
The ***/var/hpss/hsi*** is the directory containing the HSI source and wrappers/binaries. It also contains the sources for GHI-HTAR and HSIGWD.

2.4.1.3. /var/hpss/adm/core

/var/hpss/adm/core is the default directory where GHI servers put "core" files if GHI processes terminate abnormally. Core files may be large, so it is required that there be at least 2 GB reserved for this purpose on the Session node and at least 1 GB on IOM nodes.

It is up to the administrator to remove unneeded core files to prevent the /var/hpss/file system from filling up.

2.4.1.4. /var/hpss/hpssdb

The ***/var/hpss/hpssdb*** directory is the location where the database instance is stored, which is used to access the remote DB2 server on the HPSS Core Server. The minimum file system size required is 20-30 MB for the runtime DB2 client.

2.4.1.5. /var/hpss/ndapi

The ***/var/hpss/ndapi*** directory is the default directory containing the audit trail and transfer logs created by GHI-HTAR. These files grow without bound, and must be manually managed by the site. Normally a ***cron*** job is run periodically to copy filled logs into HPSS and then remove or null them out.

2.4.2. GHI Metadata Space

During the GHI planning phase, it is important to properly assess how much disk space will be required by DB2 to store GHI metadata. The first step in this process is to understand the metadata tables managed by DB2. The database table used by GHI is for storing the information in a backup and for storing files for garbage collection.
For the backup table, there is one row generated each time a GHI backup is performed. The row is 48 bytes in length. Rows are added each time a backup is initiated and are never deleted. This table resides on the HPSS Core Server node.
For the garbage collection table, there is one row generated for each deleted file contained in a backup. The row is 76 bytes in length. Rows are added each time a GHI-managed file is deleted from GPFS or altered. Rows are deleted whenever the associated HPSS file is garbage-collected, The table resides on the HPSS Core Server node.

2.4.3. System Memory and Disk Space

The following sections discuss requirements for disk space, system memory, and paging space.

2.4.3.1. System Memory and Paging Space Requirements

The memory and disk space requirements for the nodes where the GHI processes

will execute depends 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 GPFS cluster nodes running GHI processes. When GPFS 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.

2.5. GHI Interface Considerations

This section describes the user interfaces to GHI and the various considerations that may impact the use and operation of GHI.

2.5.1. GHI Server Considerations

Servers are the internal components of GHI that provide the system's functionality. They must be configured correctly to ensure that GHI operates properly. This section outlines key considerations that should be kept in mind when planning the server configuration for a GHI system.

2.5.1.1. Session Node

The Process Manager, Mount Daemon, Configuration Manager, and Log Daemon processes get started automatically when GPFS is started. GHI provides a utility, ***hpssEventNotify***, that GPFS calls when it comes online. That utility, in turn, starts the Process Manager, which in turn starts the other processes.
When GPFS is stopped, or GPFS loses quorum on the Session node, the ***hpssEventNotify*** utility is called to stop those processes, and in the event of a failover, starts then up on another node.
When a file system is mounted, the Mount Daemon receives the MOUNT event and notifies the Process Manager to start an Event Daemon and Scheduler Daemon to be associated with that file system.
When a file system is unmounted or GHI is shutdown, the Mount Daemon receives the UNMOUNT event or shutdown request and notifies the Process Manager to terminate the associated Event and Scheduler Daemons. New file transfer requests are immediately rejected, but notification to the PM is normally delayed until any in-progress transfers have completed.

2.5.1.2. Gatekeeper,I/O Manager

The IOMs are started via the ***inittab*** process and/or the ***initctl*** command. They remain in standby mode until they detect that the GPFS file system is mounted. An IOM will spawn GHI-HTAR processes to do aggregate data transfers.

2.5.1.2.1 Performance

The configuration of the I/O Managers and attached devices can have a large impact on the performance of GHI because of constraints imposed by a number of factors e.g., device channel bandwidth, network bandwidth, processor power. The IOM configuration is largely dictated by whether a site runs the processes on an NSD node, or a GPFS Client node. In the case where the the GPFS cluster is using a SAN configuration, and the NSDs can see all the GPFS data blocks, placing the IOM on the NSD nodes eliminates data transfers to gather the data blocks to send to the HPSS Mover.

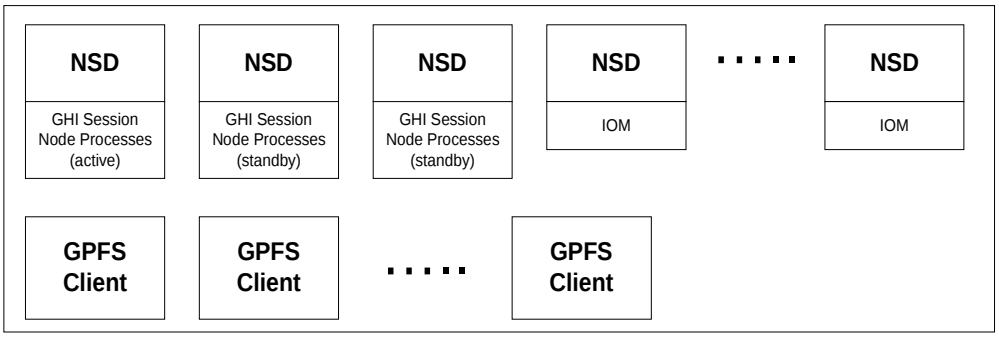

Figure 11 - IOM Layout – NSD Node Configuration

Figure 11 - IOM Layout – NSD Node Configuration shows an example configuration placing the IOMs on the NSD nodes. An IOM can run on any of the Session nodes, which is not depicted in Figure 11. There is a configuration option such that the IOM is active on the standby nodes, and if the standby Session node becomes the active node, the IOM goes dormant and does not process any data transfers.

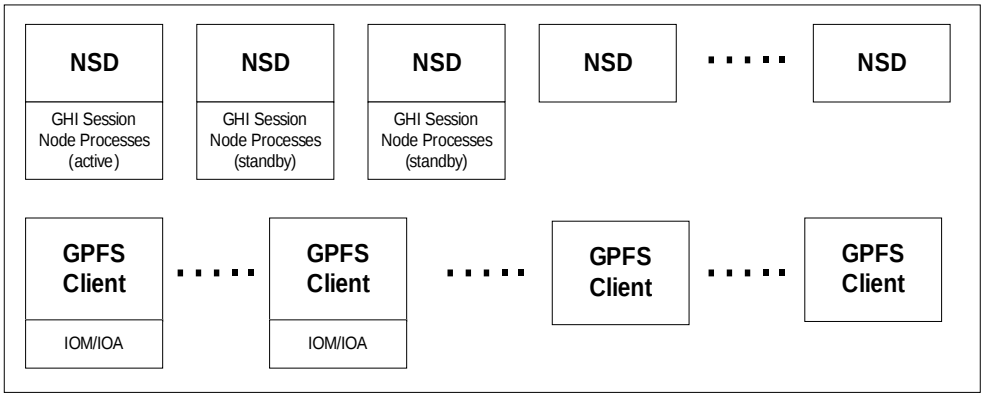

Figure 12 - IOM Layout – Client Node Configuration

Figure 12 - IOM Layout – Client Node Configuration shows an example of placing the IOMs to run on the client nodes.

For sites configuring their GPFS clusters to use locally attached devices, there will be network traffic generated for the majority of the data block transfers, since the IOMs will have to gather all the data blocks to send to the HPSS Mover.

A balanced configuration, tuned to optimize GPFS, GHI, and HPSS performance involves the coordinated input and consultation of system engineers with expertise from all three of these major components. Because changes in one part of the system can have a significant impact to other components, it is required that GPFS, GHI, and HPSS support personnel are consulted before updating the configuration or a site considers changes to the HW/architecture of the overall system.

To determine the sizing for the number of IOMs required to achieve the data throughput required, refer to Figure 13 IOM Capacity.

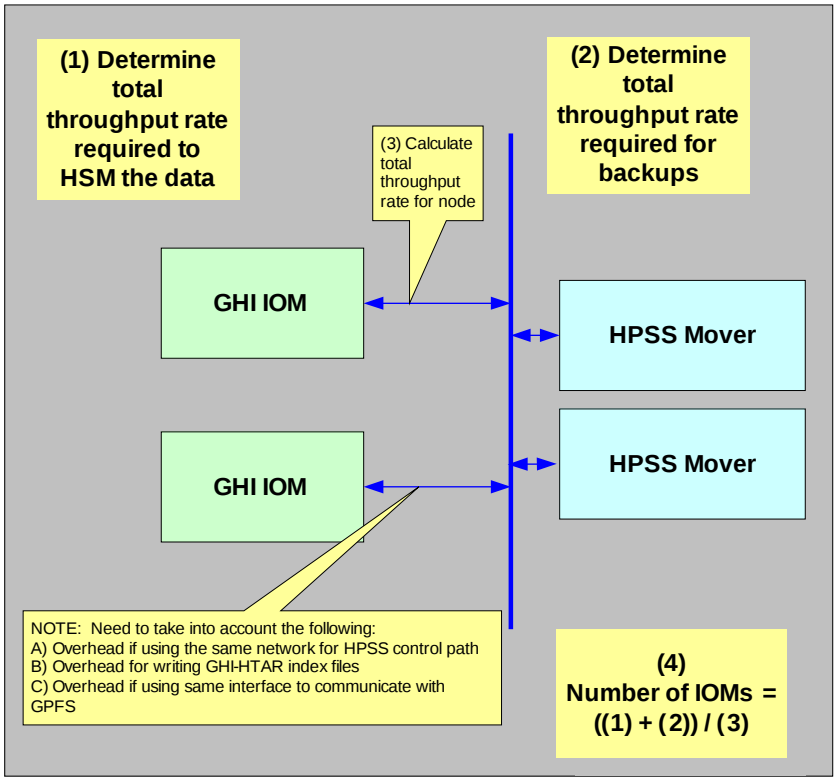

Figure 13 IOM Capacity

2.5.2. GHI-HTAR

Aggregation requires that GHI-HTAR runs on the nodes where the IOMs reside. The GHI-HTAR process is started via a script, ***htar.ksh***, that resides in the ***/opt/hpss/bin*** directory. The script is used to determine authentication of GHI-HTAR, as well as locate the GHI-HTAR executable.
A GHI-HTAR process is started when an aggregate is requested to be processed by an IOM. GHI-HTAR has GHI-specific interfaces to provide the ability to use DMAPI to perform GPFS data transfers.

2.5.3. GHI Policy Engine

The policy engine requires temporary storage space for pre-processing, sorting, and generating the output files to be used by GHI for performing the operations. For all policies except the threshold policy, the GPFS file system is used to store these files.

2.5.4. Logging Service

Logging Services are comprised of the GHI Log Daemon, which is spawned from the Process Manager on the Session node.
Log messages from all GHI servers will be written by the Log Daemon to a common log file. There is a single Log Daemon process per GHI cluster system. The Log Daemon toggles between two log files. When a log file fills up, the file is archived into HPSS.

2.6. HPSS Storage Characteristics for GHI

This section defines key concepts of HPSS storage and the impact the HPSS storage on GHI configuration and operation. These concepts, in addition to the policies described above, play a significant role with the usability of GHI.
Before a GHI system can be used, the administrator must create a description of how the system is to be viewed by the HPSS software. This process consists of learning about the intended and desired usage of the system from the GPFS users and then using this information to determine GHI/HPSS hardware requirements and the configuration of the hardware to provide the desired performance. The process of organizing the available hardware into a desired configuration results in the creation of a number of HPSS metadata objects.

2.6.1. Storage Classes

A Storage Class is used by HPSS to define the basic characteristics of storage media. These characteristics include the media type (the make and model), the media block size (the length of each basic block of data on the media), the transfer rate, and the size of media volumes. These are the physical characteristics of the media. Individual media volumes described in a Storage Class are called Physical Volumes (PVs) in HPSS.

