Overview of GANESHA's Logging Mechanisms

jbongio@us.ibm.com

September 8, 2010

GANESHA is capable of fine granularity filtering of log messages, support for multiple log destinations (files, console, syslog, etc.), support for sending different log messages to different log destinations, and many ways of setting log parameters.

These options allow GANESHA the flexibility to be used in multiple professional environments while the default settings as well as the

1 Overview of Logging

Each log message in GANESHA relates to a specific component and severity level. A component is a label created with the intent of capturing a sectin of code or a specific function in GANESHA. Severity level determines the importance of a log message. The verbosity of logging can be changed per component as well as by the severity level of messages. Log messages can be sent to either the console, syslog, or an alternative file. The default logging destination is syslog and the default log level for all components is NIV_EVENT.

A typical log message in syslog will look like the following:

```
Sep  8 02:31:05 localhost nfs-ganesha[11555]: [main] :NFS STARTUP: EVENT: \
Configuration file successfully parsed
```

The first bit with the date, hostname, process name, and process id are all prepended to the log message by syslog. *[main]* is the thread that printed the log message. *NFS STARTUP* is the component. *EVENT* is the severity of the message. Finally *Configuration file successfully parsed* is the message itself. Each log message in GANESHA follows this format.

1.1 Logging Components

Listed below are all of the components that a log message can belong to

- COMPONENT_ALL -
- COMPONENT_LOG - Reports changes to log destination or log level.
- COMPONENT_MEMALLOC - Used for tracking memory allocation through the buddy malloc system.

- COMPONENT_STATES - Mainly used with severity level NIV_FULL_DEBUG to debug NFSV4 states.
- COMPONENT_MEMLEAKS - Mainly used with severity level NIV_DEBUG and NIV_FULL_DEBUG. It is used to detect and debug memory leaks.
- COMPONENT_FSAL - Messages that relate to FSAL specific functions.
- COMPONENT_NFSPROTO - Messages that relate to the NFSV2 and NFSV3 protocol.
- COMPONENT_NFSV4 - Messages relating to the NFSV4 protocol.
- COMPONENT_NFSV4_PSEUDO - Message relating to the NFSV4 pseudo file system.
- COMPONENT_FILEHANDLE - Messages relating to the conversion from a file handle (used by clients) to an FSAL specific handle (used by the server).
- COMPONENT_NFS_SHELL - Messages relating to the GANESHA shell.
- COMPONENT_DISPATCH - Messages relating to the dispatch thread which is responsible for sending replies to clients.
- COMPONENT_CACHE_CONTENT - Messages related to cached file data that makes client requests faster.
- COMPONENT_CACHE_INODE -
- COMPONENT_CACHE_INODE_GC -
- COMPONENT_HASHTABLE -
- COMPONENT_LRU -
- COMPONENT_DUPREQ - GANESHA (and other NFS servers) keep a cache of recent requests and their replies. If a duplicate request is received GANESHA will send the same reply rather than process the request again. The messages relating to this duplicate request cache fall under this component.
- COMPONENT_RPCSEC_GSS -
- COMPONENT_INIT - Most messages logged when GANESHA is starting will be logged under this component.
- COMPONENT_MAIN -
- COMPONENT_IDMAPPER -
- COMPONENT_NFS_READDIR -

- COMPONENT_NFSV4_LOCK - Debugs file locking for NFSV4.
- COMPONENT_NFSV4_XATTR - Debugs extended attributes for NFSV4.
- COMPONENT_NFSV4_REFERRAL - Debugs NFSV4's ability to refer clients to another NFSV4 server.
- COMPONENT_MEMCORRUPT -
- COMPONENT_CONFIG -
- COMPONENT_CLIENT_ID_COMPUTE -
- COMPONENT_STDOUT -
- COMPONENT_OPEN_OWNER_HASH -
- COMPONENT_SESSIONS -
- COMPONENT_PNFS - Message related to the ability for NFS to work in parallel to other NFS servers. Parallel NFS is a compile time option and won't be seen unless it was compiled.
- COMPONENT_RPC_CACHE -

1.2 Log Severity Levels

The log levels (debug levels, severity levels) and their order of severity are as follows:

- NIV_NULL - no messages are printed.
- NIV_MAJ - only major messages are printed.
- NIV_CRIT - critical messages or higher are printed.
- NIV_EVENT - event messages or higher printed.
- NIV_DEBUG - debug messages or higher are printed. This should only be used if a user suspects they have found a bug or are developing GANESHA.
- NIV_FULL_DEBUG - extremely verbose debug messages or higher are printed.

The default log level is NIV_EVENT.

1.3 Log Location

The location where log messages will be printed is configurable when GANESHA first starts. Once GANESHA is running, the log destination cannot be changed.

Section 2 covers the different mechanisms for configuring the log destination upon startup. Each method accepts a string as the log destination. The valid strings are as follows:

- "SYSLOG" - Log messages will go to syslog. Where the log messages go after being sent to the syslog daemon depends on how syslog was configured, but the messages will probably go to "/var/log/messages". Syslog is the default log location.
- "STDOUT" - Log messages will be sent to the console that started GANESHA as stdout output. This can sometimes be useful for debugging purposes but is not generally used.
- "STDERR" - Log messages will be sent to the console that started GANESHA as stderr output. This can sometimes be useful for debugging purposes but is not generally used.
- "/some/path/to/file" - If none of the above strings are given, GANESHA will assume the string refers to a file path. Log messages will be appended to the filepath. All of the directories in this path must already exist. Otherwise an error will be thrown and GANESHA will quit.