2.6.2. Classes of Service

Class of Service (COS) is an abstraction of storage system characteristics that allows HPSS users to select a particular type of service based on performance, space, and functionality requirements. Each COS describes a desired service in terms of characteristics such as minimum and maximum file size, transfer rate, access frequency, latency, and valid read or write operations. A file resides in a particular COS which is selected when the file is created. Underlying a COS is a storage hierarchy that describes how data for HPSS user files in that class are to be stored in the HPSS system. A COS can be associated with a fileset such that all files created in the fileset will use the same COS.
Each GHI file, which appears to HPSS as a user file, belongs to a single Class of Service (COS) which is selected when the file is created. There are three classes of GHI files written to HPSS. They are as follows:

- ***Data files (aggregate and non-aggregate files).*** By default, these files use the Class of Service Maximum File Size Hints information passed to HPSS when the file is created. The 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, eg., "OPTS '-c 5'".

  NOTE: Each GHI file belongs to a single Class of Service (COS) which is selected when the file is created. The ***Force Selection*** flag can be set in the COS definition on the HPSS GUI to prevent automatic selection. If the flag is set, that COS will not be selected for storing the GHI data files.
- ***Aggregate index files.*** These files are written to HPSS using the "Aggregate

Index COS" in the GHI configuration as a default. 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>'", e.g., "OPTS '-c auto: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 relationship between storage class, storage hierarchy, COS is defined in the *HPSS Installation Guide* as well as the *HPSS Management Guide*.

2.6.3. File Families

File families are an abstraction of storage system characteristics that allows HPSS users to associate "like" files with a set of tapes. A file resides in a particular file family which is selected when the file is created (for a tape-only COS), or when a file is migrated from disk to tape based on the HPSS hierarchy.
There are three classes of GHI files written to HPSS. They are as follows:

- ***Data files (aggregate and non-aggregate files)*.** By default, these files are not associated with a file family. However, the policy can be defined to specify the file family by adding a "OPTS '–f <file family>:auto'" or "OPTS '–f <file family>'" in the policy for aggregates and non-aggregates respectively.

  NOTE: Each GHI file belongs to a single file family which is selected when either the file is created (tape-only COS), or when the file is migrated from disk to tape based on the HPSS storage hierarchy.
- ***Aggregate index files*.** By default, these files are not associated with a file family. However, the 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 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.

  NOTE: Each GHI file belongs to a single file family which is selected when either the file is created (tape-only COS), or when the file is migrated from disk to tape based on the HPSS storage hierarchy.
- ***Backup files*.** There is currently no way to associate backup files with a file family.

The relationship between storage class, storage hierarchy, COS and file families are defined in the *HPSS Installation Guide* as well as the *HPSS Management Guide*.

2.6.4. Storage Subsystems

Storage subsystems are provided in HPSS for the purpose of increasing the scalability of the system, particularly with respect to HPSS Core Servers. An HPSS system consists of one or more subsystems, and each subsystem contains its own Core Server. If multiple Core Servers are desired, this is accomplished by

configuring multiple subsystems.
This can be used to separate HPSS resources for files being migrated via GHI vs. other files being written to HPSS. Only 1 subsystem is supported per GPFS file system.

2.7. DB2

A DB2 database on the HPSS Core Server node is used to store GHI metadata. The metadata describes each of the GHI backups. Since the GHI database resides on HPSS, the backup procedures for the HPSS metadata will need to be modified to backup the GHI database as well.
Integrity of the DB2 backup images is important to HPSS, GHI and GPFS recovery. Mismanagement or corruption in the backup images could impede recovery. Multiple copies can be created, validated, and managed with each set of log files. It is recommended that backup image copies are placed on a physically separate disk and disk controller from the primary copies. That way, the disk and disk controller cannot be a single point of failure.

2.8. GHI Security Considerations

The security requirements between GHI customer environments differ widely. The GHI System Administrators must be aware of the sites security requirements and should be aware of the security configuration required in HPSS. GHI Administrators should contact their site security representative if they have questions regarding security. For more information on security, see Chapter 2: Security and System Access of the HPSS Management Guide.

2.9. Technology Insertion

As new types of digital storage technology are configured into the system, the HPSS Storage Class definition may be updated to the new device and media characteristics. Existing file contents are accessed normally, but all new file data migrations will use the updated definitions and new media.

2.10. Policy Considerations

When GPFS processes a policy, the policy's rules are processed in the order in which they appear in the policy. This means that GPFS will not select/reject a file covered by rule 2 until all processing for rule 1 has been completed. 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 or reject the greatest number of files are first.
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 that these two directories contain only a small fraction of the total files on the file system.

3. GHI COMMANDS

3.1. ghiapplypolicy

ghiapplypolicy <file system mountpoint> -P <policy file> <other mmapplypolicy arguments>
The ghiapplypolicy command is a wrapper for the GPFS mmapplypolicy. The command accepts all the mmapplypolicy arguments. If the arguments are not supplied, ghiapplypolicy will use the configured values instead.
Default values are:

-g <mount point>/scratch/.ghi
-f <mount point>/scratch/.ghi
-B <Max Aggregate Size from ghi.conf>
-N /var/hpss/ghi/etc/ghinode.conf

3.2. ghi_admin

ghi_admin
The ghi_admin command allows the administrator to cancel transfer requests, halt an in-progress backup, and reset the regions/dmapi attributes for a file. This tool should not be used without guidance from a GHI support representative.
The following features are available:

- Lookup a file in the Scheduler Daemon queue
- Cancel transfer request for a file (currently not supported)
- Delete a file from the Scheduler Daemon queue
- Move a file to the top of the Scheduler Daemon Queue
- Reset the DMAPI Extended attributes and the Managed Regions for a file
- Stop a backup that is in progress
- Reinitialize the GHI Server processes
- Get the status of transfer requests for files in filelist
- Cancel transfer requests for files in a filelist (currently not supported)
- Retrieve file data that resides in HPSS
- Reset the PM from “config in progress” status

3.3. ghi_backup

The ghi_backup command takes a backup of the GPFS file system. The command takes a snapshot of the GPFS file system, migrates any unmigrated files using the /var/hpss/ghi/policy/<file system>/backup_migration.policy, and then gathers the metadata for the GPFS file system with a second policy run. Version 2.4 of GHI only supports full backups of the entire file system (with the exception of any GPFS filesets which are unlinked at time of backup).

The actual backup taken by ghi_backup is of the file systems metadata, i.e., the namespace and attributes of each file in the namespace, e.g., size, owner, last

modification time, etc. Prior to backing up the metadata, ghi_backup will execute a GPFS ILM policy to attempt to migrate any non-migrated files into HPSS and will also wait on any other on-going migrations to finish. Thus, at a successful conclusion of ghi_backup, all file data should be written to HPSS as well as all file system metadata.
Any attempt to execute ***ghi_backup*** on a GHI read-only file system will be rejected.

3.3.1. Image backup

ghi_backup <file system> image
Calling ghi_backup with the image argument will start an image backup. Image backups use the GPFS image functionality to copy the GPFS metadata directly into HPSS. A special GHI image writer process is used to copy the data.

This command is a replacement for the old ghi_backup <filsystem> full command.

3.3.2. Namespace Backup

ghi_backup <file system> full
Calling ghi_backup with the full command will start a namespace backup. During a NS backup GPFS will generate a listing of the file system. GHI will take this list and capture the metadata for all the files listed.

This command has been replaced by ghi_backup <filesystem> image.r

3.3.3. File sets

ghi_backup <file system> image -E <file sets>
ghi_backup <file system> image -E -F <list of file sets>
The ghi_backup command requires that all file sets in the file system be linked. File sets can be excluded from the check by using the -E argument. If a file set is linked but not excluded the backup will proceed but generate a warning.

3.4. ghi_backup_manager

ghi_backup_manager <file system>
The ghi_backup_manager command allows the user to delete individual backups for a GPFS file system. The user is given a list of backups to choose from and then prompted to confirm the deletion.
When a backup is deleted any files that are no longer referenced by a GPFS file system or backup will be deleted from HPSS.
Backup manager cannot be used on a GHI read only file system. Also, if the [full access] file system has an associated read only FS, the list of backups which may be chosen for deletion will be limited to those which were taken after, i.e., have a DB2 BU Index larger than, the backup which was restored to the read only FS.

3.5. ghi_df

ghi_df <file system>
The ghi_df command allows the user to gather information about the GPFS file system. The command will run a GPFS ILM policy and shows the user:

- Number of and space used by scratch area files.
- Number of and space used by non managed files.
- Number of and space used by managed files
- Number of purged files.

3.6. ghi_ls

ghi_ls [-a] [-C] [-e|E] [-f] [-h|-H] [-l] [-n] [-R] [-u] [file…]
The ghi_ls command is similar to the UNIX ls command. It produces the same output as the ls command but adds additional information. The utility interacts directly with GPFS via DMAPI without HPSS, so users can determine the residency of their file data without regard to the availability of other HPSS services.
The residency of a file is expressed by a single letter a the start of the output:

G: The file is GPFS resident and has not been migrated to HPSS.
B: The file is dual resident. The data exists in both GPFS and HPSS.
H: The file is HPSS resident. The file data has been purged from GPFS.

The file residency indicator will be followed by a **P** if the file is also pinned (a blank ‘ ‘ if not pinned):
HPSS data is requested with the -e(E) and -h(H) options. Other UNIX-traditional output is displayed by using the traditional ls command line options. For example the “-l” option can be used to generate a listing that contains the file residency plus permissions, timestamps, uid, and gid.
If invoked against a file, it will list the state of the file. If invoked against a directory, it will list the state of all files in that directory

Command line options specific to ghi_ls are:

-h(H) Displays where the file resides in HPSS along with if it is an aggregate file.
-C Displays in ‘classic’ mode, which mimics the formatting of ls when the requested items include directories or for a recursive display. The “new” and default display mode is to omit display of directories, and display only files with complete pathnames.

-e(E) Displays the HPSS file attributes included bytes stored on the different Storage classes.

For more information about the supported command line options you may type ”ghi_ls -?” at the command line or see the ghi_ls man page.

3.7. ghi_mon

ghi_mon <file system> [iom|sd] [-a action] [-f frequency] [-p output path]
The ghi_mon command allows the user to start the GHI monitoring of the system or IOM statistics. The output is described in more detail in section 3.5 System Monitoring

3.8. ghi_pin

ghi_pin [-v] [-u] {file | wild card | -f filelist}
The ghi_pin command allows the administrator to flag a GPFS file so that it does not purged during a threshold policy run. This will ensure that the data remains on the GPFS file system. The administrator may specify a single filename, filenames with wildcards, or a file list. When a file is pinned, ghi_ls will include a 'P' after the file residency indicator.

3.9. ghi_restore

ghi_restore –g <scratch FS mount point> <file system>
The ghi_restore command allows the administrator to select a backup to restore to the GPFS file system. This will rebuild the GPFS namespace and DMAPI attributes used by GHI to map files to HPSS. No file contents are restored.

It is up to the site administrator to generate a recall ILM policy based on restoring files in priority order. The **ghi_stage** command can also be used to restore file data. A file's data will also be restored on-demand if a user attempts to access the file, e.g., to 'vi' it.

To to a test or verfication restore, the file system must have been created with the -r option. ghiIf the file system to be restored has an associated GHI read only file system, the list of backups which may be chosen for restore will be limited to the backup which has been restored to the read only FS and those taken thereafter, i.e., have a DB2 BU Index which is larger. If it is desired to restore to a backup which pre-dates that on the associated read-only FS, either the read-only FS will need to be destroyed or first restored to the same backup to the one restored to the full-access FS or to an even earlier backup.

3.10. ghi_stage

ghi_stage [-t timeout] [–v] {file|directory|-f filelist}
The ghi_stage command allows the user to stage files from HPSS without running a policy or waiting for a DMAPI event to complete. The command allows the user to input a list of files to stage. The files must be in the same GPFS file system. Optional arguments are:

-t timeout This is how long the command will wait for the stages to complete.

–**t 0** means don’t wait
No argument means wait forever.
-v This tells ghi_stage to enable verbose output.

3.11. ghi_state

ghi_state <file system>
The ghi_state command displays the state of the system. The following information is displayed:
Cluster status.
Disk status.
Mount status.
Manager status.

4. GHI MANAGEMENT

4.1. Start/Stop GHI Servers

The GHI PM, LD, MD, and CM session node "daemon" processes are designed to start automatically whenever GPFS is started. This is done by using the GPFS call back functionality. File system specific session node "daemon" processes, i.e., the ED and SD, are started by the PM via the MD whenever the associated file system is mounted.
All GHI session node "daemon" processes are stopped by GPFS whenever it shuts down.
The IO Managers are started on each node at boot-time by the 'init' (PID = 1) process, which thereafter monitors them for loss and restarts them if they go away.
The "ghishutdown" script provides a way to do a forced-restart on one or more IOMs.
GHI provides two scripts to manually start and stop GHI. These scripts may also be used to start and stop GPFS. They are "ghistartup" and "ghishutdown". They may be executed from any node in the cluster on which they've been installed.

4.1.1. ghistartup

ghistartup [-g]
ghistartup does the following:

1. If executed with '-g', issue an "mmstartup -a" on the local node to bring up GPFS. (If GPFS is already running on one or more nodes in the cluster, mmstartup will so indicate.) Pause for five seconds for GPFS to begin operation after mmstartup completes.
2. With or without '-g', query GPFS for the current GPFS cluster manager, and if the query doesn't return success, exit after reporting that GPFS is not running.
3. If '-g' was not specified, query the cluster manager to see whether or not the GHI PM is currently running, and if not, start the PM.
4. If '-g' was specified, assume that GPFS will start the PM.
5. Wait for up to 60 seconds for a PM process to appear on the cluster manager and report its process ID.

4.1.2. ghishutdown

ghishutdown [-f] [-g]
ghishutdown [-f] -i { ALL | <FS_name> } [<node> ...]
The first form is used to bring down GHI and possibly also GPFS. The second form, with the '-i', is used to force a restart [via the 'init' process] of the IO Managers for some combination of filesystems and nodes. Options '-g' and '-i' are mutually-exclusive.
If '-i' was not specified (shutdown GHI/GPFS), ghishutdown does the following:

1. Query GPFS for the current GPFS cluster manager, and if the query doesn't return success, exit after reporting that GPFS is not running (and by extension, GHI).
2. Query the cluster manager to see whether or not the GHI PM is currently running, and if not, so indicate in a message to the user and proceed to step (5).
3. If '-f' was not specified, send a SIGTERM to the PM to instruct it to do an orderly shutdown of GHI that includes waiting for any on-going transfers to or from an HSM to complete. This waiting is known as "quiescing the FS". While the FS is being quiesced, no new transfer requests will be accepted, and any requests which have been queued but not yet started will be terminated with error code -19 (ENODEV). Then, wait for up to 60 seconds for the PM to terminate and disappear from the cluster manager. If this does not occur after 60 seconds, exit after reporting inability to stop GHI.
4. If '-f' was specified, send a SIGUSR1 to the PM to instruct it to do an immediate shutdown. Then, wait for up to 10 seconds for the PM to terminate and disappear from the cluster manager. If this does not occur after 10 seconds, issue a message so indicating.
5. If '-g' was specified, issue a GPFS "mmshutdown -a" command.

The script may be executed without the '-f' as often as needed until GHI (and GPFS) comes down.

If '-i' was specified (restart IOMs), ghishutdown does the following:

1. Determine the list of nodes on which to restart IOMs. If one or more nodes were specified, they are the list. Else, if the FS name is "ALL", the list is every node on which GHI has been installed and configured, or (FS name not "ALL") the list is every node on which the IOM(s) for the specified FS has been configured.
2. Determine the signal to be delivered to targeted IOMs. If '-f' was not specified, it will be SIGTERM, which will result in the IOM attempting an orderly shutdown. Else, if '-f' was specified, a SIGKILL will be sent, which will result in an immediate shutdown.
3. For each node in the list of nodes create in step (1):
    a. Display the node name.
    b. Query for either all currently-running IOMs (FS name = "ALL") or for an IOM associated with the FS.
    c. For each IOM found to be running on the current node in the list:
        i. Display a message with its process ID and an indication that it is being restarted.
        ii. Send to the IOM the signal determined in step (2) and wait for up to 10 seconds for the init (PID = 1) process to sense termination of that IOM process and start a new one. When a new IOM process is detected, output a message to indicate a successful restart, or if a new IOM is not detected after 10 seconds, issue a message that the restart might not have been successful.

When only the node name output in step (3)(a) is displayed, with no corresponding output from step (3)(c), this indicates that no targeted IOM was running on that

node. This is not necessarily an indication of a problem, especially if the FS name was "ALL" and no nodes were specified.

4.2. GHI Process Failure/Recovery

GHI needs to provide a fault tolerant system in order to keep the file system online and available. GPFS 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 migrate 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, quorum loss).
The events will invoke either a single "user" defined script only, an HA/NFS defined script only, or both. The scripts will be invoked on those nodes that have the exit script installed.

4.2.1. Node Failures

4.2.1.1. Session Node

The node defined as the Session node is selected by GPFS when the system is brought online. It is typically the node that is the GPFS cluster configuration Manager node. GHI will utilize the GPFS heartbeat mechanism to monitor the nodes in the cluster that are potential Session node candidates. During startup, GPFS will execute the script, ***hpssEventNotify***, to start all GHI processes and mount the file systems. Likewise, during failure, GPFS will execute the script, ***hpssEventNotify***, to unmount the file systems and stop all GHI processes. If the node fails, and another node needs to take over, GPFS will select the new Session node.

4.2.1.2. Manager Node

The nodes running the I/O Managers start the processes using ***inittab***. The I/O Managers, once started, will remain idle until the file system is mounted on that node. If a file system is subsequently un-mounted, its associated IOM will go back to being idle.
If an I/O Manager node fails, there are two scenarios:

1) The Scheduler will lose the connection to the I/O Manager, and will cancel all requests to the failed I/O Manager and send them to a new I/O Manager that is active.
2) If a policy script was being run on the node that failed, the policy manager will be notified that the request failed. In the case of a backup, the backup will have to be rerun. In the case of a migration/recall/purge, no user action will need to be taken.

4.2.1.3. Client Node

There is no special failover logic for these processes. If the node fails, the I/O Manager will detect a completion failure of transferring the data. There is retry

logic in the IOM to retry the data transfer, if the request is for a non-aggregate. For GHI-HTAR requests, the IOM does not spawn off a new GHI-HTAR process if the return from GHI-HTAR indicates a failure.

4.2.2. Single Process Failures

4.2.2.1. ILM Client

These 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 detect that the HPSS process has terminated abnormally, the process will be restarted. The new process will start processing the policy file from the beginning.

4.2.2.2. Process Daemon

In the case of an error or termination of GPFS on the GHI Session node, the ***hpssEventNotify*** script will be executed. Running this script will shutdown the GHI processes. The script will notify the Process Manager to shut the other processes down, and then terminate itself.

4.2.2.3. Mount Daemon

Failure of the Mount Daemon will impact the file system by failing to allow file systems to mount. If the Mount Daemon abnormally terminates, the Process Manager will automatically restart it. There is no special recovery logic for this process.

4.2.2.4. Log Daemon

Failure of the Log Daemon will impact the file system by failing to log critical log messages. If the Log Daemon abnormally terminates, the Process Manager will automatically restart it. There is no special recovery logic for this process.

Failure of the Mount Daemon will impact the file system *mount* and *unmount* requests. Those requests that are not handled will simply hang, and the user will need to kill and retry the *mount* or *unmount* requests. *Mount* and *unmount* requests can only be handled when the Mount Daemon is registered to receive those DMAPI events.

If the Mount Daemon abnormally terminates, the Process Manager will automatically restart it. There is no special recovery logic for this process. It will wait for new *mount/unmount* requests.

4.2.2.5. *GHI Configuration Utility* Daemon

Failure of the *GHI Configuration Utility* Daemon will impact system operability by inhibiting the capability to make changes to the configuration and by loss of periodic cross-cluster consistency checking. If the *GHI Configuration Utility* Daemon abnormally terminates, the Process Manager will automatically restart it. There is no special recovery logic for this process

4.2.2.6. Event Daemon

The Event Daemon will be 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 un-mounted.
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 will automatically restart it. Upon restart, it will assume the current Session ID, and check for outstanding DMAPI events. The ED will add the events to the internal queue and then wait for responses from the Scheduler. It will then do normal processing and wait for new DMAPI events.

4.2.2.7. Scheduler Daemon

The Scheduler Daemon will be 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 un-mounted. If the Scheduler process abnormally terminates, the Process Manager will restart it. There is no special recover 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.

4.2.2.8. I/O Manager

I/O Managers will be started using the ***inittab*** on the IOM host systems. If the I/O Manager abnormally terminates, it will be automatically restarted by ***inittab***. It will

then wait for new requests from the Scheduler. If the Scheduler Daemon detects that an I/O Manager has abnormally terminated, it will attempt to re-assign its workload to another IOM(s). There is no special recovery logic for this process.

4.2.1. Multiple Process Failures

4.2.1.1. ILM Client and Scheduler

If one or more ILM clients abnormally terminate and the Scheduler terminates as well, the original request will have to be resent when the client and Scheduler are restarted.

4.2.1.2. 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 will send any outstanding DMAPI requests to be processed. Those requests will be 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.

4.2.1.3. Scheduler and I/O Manager

If the Scheduler and one or more I/O Managers abnormally terminates, the Process Manager will restart the Scheduler, and the I/O Manager(s) will be restarted by ***inittab***. All outstanding requests being processed by the I/O Manager, which was abnormally terminated, will have to be re-tried by the application. The Event Daemon will resend all DMAPI requests, which may cause duplicate stages being performed.

4.2.2. HPSS Unavailability

When HPSS is unavailable in the GHI system, most file system operations will continue to work. The operations that require data to be transferred between GPFS/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 GPFS.
  - When a user requests the file through a DMAPI event, an abort will be sent to the application.
- All policy manager runs.
  - 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.

4.3. System Monitoring

GHI provides a monitor utility, ***ghi_mon***, to watch the major activity on the system. GHI currently supports watching the Scheduler Daemon and the IOM progress. Monitoring can be scheduled to start when the SD and/or IOMs are online. This is done by configuring it via the *GHI Configuration Utility* (see section "4.4. GPFS Configuration Management"). Otherwise, the user can call ***ghi_mon*** <***task***> to start the task.

4.3.1. Scheduler

The following figure depicts the internals of the Scheduler.

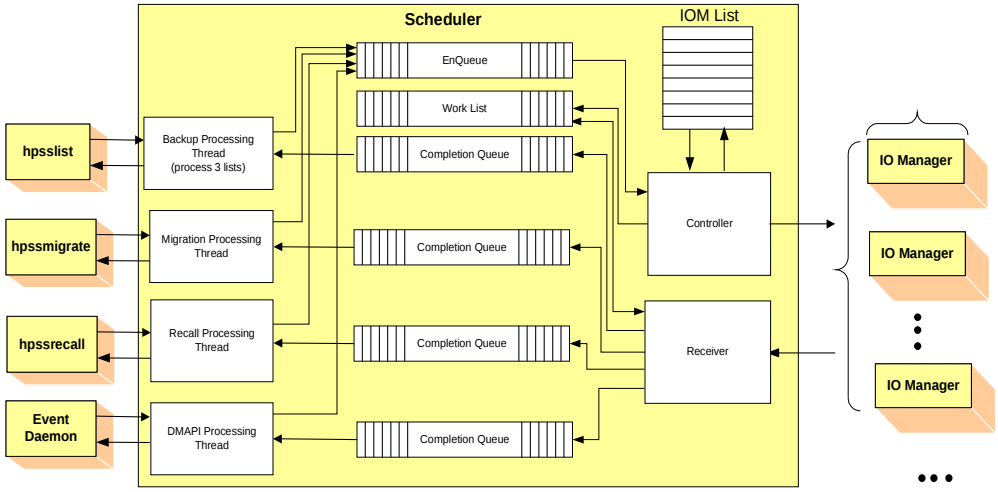

Figure 14 - Scheduler Internals

The Scheduler will contain four different Schedule Queues:

- Backup requests.
- Migration requests.
- Recall requests.
- DMAPI requests.

As requests come in, the Scheduler will place 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 will be printed out 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 Schedule Queue.
- ***Working***: Current number of entries being worked off.
- ***Migrations(A/N)***: Total numbers of aggregates/non-aggregates that have been processed since the Scheduler was started up 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/purges that have failed.
- ***IOM(A/T)***: Total number of IOMs that are active/Total number of IOMs configured.

4.3.2. I/O Manager

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

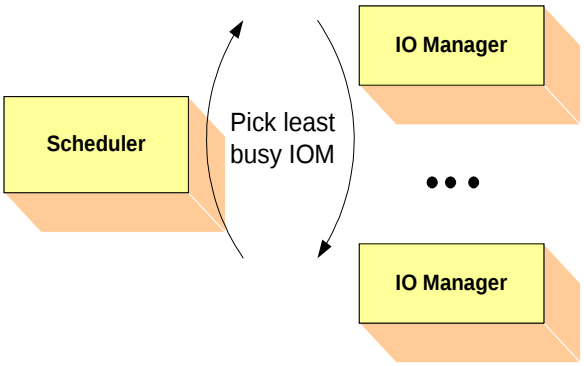

Figure 15 - I/O Manager Internals

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 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 the connection to the IOM.

When monitoring the IOMs, one or two lines will be printed out for each IOM configured. The first line will contain the following information:

- ***State***: The current state of the IOM (see node definitions above)
- ***Requests***: Number of requests (total and by request type) completed and that the IOM is currently working.
- ***Errors***: Number of errors (total and by request type) encounter since the IOM was started.
- ***Processing Rate:*** How busy the I/O is. If a '*" is shown, that indicates

that I/O Manager will be selected next for work.

- ***Node***: Hostname/Port where IOM is running.

If the IOM is currently transferring a file, a second line will be displayed to provide information on the longest-running active transfer.

4.3.3. File-Transfer Performance Monitoring

The GHI File-Transfer Performance Monitoring capabilities allow file transfers to be performance-profiled at various stages in the transfer process to help in determination of location of possible bottlenecks. Statistics will be 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 an execution of the ghi_stage utility.

The processing stages to be monitored are specified via the GHI Configuration Utility (see Performance Logging (ghichfs --perf)), and are:

ED - extends from the ED's receipt of a DMAPI event from GPFS until it returns the response back to GPFS. The reported elapsed time will include all processing needed to generate the response, i.e., the SD, IOM, and PIO or HTAR processing.

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, i.e., the process's lifetime, will include all processing by the SD, IOM, and PIO or HTAR. Execution of ghiapplypolicy may result in launching of multiple ILM processes -- each with its own reporting trail.

SD - extends from 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 will include processing by the IOM and PIO/HTAR.

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 will include the PIO or HTAR processing.

HTAR - for aggregate file transfers, extends from IOM's launching of the HTAR process until it detects that the process has terminated. HTAR does all the DMAPI processing, so this will not be reported under IOM.

PIO - for non-aggregate file transfers, extends just around the actual PIO processing to effect the data transfer. PIO does no DMAPI processing, so this will be reported under IOM.

4.3.3.1. Extracting the Processing Trail for a Transfer Request

The following paragraphs provide information which will allow for gathering of records generated by any given file transfer request. In order to do this for ILM

policy runs, it may be necessary to run with '-d' or '-D' specified in the policy's OPTS field so that generated filelists remain available after process completion.
The first thing to note is that as a file transfer request gets processed, one or more records per each of the above types may get written to the performance log file. For example, an ILM run (ghiapplypolicy) on non-aggregated files will result in one ILM and one SD record for each ILM process that gets spawned, and 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. ED and SD records are always written on the GHI session node, while ILM, IOM, PIO, and HTAR records are written on the node on which the process runs. When multiple ILM processes get spawned from a ghiapplypolicy run, the processes may be on multiple nodes. The SD parcels out work to the IOMs independent of the request source.
And lastly, the order of records within a file may not correspond to actual processing order. For instance, the ILM record may precede an IOM record, or an IOM record may precede an HTAR record, even though one would expect the IOM to complete its work and write its record prior to reporting its status back to the SD, which in turn reports status back to the ILM -- which would then write its record (likewise with the IOM and HTAR). Underlying timing uncertainties with parallel processing account for this seeming effect-before-cause behavior.
Keeping all the above in mind...multiple queries may need to be made, as the transfer identifier may have varied as the transfer proceeded thru 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 file of interest has the inode and igen, and 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 use of '>>' 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, asks on each node may not be in sync with each other.

4.3.3.2. Performance Log Record Formats

The following paragraphs describe the format for each of the record type which may be written to the performance log file, i.e., ED, ILM, SD, IOM, PIO, and HTAR. Additionally, at GHI startup, daemon restart, or refresh via SIGHUP, each daemon that will be writing performance data to the log outputs a timestamp record similar

to:

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

The '----' indicates a non-measurement daemon-startup timestamp record, with the timestamp following the '----'. Following the timestamp is the daemon's PID, the daemon type (ED, SD, or IOM), and the final field is the associated mountpoint. As with GHI daemon logs, each IOM writes to its own local performance log file.

4.3.3.2.1 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 GPFS to determine that the file is not already resident in GPFS and to be able to make the recall request to HPSS. The newly-created record will contain the file data and current (wall-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, along with the current time for the end time, 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

Fields are:

timestamp	-	Date/time 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

The record format will be identical for both aggregates and non-aggregates, because DMAPI requests are non-differentiated between the two GHI file types.

For example (line break is an artifact of MS-WORD):

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

4.3.3.2.2 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 (wall-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, along with the current time for the end time, and the completed record written to the performance log file. The ILM record format is:

ILM timestamp PID operation status 0 elapsed_time filelist

Fields are:

timestamp	-	Date/time 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

For example (line break is an artifact of MS-WORD):

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

4.3.3.2.3 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 essentially 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’s 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 (wall-clock) time for the start time. When the transferring IOM indicates transfer-complete, the response data from the IOM will be added to the record, along with the current time for the end time, 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

Fields are:

timestamp - Date/time 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 request being for an individual file or a filelist
  file: RqstID: *RqstID* Inode: *Inode*, Igen: *Igen*, SnapID: *SnapID*
  filelist: RqstID: *RqstID* Filename: *pathname*, Size: *size*

For example (line break is an artifact of MS-WORD):

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

4.3.3.2.4 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 will need 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 (wall-clock) time for the start time. When the transfer is completed, the transfer data will be added to the record, along with the current time for the end time, and the completed record written to the [IOM’s local] performance log file. The IOM record format is:

IOM timestamp PID operation status size elapsed_time file_data

Fields are:

timestamp	-	Date/time 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 request being for an individual file or a filelist
		file: RqstID: *RqstID* Inode: *Inode*, Igen: *Igen*, SnapID: *SnapID*
		filelist: RqstID: *RqstID* Filename: *pathname*, Size: *size*

For example (line break is an artifact of MS-WORD):

```
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
```

4.3.3.2.5 **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 processing with HPSS. The newly-created record will contain the inode and igen of the non-aggregate file, and the current (wall-clock) time for the start time. As soon as PIO processing is completed, the transfer data will be added to the record, along with the current time for the end time, and 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

Fields are:

timestamp	-	Date/time 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 request being for an individual file (migrate, recall) or a filelist (list)
		file: RqstID: *RqstID* Inode: *Inode*, Igen: *Igen*, SnapID: *SnapID*
		filelist: RqstID: *RqstID* Filename: *pathname*, Size: *size*

For example (line break is an artifact of MS-WORD):

```
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
```

4.3.3.2.6 HTAR Transfer Record

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

HTAR timestamp PID operation status size elapsed_time throughput file_data

Fields are:

timestamp	-	Date/time 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*

For example (line break is an artifact of MS-WORD):

```
HTAR 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
```

4.4. GPFS Configuration Management

The GHI configuration is managed via a set of commands which together comprise the *GHI Configuration Utility*, each command being used to manage a set of configuration items. Each configuration item and command will be covered in detail after the following general discussion pertaining to GHI configuration and the semantics of command entry.

Two GHI configuration commands, ***ghicrcluster*** and ***ghiupdate***, are only used when installing or upgrading a GHI installation and are covered in the *GHI Install Guide*.

GHI configuration data is stored in files named "ghi*conf*" within "$HPSS_GHI_PATH". These files should never be edited unless so-directed by GHI support personnel. The *GHI Configuration Utility* commands maintain the integrity of these files and "by-hand" editing leaves room for error.

The *GHI Configuration Utility* Daemon (the "ghi_cm" [CM] server process that runs on the GHI session node), does a one-daily consistency check of the configuration across the cluster. The check occurs in two stages: first is the session node check, in which the working copies of these files are compared with saved copies and the any non-comparing working copy overwritten with the saved copy. Next, the working copies on each node in the cluster are compared with the working copies on the session node and updates pushed from the session node if needed. This activity should not take more than a few seconds to complete and may also be executed on-demand by sending a SIGHUP to the CM process.

4.4.1. General Discussion of GHI Configuration

GHI configuration can be broadly divided into three areas of decreasing effect: cluster-wide, FS-specific, and IOM-specific. Cluster-wide commands are: ***ghilscluster***, ***ghichcluster***, ***ghilsnodes***, ***ghiaddnode***, ***ghidelnode***, ***ghilsfsdefaults***, and ***ghichfsdefaults***. Changes to cluster-wide configuration items can adversely affect performance across the entire GHI cluster. ***ghilscluster*** and ***ghichcluster*** are used to manage configuration items, while ***ghilsnodes***, ***ghiaddnode***, and ***ghidelnode*** are used to manage nodes within the cluster. ***ghilsfsdefaults*** and ***ghichfsdefaults*** are used to manage the initial configuration used when adding a file system.
FS-specific commands are: ***ghilsfs***, ***ghiaddfs***, ***ghichfs***, and ***ghidelfs***. These commands are used to manage the Event and Scheduler Daemons associated with the FS, as well as to provide general configuration for the associated IOMs and to define how the filesystem's data will be stored into HPSS. Configuration changes made via these commands generally do not adversely affect portions of GHI beyond the particular FS to which they are applied.
And finally, the IOM-specific commands are: ***ghilsiom***, ***ghiaddiom***, ***ghichiom***, and ***ghideliom***. These are used to manage individual IOMs. Configuration changes made via these commands generally do not adversely affect portions of GHI beyond the particular IOM to which they are applied, although they can adversely affect performance of the node on which the IOM resides.
All of the *GHI Configuration Utility* commands may be executed from any GPFS node in the cluster. A command begins execution 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 temp files. The ***ghils**** commands do nothing more than pull data from the temp files and "pretty print" it for display. The other commands apply the requested updates to the temp files and then push the updated files to all GHI nodes to replace their configuration files. As a command executes, it maintains a record of processing steps executed since start-up. If an error should occur, it uses this list to undo all changes made prior to detection of the error.
All *GHI Configuration Utility* commands that can result in modification to the GHI configuration, i.e., 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, its use is highly recommended in case an error occurs and the automated error-recovery fails and must be effected manually. Verbose output includes the record of processing steps mentioned in the preceding paragraph along with the steps undertaken in automated error-recovery. (The ***ghils**** do also take the '-v' option, but the only added output is a display as it makes the call to the "ghi_cm" [CM] server process that runs on the GHI session node.)
These commands require both GPFS and GHI to be running, and they execute such that no other *GHI Configuration Utility* command can run while they are running. They do this via a locking mechanism built into the GHI *GHI Configuration Utility* Daemon (the "ghi_cm" [CM] server process that runs on the GHI session node). "ghi_cm" also does a one-daily consistency check of the configuration across the cluster and this activity, which should not take more than a few seconds to complete, counts as an executing *GHI Configuration Utility* command.

Each GHI configuration item consists of a ‘Key’, ‘Description’, and ‘Value’. The Value may include a comment to be used however desired. (See the discussion of ‘date-stamped history’ at the end of this section). The Key is in the form ‘--xxx’, where ‘xxx’ is a short alphanumeric string. The Description is a longer string, which may contain spaces as well as special characters. The format of the Value depends on what data is being conveyed, and may be a string, number, boolean, member of a list, etc. If the Value contains a ‘#’ preceded by at least one space, the preceding space(s), “#”, and whatever follows makes up the comment, which except for storage and retrieval is otherwise ignored.

Each of the ***ghich**** commands take “<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/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 being included. Note that single or double quotes are necessary with a comment since the shell process in the terminal window would most likely interpret an un-quoted ‘#’ as a comment to the command and not pass it along as a parameter.

The second way of supplying a <key_value> uses 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/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.

It is anticipated that to reduce typing when entering commands at the terminal, the Key form will be used to specify a <key_value>. And, the Description form -- which is more self-documenting -- will be be used for better readability from within a script.

For example, to change the maximum number of concurrent connections allowed to the Log Daemon (Description = “LD Max Connections”) to 75 , enter the following command:

```
% ghichcluster -v --ldmaxc "75 # CR 475"
<verbose output skipped>
Key        Description          Value (# comment)
-------------------------------------------------------------------
----
--ldmaxc   LD Max Connections   75   # CR 475
Distributing updated GHI config to all GHI nodes...
<verbose output skipped>
Done.
```

When a command that can result in modification to the GHI configuration

completes normally, it displays “Done.” as shown above. Otherwise, it will display a message to indicate the error and attempt to back out any changes as stated earlier. For example:

```
% ghichfs --bucos
Usage: ghi_ch_fs [-Hv] <FS> [-c "# <comment>"] [ <key_value> ... ]
*** s have been made ***
```

If it occurs prior to any permanent changes being made, there’s nothing to be backed-out and “*** No changes have been made ***” is displayed.
If an error occurs after processing has progressed to where permanent changes have been made anywhere in the cluster, the message
“!!! ABORTING -- UNDOING CHANGES !!!” is displayed and the command enters back-out processing.
A few configuration items are not displayed with a Key by the corresponding ***ghils**** command, which signifies either that they cannot be modified once set or that their entry doesn’t follow as described above. For example, configuration item “GHI Version” (displayed via ***ghilscluster***) cannot be changed (without upgrading GHI). Therefore, the following command would fail:

```
% ghichcluster “GHI Version = 12345”
Invalid key - "GHI Version"
*** No changes have been made ***
```

IOMs are specified using a pseudo Key or Description. They are specified as *--Node*:*Port* for the ‘Key’ (e.g. --node3:8023) or if the ‘--‘ is omitted, it becomes a ‘Description’. Given the following:

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

An IOM’s node or port cannot be changed with ***ghichiom***. The IOM must be deleted and re-added via ***ghideliom*** and ***ghiaddiom***.)
As mentioned earlier in this section, 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 underlying files 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>

<date> is when the change was made, and <old_value> and <old_comment> (which includes the ‘#’) are what was in place for the configuration item. If it is desired that a history of a change not be maintained, each of these commands takes option ‘-H’ to so indicate.

4.4.2. GHI Configuration Items

The following sections comprise a list of all configuration items, grouped by their associated *GHI Configuration Utility* commands. The title of each section is the configuration "Description" for each item followed in parentheses by the particular ***ghich**** command and key needed to effect a modification. To see the current value for an item, issue the corresponding ghils* command, (e.g. ghilscluster for "LD Max Connections"). For the configuration items modified by ***ghichfs***, it is to be understood that the new file system default value for that item is modified [listed] via ***ghich[ls]fsdefaults***.
For all configuration items which start with the prefix "number of ...", acceptable values are integers greater than zero unless noted otherwise.

4.4.2.1. LD Max Connections (ghichcluster --ldmaxc)

Number of connections the Log Daemon can have. The value should be the number of connections to the PM, MD, EDs, SDs, and IOMs and any administrative utility.

4.4.2.2. LD Thread Pool Size (ghichcluster --ldtps)

Number of threads that will be working off incoming log messages.

4.4.2.3. LD Request Queue Size (ghichcluster --ldrqs)

Number of slots in the wait list. The wait list is used to keep the overflow requests when the thread pool fills up.

4.4.2.4. LD HPSS Base Path (ghichcluster --ldbase)

Location within HPSS of where to store the Central Log files.

4.4.2.5. Level Of Detail To Log (ghichcluster --log)

Amount of detail to be logged by GHI processes. To set, create a space or comma separated concatenation of the following values:
Event: Log general events such as "process initialized", "process terminated" These log messages are written to both the central log and the process log. Operator intervention should not be required.
Minor: Log events for minor errors such as "Failed to create a directory". These log messages are written to both the central log and the process log. These kinds of errors are not in and of themselves expected to be more than 'inconvenient', but they may lead to additional problems. Operator intervention is probably not immediately required.
Major: Log events for major errors such as "Failed to open file". These log messages are written to both the central log and the process log. These kinds of errors signify that some substantial GHI processing or capabilities are definitely lost or malfunctioning. Operator intervention will probably be required.
Critical: Log events for critical errors such as "Failed to start Event Daemon". These log messages are written to both the central log and the process log. These errors signify loss of multiple capabilities and GHI will probably require immediate

operator intervention.
Info: Log events for information for administrators, such as "Sending stripe group info to clients". These are not error messages These log messages are written only to the process log. Operator intervention should not be required.
Debug: Log events and data used to debug error conditions. These log messages are written only to the process log. Normally meaningful only to GHI support personnel, and operator intervention is not required except at their direction.
Trace: Log events for entry and exit of routines. These log messages are written only to the process log. Normally meaningful only to GHI support personnel, and operator intervention is not required except at their direction.
They can be expressed in any order.
Two additional values are allowed, which are mutually-exclusive with each other and with all of the above-listed values:
All: Set logging to all possible values.
None: Set logging so that nothing is logged.
Each GHI process' log is limited to two files of 10MB each. When the currently-active log file reaches 10MB, the other log file is activated by truncating it to length zero and log messages are sent to it.

4.4.2.6. Min Files To Make Aggregate (ghichfs --minagg)

The minimum number of files to be placed in an aggregate. This value, together with "Max Files Per Aggr" (see next section) determines whether or not all files selected for aggregation during a migration are actually migrated. Selected files will be migrated into an aggregate in "Max Files Per Aggr" bulks until the number remaining to be migrated is under this value. If at least "Min Files To Make Aggregate" files remain to be migrated, they will be placed into an aggregate; else, they will remain un-migrated until a subsequent migration policy is executed. During backups, this minimum value will not have any affect, such that all selected files get migrated.

4.4.2.7. Max Files Per 12Aggregate (ghichfs --maxagg)

The maximum number of files to be placed in an aggregate. See the discussion in the preceding section.

4.4.2.8. Aggregate Index COS (ghichfs --aggcos)

The HPSS class of service to be used to store the GHI-HTAR index files.

4.4.2.9. Aggregate Thread Pool Size (ghichfs --aggtps)

The thread count to be used by GHI-HTAR Refer to the –T option in GHI-HTAR.

4.4.2.10. Backup Bulk Count (ghichfs --bbc)

The number of files to be distributed to an IOM to be backed up in a single backup file. This allows for distribution of the processing during a backup.

4.4.2.11. Backup COS (ghichfs --bucos)

The HPSS class of service to be used to store namespace and attribute files (metadata files) created as part of a backup.

4.4.2.12. HPSS Junction (ghichfs --junct)

The HPSS Junction name is used to link to the fileset of a subsystem in the HPSS namespace. It must not be blank, use forward slash when not specifying any directory.

4.4.2.13. HPSS Base Path (ghichfs --basep)

The high-level directory to store migrated files into HPSS. This will be placed directly under "HPSS Junction".

4.4.2.14. HPSS Backup Path (ghichfs --bupath)

The high-level directory to store backup namespace and attribute files (metadata files) into HPSS. This will be placed directly under "HPSS Junction".

4.4.2.15. Performance Logging (ghichfs --perf)

The type of file-transfer performance logging to produce. To set the type of logging to be produced, create a space or comma separated concatenation of the following values:
ED: Log performance of the ED's handling of DMAPI events.
ILM: Log performance of of policies executed via the 'ghiapplypolicy' script.
SD: Log performance of the SD's handling of data transfer requests.
IOM: Log performance of the IOM's handling of data transfer requests.
HTAR: Log performance of the HTAR's handling of data transfers into/out of aggregates.
PIO: Log performance of the PIO portion of non-aggregate data transfers.
They can be expressed in any order.
Two additional values are allowed, which are mutually-exclusive with each other and with all of the above-listed values:
All: Set to produce all possible types of performance logging.
None: Set to produce no performance logging whatsoever.
The usual setting is "None". Performance logging can result in exceedingly large data files because unlike the GHI error logs, which are capped at 10MB each, the performance log has no upper limit to the size it can attain. For this reason, performance logging is normally enabled only to gather supporting data if performance issues are suspected.
See File-Transfer Performance Monitoring for details on using this capability.

4.4.2.16. Purge Only If On Tape (ghichfs --poiot)

This setting determines whether or not GHI will ensure that HPSS has rolled-off to tape a migrated file before it will execute a purge ("hole punch") request on that file. The possible values are "true" and "false". A setting of "false" means that all

files selected for purging will be purged as long as they have been migrated into HPSS regardless of whether or not HPSS has rolled them to tape. A setting of "true", which means checking with HPSS for every to-be-purged file, is resource-intensive.

4.4.2.17. Purged File Size (ghichfs --pblock)

The number of data bytes of file data to remain resident in the 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 GPFS.

4.4.2.18. ED Max Connections (ghichfs --edmaxc)

Number of concurrent open connections the Event Daemon will allow. This should be the number of connections to the SD and any administrative utility.

4.4.2.19. ED Thread Pool Size (ghichfs --edtps)

Number of threads that will be working off the DMAPI events.

4.4.2.20. ED Request Queue Size (ghichfs --edrqs)

Number of slots in the wait list. The wait list is used to keep the overflow requests when the thread pool fills up.

4.4.2.21. 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.

4.4.2.22. IOM Max Connections (ghichfs --iommaxc)

Number of concurrent open connections the IOM will allow. This should be the number of connections to the SD and any administrative utility.

4.4.2.23. IOM Thread Pool Size (ghichfs --iomtps)

Number of threads that will be working off the DMAPI events.

4.4.2.24. IOM Request Queue Size (ghichfs --iomrqs)

Number of slots in the wait list. The wait list is used to keep the overflow requests when the thread pool fills up.

4.4.2.25. IOM PIO Blocksize (ghichfs --iopiob)

This is used for performance logging of PIO transfers. Transfers are broken up into blocks of this size, with an entry made to the performance log upon completion of each block. Acceptable values are an integer greater than zero optionally followed by a units indicator (KB, MB, GB, etc.).

4.4.2.26. IOM Monitor Flag (ghichfs --iomon)

This flag indicates that the SD should perform IOM activity monitoring (value = “on”) or should not be performed (value = “off”).

4.4.2.27. 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>.

4.4.2.28. 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 which is 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 x 50) x 6 = approximately 60K bytes per minute.

4.4.2.29. SD IOM Max Connections (ghichfs --simaxc)

Number of connections the Scheduler can have. The value should be the number of connections to the IOMs and any administrative utility. The connections to the IOMs are short lived connections, so the value can be slightly smaller.

4.4.2.30. SD IOM Thread Pool Size (ghichfs --sitps)

Number of threads that will be working off the DMAPI events.

4.4.2.31. SD IOM Request Queue Size (ghichfs --sirqs)

Number of slots in the wait list. The wait list is used to keep the overflow requests when the thread pool fills up.

4.4.2.32. SD Client Max Connections (ghichfs --scmaxc)

These connections are used by the ILM client worker threads. The number of ILM clients is based on the number of files to be transferred divided by the bulk count.

4.4.2.33. SD Client Thread Pool Size (ghichfs --sctps)

Number of threads that will be working off the DMAPI events.

4.4.2.34. SD Client Request Queue Size (ghichfs --scrqs)

Number of slots in the wait list. The wait list is used to keep the overflow requests when the thread pool fills up.

4.4.2.35. 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.

4.4.2.36. SD Monitor Flag (ghichfs --sdmon)

This flag indicates that the SD should perform queued, working, and completed-items monitoring (value = “on”) or should not be performed (value = “off”).

4.4.2.37. 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>.

4.4.2.38. 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 (i.e. 1/4K byte) will be written to the monitor output log.

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

An IOM’s node or port cannot be changed. To effect a change in either, the entire IOM must be deleted and re-created via ***ghideliom*** and ***ghiaddiom***.

4.4.2.40. Active Session Node (ghichiom --asn)

Flag with values of “true” or “false”. A value of “true” indicates the IOM should be allowed to run whenever the node is the GHI Session Node. This flag is ignored whenever the IOM node is not also the GHI Session Node.

4.4.2.41. 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, etc.).

4.4.3. GHI Configuration Commands

The following sections provide details of each *GHI Configuration Utility* command. For general information on usage and error-handling, refer to the discussion at the start of this major section.

4.4.3.1. ghilscluster

ghilscluster
Generates a display of cluster-wide configuration items.

```
% ghilscluster
Key        Description                 Value (# comment)
-------------------------------------------------------------------
----
           GHI Version                 2.4
--ldmaxc   LD Max Connections          400
--ldtps    LD Thread Pool Size         400
```

```
--ldrqs    LD Request Queue Size (?)   400   # xxx
--ldbase   LD HPSS Base Path           /ghi_log/miami
--log      Level Of Detail To Log      EVENT CRITICAL MAJOR MINOR
```

4.4.3.2. ghichcluster

ghichcluster [-Hv] <key_value> ...
Used to modify cluster-wide configuration items. No special usage instructions apply. Configuration changes will not be picked up by the GHI server processes until the next time GHI starts up or until the ***ghi_admin*** command is used to re-initialize them.

```
% ghichcluster --ldmaxc "75 # CR 475"
Key        Description          Value (# comment)
-------------------------------------------------------------------
----
--ldmaxc   LD Max Connections   75   # CR 475
Distributing updated GHI config to all GHI nodes...
Done.
```

4.4.3.3. ghilsnodes

ghilsnodes [-c <node>]
Generates a display of GHI nodes. Node[name] and Type are as returned by the GPFS 'mmlscluster' command.

```
% ghilsnodes -c wor
Node                                Type
-------------------------------------------------
miami.clearlake.ibm.com             quorum
worcester.clearlake.ibm.com
   IOM for FS gpfs1
```

4.4.3.4. ghiaddnode

ghiaddnode [-v] <node> is readable by the UNIX
Used to add a node to the GHI configuration. The node to be added must already be configured in GPFS. <node> must be specified such that a **grep -E "^<node>(\.| $)"** of the "Daemon node name" or "Admin node name" output of the 'mmlscluster' command results are not empty.
The specified node will be added to the configuration. If it is to be used as an IOM , that configuration needs to be performed separately.

4.4.3.5. ghidelnode

ghidelnode [-fvd] <node>
Used to delete a node from the GHI configuration. A prompt for confirmation of the delete request will be presented for confirmation unless the '-f' option is specified, in which case the deletion will proceed without further user intervention. The '-d' option is for deferring sending of configuration updates to nodes besides the GHI session node. To speed up deletion of multiple nodes, a final ***ghidelnode*** command would not include the '-d', which results in configuration updates being pushed to all remaining GHI nodes in the cluster.
The node to be deleted must not be configured as an IOM for any file system. Use

ghideliom to remove configuration of any still-configured IOMs. The current GHI session node can not be deleted. The following example shows an attempt to delete a node which 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 miami.clearlake.ibm.com
Retrieving from GHI session node -- /var/hpss/ghi/etc/ghi.conf
  scp miami.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
miami.clearlake.ibm.com:/var/hpss/ghi/etc/ghi_gpfs3fs.conf
/tmp/ghi_gpfs3fs.conf
fresno.clearlake.ibm.com is still referenced within configuration
for gpfs3fs!
!!! ABORTING -- UNDOING CHANGES !!!
  TMP_FILE:  -- /tmp/ghi_gpfs3fs.conf
  TMP_FILE:  -- /tmp/ghi.conf
```

4.4.3.6. ghilsfsdefaults

ghilsfsdefaults

Used to display the configuration that will be applied whenever ***ghiaddfs*** is used to create a new file system. Once an FS is created, ***ghichfs*** can be used to modify its configuration as necessary.

Any of these default configuration items can contain a "?xxx" to signify that it serves as a template to be set by ***ghiaddfs*** based on its command-line parameters. 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".

4.4.3.7. ghichfsdefaults

ghichfsdefaults [-Hv] [-c "# <comment>"] [<key_value> ...]

Used to set the configuration that will be applied whenever ***ghiaddfs*** is used to create a new file system. Once an FS is created, ***ghichfs*** can be used to modify its configuration as necessary.

Any of these default configuration items can contain a "?xxx" to signify that it serves as a template to be set by ***ghiaddfs*** based on its command-line parameters. 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". The "# <comment>" is comment text which will be applied to the "?FS_Name" whenever the ***ghiaddfs*** command is executed.

4.4.3.8. ghilsfs

ghilsfs [<FS> ...] [<config_item> ...]
Generates a display of the specified file system (FS)-specific configuration items for the specified FS(s). If no configuration items are given, then the complete list will be displayed. If no FSs are specified, then the display will be for all FSs configured on this cluster. If neither is specified, the display will be the names of all FSs configured on this cluster.

4.4.3.9. ghiaddfs

ghiaddfs [-v] <FS_name> [-r <FA_FS>] <mount_point> [<SD_port> <ED_port>]
Used to add a file system to the GHI configuration.
<FS_name> and <mount_point> must each be unique among all GHI clusters which connect to any given HPSS system.
The default SD end 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 for the first-configured FS, 8020 and 8021 for the second-configured FS, and so on. The actual configured port numbers will be the first available ports starting with the default. If the file system being added is number is the 10th or after, then the port numbers will need to be explicitly specified because port numbers are limited to 65535.
The FS to be added must be known to GPFS and unmounted.
Without the '-r', the FS will be full-access. To configure a read-only FS, include the '-r' and <FA_FS>, which is the <FS_name> of the full-access FS with which this read-only FS will be associated. <FA_FS> must have already been configured and may reside on the same or a different GHI cluster.

4.4.3.10. ghichfs

ghichfs [-Hv] <FS_name> [-c "# <comment>"] [<key_value> ...]
Used to alter the configuration of a file system. Neither the FS's name nor its mount point can be modified. To change them the FS has to be deleted and re-created (via ***ghidelfs*** and ***ghiaddfs***). All other configuration items can be modified. The "# <comment>" is a comment text which will be applied to "<FS_Name>", replacing whatever may already exist.

4.4.3.11. ghidelfs

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

4.4.3.12. ghilsiom

ghilsiom [-t] <FS_name> [<IOM>]
Generates a display of IOM-specific configuration items for a particular FS, either the complete configuration for a single IOM or a summary configuration for all configured IOMs.

4.4.3.13. ghiaddiom

ghiaddiom [-vd|D] <FS_name> [-c "# <comment>"] <IOM_node>[:<port>] <active_on_session_node> <est_XFER_rate>
Used to add an IOM to the GHI configuration. The '-d|D' option is for deferring sending of configuration updates to nodes besides the GHI session node and the node on which the new IOM is being configured. The '-D' option also inhibits re-starting of the Schedule Daemon associated with <FS_name>. To speed up adding multiple IOMs, a final ***ghiaddiom*** command can be executed without specifying '-d|D' option. This would result in the updated configuration being pushed to all GHI nodes in the cluster. The Schedule Daemon will also being re-started to make use of the updated IOM configuration.
<FS_name> is the name of the file system for which the IOM is being configured.
<IOM_node> is the node on which the IOM will run, optionally on the associated ":<port>". The default port is 80x2, where 'x' is the order in which file systems were configured, i.e., 8012 for the first-configured FS, 8022 for the second-configured FS, and so on. The actual configured port numbers will be the first available ports starting with the default. If the file system being added is the 10th or after, then the port numbers will need to be explicitly specified because port numbers are limited to 65535.
<active_on_session_node> is either 'true' or 'false', and indicates whether or not the IOM is to be activated at GHI start-up if <IOM_node> should also happen to be the current GHI session node.
<est_XFER_rate> is the expected throughput, in bytes/second, expressed as an integer, optionally followed by units of KB, MB, or GB.

The "# <comment>" is comment text which will be applied to the entire IOM.

4.4.3.14. ghichiom

ghichiom [-Hv] <FS_name> <IOM> [-c "# <comment>"] [<key_value> ...]
Used to update the configuration for an IOM. Neither the IOM's node nor port can

be altered, to do either of these requires the IOM to be deleted and re-created (via ***ghideliom*** and ***ghiaddiom***). But, all other configuration items can be modified. The "# <comment>" is comment text which will be applied to the entire IOM, replacing whatever may already exist.

4.4.3.15. ghideliom

ghideliom [-fvd|D] <FS_name> <IOM>
Used to delete an IOM from the GHI configuration. A prompt for confirmation of the delete request will be presented for confirmation unless the '-f' option is specified, in which case the deletion will proceed without further user intervention. The '-d|D' option is for deferring sending of configuration updates to nodes besides the GHI session node and the node on which the IOM was configured. The '-D' option also inhibits re-starting the Schedule Daemon associated with <FS_name>. To speed up deletion of multiple IOMs, a final ***ghideliom*** command would not include the '-d|D' option, which would result in the updated configuration being pushed to all GHI nodes in the cluster. The Schedule Daemon will be re-started to make use of the updated IOM configuration.

4.4.4.

4.5. Upgrade DB2

Use the DB2 upgrade instructions to upgrade to the required DB2 version. Verify the DB2 levels on the GHI nodes are synchronized with the version on HPSS.

4.6. Upgrade GHI

Please refer to the "GHI Install Guide" for upgrade instructions.

4.7. Upgrade GPFS

Refer to the GPFS Administration and Programming Reference to upgrade GPFS.

Site administrators must coordinate with their GHI support representative to get concurrence before upgrading GPFS to a newer version or PTF level.

4.8. Upgrade HSIGWD/GHI-HTAR

Refer to the http://www.mgleicher.us/HSI_Admin URL for upgrade instructions.

4.9. Upgrade HPSS

Refer to the *HPSS 7.3 Conversion Guide* for upgrade instructions.
Once the software for the HPSS Client API is upgraded, the GHI software needs to be rebuilt. The libraries and executable files will then need to be distributed to all the nodes running GHI processes.

4.10. Daily Monitoring of the System

To monitor the system, it is recommended that sites perform the following actions on a daily basis:

- Save output from each of the policy runs to be reviewed. Since policy runs can take several hours, it is recommended to save the output for each of the runs in a specific location to be reviewed on a daily basis. Policy runs generate **.ok* and **.exc* files; Policies use the "*-b*" option to save these files. It is up to the site administrator to delete these files.
- Review the output from a backup on a daily basis in detail. Output will indicate success or failure, but the details need to be reviewed to acknowledge how critical individual failures are.
- Monitor SD and IOM output generated from the *ghi_mon* utility. By default, the output is activated during system startup time. Refer to the *ghi.conf* configuration file.
- Monitor the output in the central log files: */var/hpss/ghi/log/logfile*.log*.

5. PROBLEM DIAGNOSIS AND RESOLUTION

This chapter provides problem determination and resolution advice for GHI infrastructure components, servers, and user interfaces. Note that a problem may have more than one diagnosis and resolution.
GHI logs only event, minor, major and critical log messages by default, and sends these to the central log file in the /var/hpss/log directory.
To alter the logging, the following steps need to be performed:
Run the ***ghichcluster*** command to set logging as desired.
The changes will take effect across all of GHI the next time GHI is re-started.
To make the changes to be immediately effective for a file system's SD and ED, unmount and then re-mount the FS on the GHI session node.

To make the changes to be immediately effective for certain process(es), e.g., an IOM, issue a ***kill -SIGHUP <pid>*** command to the process(es).

5.1. GHI Infrastructure Problems

The sections below describe possible RPC and Security infrastructure errors.

5.1.1. RPC Problems

5.1.1.1. One GHI server cannot communicate with another

***Diagnosis 1*:** The target server may not have registered its RPC endpoint properly.
***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.
***Diagnosis 2*:** A communications failure may exist or security may be disallowing communication.
***Resolution**:* First verify that 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**:* If ***/var/hpss is full, try to determine what is causing the file system to fill up. Common problems are ***/var/hpss*** being too small, or log files that are not being archived properly.
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, 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 being 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 other configuration issues.
Resolution: Verify the appropriate server configuration.

5.1.1.2. A server cannot obtain its credentials

Diagnosis 1: There may be a problem with the keytab table.
Resolution: Make sure the keytab table (usually ***/var/hpss/etc/***) is readable by the UNIX username under which the server is running. Make sure that the key contained in the keytab table is the correct one. Look for extra versions of the servers key; they can interfere with the authentication process.

5.1.1.3. A server cannot register its RPC info

Diagnosis 1: 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.

5.1.1.4. The connection table may have overflowed

Diagnosis 1: The server may be so heavily loaded that it is unable to free up connections easily.
Resolution: Reduce the load on the server. The problem may also indicate that a server is configured incorrectly, or that there is a software problem in handling connections properly. To solve the problem, increase the **Maximum Connections** parameter in the configuration for the specific server.

5.1.1.5. Servers cannot talk to one another

Diagnosis 1: 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 support then fix the DNS.

5.2. GHI Server Problems

The paragraphs below discuss problems common to all servers.

5.2.1. Process Manager Problems

5.2.1.1. The Process Manager dies after a mount request (PPC only)

Diagnosis 1: The Process Manager core dumps with a stack dump.
Resolution: Set the HPSS_:PTHREAD_STACLK=262144 in /var/hpss/etc/env.conf

5.2.2. Mount Daemon Problems

5.2.2.1. Failed to get events

Diagnosis 1: Failed to retrieve DMAPI events.
Resolution: Verify the file still exists. Also, verify the Session ID is valid for that node, and is not owned by another node.

5.2.2.2. Failed to respond to an event

Diagnosis 1: Failed to respond to file system mount/unmount request.
Resolution: Verify the file system still exists.

5.2.2.3. Failed to mount a file system

Diagnosis 1: Failed to mount file system.
Resolution: Verify the file system still exists and that it is not a GHI read-only file system in need of being restored to a backup of its associated full-access FS.

5.2.3. Event Daemon Problems

5.2.3.1. Failed to get events

Diagnosis 1: Failed to retrieve DMAPI events.
Resolution: Verify the file still exists. Also, verify the Session ID is valid for that node, and is not owned by another node.

5.2.3.2. Failed to respond to events

Diagnosis 1: Failed to respond to an event for a request accessing a file.
Resolution: Verify the process had not been restarted.

5.2.3.3. Failed to get attributes on a file

Diagnosis 1: When trying to get the attributes of a file, the call failed.
Resolution: Verify the file still exists. Also, verify the Session ID is valid for that node, and is not owned by another node.

5.2.4. Scheduler Daemon Problems

5.2.4.1. Stuck in "QUIESCING_FS" mode

Diagnosis 1: All file transfer requests are terminating with error code -19 (ENODEV) and the monitor output is showing the SD's state to be

"QUIESCING_FS". This is happening because 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 quiescing the FS but the FS could not be un-mounted. There may be ongoing file transfers to/from an HSM which must be allowed to complete before the SD will terminate and allow GHI shutdown to complete. If a command to unmount the FS has failed, attempt to remove the condition which is blocking its completion and re-try it, or try a forced unmount (first see the GPFS documentation on the "mmunmount" command). If a GHI shutdown command has failed, re-try 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 to the SD process on the GHI session node. Observe the SD monitor output to verify that the SD's state returns to active, i.e., the first 14 characters on the second line go to all blanks. This should not be done after a GHI shutdown because the status of other GHI system processes is indeterminate. This may be done after a failed FS unmount if the FS still shows as mounted and it is desired to continue normal operations and not attempt another unmount.

5.2.4.2. Out of completion queues

Diagnosis 1: 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.

5.2.4.3. Failed to set regions (punching a hole)

Diagnosis 1: Unable to set the regions for a file, dm_set_region.
Resolution: Verify the file still exists. Also, verify the Session ID is valid for that node, and is not owned by another node.

5.2.4.4. Failed to punch a hole in a file

Diagnosis 1: Unable to punch a hole in a file, dm_punch_hole.
Resolution: Verify the file exists.

5.2.4.5. Recovery started for an IOM

Diagnosis 1: 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.

5.2.4.6. Failed to get a DMAPI handle for a file

Diagnosis 1: When attempting to ready the DMAPI handle for file before performing a data transfer, the call failed.
Resolution: Verify the file has not been deleted. Also, verify the SessionID is still valid for that IOM.

5.2.5. I/O Manager Problems

5.2.5.1. The IOM is in ECONN mode.

Diagnosis 1: The IOM is initializing
Resolution: Wait until it finishes the connection logic. If it has not connected after a short time, contact your GHI Support Representative.
Diagnosis 2: The IOM is configured incorrectly.
Resolution: Fix configuration issues.
Diagnosis 3: The IOM is configured incorrectly in **/etc/inittab**.
Resolution: Verify the entry in inittab is correct, and if not, fix the entry and recycle ***inittab* (*kill –1 <inittab pid>).***
Diagnosis 4: The authentication configuration is incorrect in ***/var/hpss/etc***.
Resolution: Copy the ***/var/hpss/etc/*** directory from the master location.
Diagnosis 5: There is a time difference between the IOM node and the Session node that is greater than 5 minutes.
Resolution: Run an NTP daemon on all the nodes, or sync-up the date/time with the "date" command.

5.2.5.2. IOM is in STANDBY mode.

Diagnosis 1: IOM loses connection to the SD.
Resolution: Verify the Scheduler is running. Recycle the IOM.
Diagnosis 2: File system is not mounted on that node.
Resolution: Mount the file system on that node.

5.2.5.3. Failed to make a handle to a file

Diagnosis 1: Unable to get a handle for a file.
Resolution: Verify the file still exists. Also, verify the SessionID is still valid for that IOM.

5.2.6. GHI-HTAR Problems

5.2.6.1. GHI-HTAR fails to communicate with the HSIGWD

Diagnosis 1: Check the ndapi.log file. There is no output in the ndapi.log
Resolution: Telnet localhost 1217 and verify it appears to hang, and if not, ***/var/log/messages*** is a good place to look for problems. If so, then type ***"<cntl><enter>*** to get back to the telnet prompt and the "quit. When this happens, there should be output in the ***ndapi.log*** file.
Run a sample GHI-HTAR to verify it is correct: "***/opt/hpss/bin/htar.ksh –cvf /ghi/xxx /etc/motd***".

5.2.6.2. GHI-HTAR fails to run

Diagnosis 1: GHI-HTAR fails with "ndad_keytab_check: failed (code=22) for principal hpssdmg" found in the ndapi.log
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: GHI-HTAR fails with 18431.
Resolution: Unable to open file.
Diagnosis 3: GHI-HTAR fails 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: ***"/opt/hpss/bin/htar.ksh –cvf /ghi/testfile /etc/motd***".
Diagnosis 4: GHI-HTAR fails with -52 htar_GhiClose, Error setting managed regions".
Resolution: Verify the SessionID is correct.
Diagnosis 5: The HSIGWD executable is not found in ***/opt/hpss/bin***.
Resolution: Run "netstat –an | grep 1217" to verify it is listening correctly.
Diagnosis 6: GHI-HTAR fails to open the file in GPFS.
Resolution: Verify the file still exists.

5.2.6.3. GHI-HTAR appears to be hung or locked up.

Diagnosis 1: The location for the GHI-HTAR 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.

5.3. Policy Interface Problems

The paragraphs below discuss interface problems with running the policy for migrations and recalls.

5.3.1. Migration problems

These problems are displayed as output from the ***ghiapplypolicy*** run.

5.3.1.1. A "-1 makeXHandle" error was encountered

The output from a migration policy failed with a call to makeXHandle.
Diagnosis 1: 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 validity configuration scan and correction cycle. Recycle the IOM.

5.3.1.2. A "Failed to migrate files, RPCError = 0, rc = -1" error was encountered

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

5.3.1.3. A "-5 PIOXferMgr" error was encountered

The output from a migration policy shows a file failed with "-5 PIOXferMgr" error.
Diagnosis 1: 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.

5.3.1.4. A "-19 sd_quiesce_FS" error was encountered

The output from a migration policy shows a file failed with "-19 quiescing_FS" error.
Diagnosis 1: The FS 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 the FS is re-mounted or GHI is brought back up.

5.3.1.5. A "-28 PIOXferMgr" error was encountered

The output from a migration policy shows a file failed with "-28 PIOXferMgr" error.
Diagnosis 1: 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.

5.3.1.6. A "-78 PIOXfer" error was encountered

The output from a migration policy shows a file failed with a "-78 PIOXfer" error.
Diagnosis 1: The inetd was incorrectly configured.
Resolution: Refer to Section 5.2.5 - I/O Manager Problems for diagnosis and resolution.
Diagnosis 2: The HPSS Mover(s) are having issues.
Resolution: Verify the HPSS Mover(s) are green. Also, look at the HPSS ***local.log*** to see if there are any error messages.

5.3.1.7. GHI-HTAR failed

The output from a migration policy shows that GHI-HTAR failed.
See Section 5.2.6 - GHI-HTAR Problems for diagnosis and resolution.

5.3.2. Recall problems

These problems are displayed as output from the ghiapplypolicy run.

5.3.2.1. A "-19 sd_quiesce_FS" error was encountered

The output from a recall policy shows a file failed with "-19 quiescing_FS" error.
Diagnosis 1: The FS 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 the FS is re-mounted or GHI is brought back up.

5.3.2.2. A "-78 PIOXfer" error was encountered

The output from a recall policy shows a file failed with a "-78 PIOXfer" error.
Diagnosis 1: The inetd was incorrectly configured.
Resolution: Refer to Section 5.2.5 - I/O Manager Problems for diagnosis and resolution.

5.4. File System Problems

The sections below discuss problems encountered when reading or writing to the file system. It also discusses problems if the file system is filling up and files are not being purged.

5.4.1. Mounting file system problems

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 was able to take over the Session ID.
Diagnosis 3: The ***mount*** command returned an error for "can't read superblock"
Resolution 1: 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 2: The file system is a GHI read-only FS and no backup has yet been restored to it. To check, issue the following command:

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

The following result will confirm:

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

5.4.2. Threshold problems

5.4.2.1. Error indicating file is not managed by HPSS.

The output from a policy run shows a file failed with an error indicating that the file is not managed by HPSS.

Diagnosis 1: The purge policy that was configured is not excluding files that are already co-managed.
Resolution: Verify the policy has an entry as follows:
Rule "exclude_rule" EXCLUDE FROM POOL "system" WHERE MISC_ATTRIBUTES NOT LIKE '%M%'
This rule will exclude files that are not HPSS managed.

5.4.2.2. A file fails to purge data blocks from GPFS

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 as well as the location of the file (via ***ghi_ls***).
Diagnosis 3: Verify the file is larger than 1 data block.
Resolution: Files that are less than or equal to 1 data block will not be selected as purge candidates.

5.4.3. File read/write problems

5.4.3.1. Failed to read/write a file in the file system

Diagnosis 1: The request returns an Input/Output error.
Resolution: Verify neither the Event Daemon nor the Scheduler has been recycled recently. Verify the HPSS Mover(s) are green. Also, look at the HPSS ***local.log*** to see if there are any error messages.

5.4.3.2. Reading/Writing a file appears to hang

Diagnosis 1: The request returns a timeout.
Resolution: Find out where the file resides in HPSS. If it resides on tape, verify there is not an outstanding tape mount request, and there is a free tape drive.

5.5. GHI Utility Problems

5.5.1. General Utility Problems

Here is a list of items to check when a utility is not running as expected:

- Check command-line 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 that default arguments are being overridden when necessary. Many utilities use default values for several of their parameters. If the parameter is not overridden with a specific value, unexpected behavior may result.

5.5.2. ghi_mon Problems

5.5.2.1. The ghi_mon SD error count increases

Diagnosis 1: Numerous errors are encountered during migrations, recalls, stages, and/or purges.
Resolution: Review the actions being 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.

5.5.2.2. The ghi_mon IOM error count increases

Diagnosis 1: The errors displayed from ***ghi_mon*** indicate the IOM was failing when perform 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 the issues are. Also, review the local.log file to see what errors are occurring from the HPSS Movers.

5.5.2.3. The ghi_mon shows the SD restarted

Diagnosis 1: The Session node failed over
Resolution: View the central log to determine why the Scheduler was recycled. If the Scheduler Daemon terminates again, turn on additional logging, so that the next time the schedule is recycled, additional error information can be viewed.

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

Diagnosis 1: The FS has been unmounted and/or GHI is shutting down
Resolution: See Stuck in "QUIESCING_FS" mode.

5.5.2.5. Failed to connect to the SD

Diagnosis 1: The SD is not running because the file system is unmounted
Resolution: Mount the file system, and verify the SD is started.

5.5.3. Backup Problems

5.5.3.1. GHI backup cannot communicate with DB2

Diagnosis 1: DB2 is not running.
Resolution: Verify whether DB2 is running and restart as appropriate. 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.

5.5.3.2. Failed to backup a file from a snapshot

The output from a backup shows that GHI failed to backup a file's data from a snapshot.
Check the migration problem section to determine why the file failed to migrate.

5.5.3.3. Failed to backup namespace information

The output from a backup shows that GHI failed to backup a namespace file.
Diagnosis 1: This is caused by failure transferring the file to HPSS.
Resolution: Refer to Section 5.2.5 - I/O Manager Problems.

GLOSSARY OF TERMS AND ACRONYMS

ACL	Access Control List.
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.
co-managed	File data resides in both GPFS and HPSS.
Configuration	The process of initializing or modifying various parameters affecting the behavior of an 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 Name Space 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 sub-system 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.
Delog	The process of extraction, formatting, and outputting HPSS central log records.
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.
DRP	Disaster/Recovery Plan
ED	Event Daemon.
Event	A log record message type used to log informational messages (e.g. subsystem starting, subsystem terminating).
Export	An operation in which a cartridge and its associated storage space are removed from the HPSS system Physical Volume Library. It may or may not include an eject, which is the removal of the cartridge from its Physical Volume Repository.
File	An object than 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.
file system ID	A 32-bit number that uniquely identifies an aggregate.
FSID	File system unique identifier
GB	Gigabyte (2^{30}).
GPFS	General Parallel File System.
GHI	GPFS/HPSS Interface.
GHI-HTAR	Specially modified GHI-specific version of the HTAR program.
GSS	Generic Security Service.
Hierarchy	See Storage Hierarchy.
HPSS	High Performance Storage System.
HSI	Hierarchical Storage Interface.
HSIGWD	HSI Gateway Daemon.
HTAR	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.
I/O	Input/Output.
IOM	I/O Manager.
IP	Internet Protocol.
junction	A mount point for an HPSS fileset.

KB	Kilobyte (210).
LAN	Local Area Network.
LANL	Los Alamos National Laboratory.
LLNL	Lawrence Livermore National Laboratory.
MB	Megabyte.
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 and/or writable 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 and/or HPSS namespaces.
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).
name space	The set of name-object pairs managed by the HPSS Core Server.
NLS	National Language Support.
NSL	National Storage Laboratory.
Object	See Managed Object.
OSF	Open Software Foundation.

PB	Petabyte (2^{50}).
PM	Process Manager.
POSIX	Portable Operating System Interface (for computer environments).
RPC	Remote Procedure Call.
SCSI	Small Computer Systems Interface.
SNL	Sandia National Laboratories.
SSA	Serial Storage Architecture.
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) 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.

REFERENCES

- ***HPSS Installation Guide, Release 7.4.2.***
- ***HPSS Management Guide*** Release 7.4.2.
- ***HPSS User's Guide***, Release 7.4.2.
- ***HPSS Conversion Guide, March 2013,***Release 7.4.1.
- ***GPFS Data Management API Guide****, version 3.5*
- ***GPFS Administration and Programming Reference ,****version 3.5*
- ***GPFS Advanced Administration****, version 3.5*
- ***HTAR:*** http://www.mgleicher.us/index.html/, *version 5.0*
- ***HSI:*** http://www.mgleicher.us/index.html/, *version 5.0*
- ***POSIX 1003.1-1990 Tar Standard***

TSM TO GHI CONVERSION

Prerequisites

The following table summarizes the prerequisites for the deployment of the GHI 2.4 TSM conversion feature:

Prerequisite	Description
GPFS 3 .5 PTF 16	This is the level of GPFS tested for TSM conversion feature.
HPSS 7.4.2	See the HPSS documentation for HPSS required patches
HSI/HTAR 5.0	Deploy the HSI service on the HPSS Core server. HTAR must be configured on all GHI IOM nodes.
GPFS Mirror Mount point	This is the target mount point where users are going to be putting/deleting files from
GPFS or NFS Mount point	This is the source TSM file system. **It must be available on all the GHI nodes as a remotely mounted GPFS file system.** **The mount point MUST be directly under "/".**

GHI and TSM compatibility

GHI and TSM can not run on the same cluster. The TSM file system must be remotely mounted on the destination GHI cluster. Obtaining TSM Source Code

The TSM conversion code is included in the GHI 2.4 build.

Compiling GHI TSM conversion code

These instructions assume the HPSS and GHI code are placed into "/opt/hpss/src/" and "/opt/hpss/src/ghi."

To deploy the TSM code on a GHI cluster, follow these steps:

- Install the HPSS client
- Place the GHI code in:
  /opt/hpss/src/ghi
- Open the following file for editing:
  /opt/hpss/src/ghi/Makefile.macros

- Change the Makefile.macros SUPPORT_FOR_TSM setting as follows:
  SUPPORT_FOR_TSM = on

- Change to the following directory:
  /opt/hpss/src/ghi/tools/tsm_conversion

- Open the following files for editing:
  /opt/hpss/src/ghi/tools/tsm_conversion/ghi_tsm_copy.c
  /opt/hpss/src/ghi/tools/tsm_conversion/ghi_tsm_build_fs.c

- The **ghi_tsm_copy.c** and **ghi_tsm_build_fs.c** program contains the settings for the TSM mount point and GHI mount points. Modify the source code as follows:

  - The GHI source code is released with the **TSM_MNTPT** set to /vicepm. Change the **TSM_MNTPT** variable to be the TSM mount point of your system. The following is an example for a mount point on /tsm_mnt:
    #define TSM_MNTPT /tsm_mnt

  - The GHI source code is released with the **GHI_MNTPT** set to /ghi. Change the **GHI_MNTPT** variable to be the GHI mount point of your system. The following is an example for a mount point on /ghi33:
    #define TSM_MNTPT /ghi33

Note: A work request has been opened in GHI release management control systems to request implementation of specifying the TSM parameters as a configuration option.

- Change to the GHI source tree directory located at:
  /opt/hpss/src/ghi

- Execute ***make*** to build the GHI source product

  make clean clobber; make 2>&1 | tee make.out

- Continue with the GHI installation or upgrade.

Restarting GHI Services

Once GHI has been compiled to run on a TSM-enabled environment, GHI needs to be restarted. To restart GHI issue the following commands:

- Stop GHI
  - ghishutdown -g
- Restart the GHI IOMs
  - ghishutdown -i ALL
- Restart GHI
  - ghistartup -g

The TSM file system

The TSM file system must be available locally on each of the GHI nodes. It can be a NFS mount that is available on all nodes or a GPFS remote mount point (different from the GFPS-TSM mirror). The mount point **must** be directly under the root path "/".

Building GPFS-TSM mirror file system

Before creating the GHI file system mirror of the TSM file system, start GPFS and GHI and make sure you have compiled and restarted GHI for the TSM mode as described on the Compiling GHI TSM conversion code and Restarting GHI Services sections of this document.

Execute the following steps:

1. Build a copy of the TSM namespace on the TSM managed cluster using the GPFS policy located at /var/hpss/ghi/policy/build_fs.policy. This must be run on the TSM cluster as GPFS does not support running policies on remotely mounted clusters.

   mmapplypolicy <filesystem> -P /var/hpss/ghi/policy/build_fs.policy –I defer

   The policy is designed to select every file in the file system. Files that should be excluded from the conversion should be excluded from selection in the policy.

2. Run the conversion script.

   /opt/hpss/bin/ghi_tsm_build <shell command> <file list>

a) The shell command should be the method used to communicate between the GHI nodes. Either SSH or RSH

b) The file list must be stored in a file system that is accessible from all the nodes in the cluster.

Backups and Restores of GPFS

There are no special procedures for backing up or restoring a GPFS file system that was built using a TSM conversion.

DEVELOPER ACKNOWLEDGMENTS

The GHI feature of HPSS was developed by IBM Global Business Services – Federal. HPSS is jointly owned and developed by the HPSS Collaboration consisting of Lawrence Berkeley National Laboratory, Lawrence Livermore National Laboratory, Los Alamos National Laboratory, Oak Ridge National Laboratory, Sandia National Laboratories, and IBM.

We would like to acknowledge IBM Almaden Research Center for the software development collaboration between GPFS and HPSS to implement the GPFS/HPSS Interface.

We would like to acknowledge NERSC, the National Energy Research Scientific Computing Center, for their help with initial design and development.

We would like to acknowledge Gleicher Enterprise, LLC, for providing a modified GHI-specific version of GHI-HTAR to support GHI file aggregation.

We would like to acknowledge SDSC, San Diego Supercomputing Center, for their help with the initial requirement review.

We would also like to acknowledge HLRS, High Performance Computing Center of the University of Stuttgart, and NCSA, National Center for Computing Applications, for providing testbeds for the initial GHI release.