1.4 Changing the Log Destination

There are two ways to change where log messages will go, through the commandline or through configuration file variables. It is recommended people use the default of syslog, but there are may be some scenarios where separating the logging of certain components will be useful.

1.5 Commandline Arguments

```
/usr/bin/gpfs.ganesha.nfsd -d -f /etc/ganesha/gpfs.ganesha.nfsd.conf -N \
NIV_EVENT -L SYSLOG
```

1.6 Configuration File Variables

Table 1 lists the configuration file stanzas that contain a variable for the location of its log files. Setting the variable for that stanza, displayed in the second column of the table, will set the log destination for a specific logging component, displayed in the third column.

Each stanza uses the variable name of "LogFile". So an example of setting the FSAL stanza to using stdout output is:

Configuration file stanza	configuration variable	logging component
FSAL	LogFile	COMPONENT_FSAL
CacheInode_Client	LogFile	COMPONENT_CACHE_INODE
FileContent_Client	LogFile	COMPONENT_CACHE_CONTENT
BUDDY_MALLOC	LogFile	COMPONENT_MEMALLOC COMPONENT_MEMLEAKS

Figure 1: Configuration File Variables for Changing Log Destination

```
FSAL
{
    LogFile="STDOUT";
}
```

2 Methods for Controlling the Log Level of Components

There are a total of six different ways that the log level of components is set. The order of priority for the six is shown in Figure 2. First there is a default log level for all components. Second, environment variables can be defined

compiled defaults → environment variables → command-line arguments →
configuration file parameters → signal handlers ↔ snmp

Figure 2: Order of Priority of Mechanisms for Changing Log Levels

2.1 Default Log Levels

The default log level of each component is NIV_EVENT. If no other mechanism is used, the log level will remain at NIV_EVENT.

2.2 Environment Variables

The name of each component, listed in Subsection 1.1, can be used as the name of an environment variable to set the debug level of that component.

This option is not recommended for use in a GANESHA deployment, but meant for fine granularity tuning of debug messages for development purposes.

2.3 Commandline Arguments

```
/usr/bin/gpfs.ganesha.nfsd -d -f /etc/ganesha/gpfs.ganesha.nfsd.conf -N \
NIV_EVENT -L SYSLOG
```

2.4 Configuration File Variables

Configuration file stanza	configuration variable	logging component
FSAL	DebugLevel	COMPONENT_FSAL
CacheInode_Client	DebugLevel	COMPONENT_CACHE_INODE
FileContent_Client	DebugLevel	COMPONENT_CACHE_CONTENT

Figure 3: Configuration File Variables for Changing Log Levels

Each stanza uses the variable name of "DebugLevel". An example of setting the FSAL stanza to using NIV_DEBUG is:

```
FSAL
{
    DebugLevel="NIV_DEBUG";
}
```

2.5 Signal Handlers

To incrememnt log level of all components:

```
$ killall -s SIGUSR1 /usr/bin/gpfs.ganesha.nfsd
$ tail -1 /var/log/messages
Sep  8 02:28:47 localhost nfs-ganesha[11287]: [main] :LOG: SIGUSR1 Increasing \
log level for all components to NIV_DEBUG
```

To decrement log level of all components:

```
$ killall -s SIGUSR2 /usr/bin/gpfs.ganesha.nfsd
$ tail -1 /var/log/messages
Sep  8 02:29:00 localhost nfs-ganesha[11287]: [main] :LOG: SIGUSR2 Decreasing \
log level for all components to NIV_EVENT
```

2.6 SNMP

It is necessary to setup an SNMP server in order to use the SNMP support in GANESHA. For instructions on how to install and configure this server as well as how to access GANESHA through SNMP, refer to the GANESHA SNMP ADMINISTRATION guide. Once SNMP is properly installed, the tool presented in Section 3 can be used for easy local and remote administration.

3 GANESHA Tool for Reading/Modifying Log Levels with SNMP

GANESHA contains a program written in Perl that provides easy access to log level information. From the root directory of the GANESHA code base, the

path to the program is *src/cmdline_tools/ganesha_log_level* and, if installed through the rpm, the program is installed to */usr/bin/ganesha_log_level*.

ganesha_log_level requires a configuration file to locate and access the SNMP server. A sample configuration file is installed to */etc/ganesha/snmp.conf*. The file is very simple:

```
# This is a sample snmp.conf to be used with ganesha to allow
# the ganesha_log_level tool to work. Put it in /etc/ganesha

community_string = public
```

It is essential that the community_string parameter matches the community column in */etc/snmp/snmpd.conf* that looks like the following:

```
##       sec.name  source          community
com2sec  local     localhost       ganesha
com2sec  mynetwork 9.47.69.0/24    ganesha
```

3.1 Examining Log Levels with *ganesha_log_level*

List possible log levels:

```
$ ganesha_log_level -L
Valid Log Levels (less to more messages):
  NIV_NULL
  NIV_MAJOR
  NIV_CRIT
  NIV_EVENT
  NIV_DEBUG
  NIV_FULL_DEBUG
```

List current log levels on components:

```
$ ganesha_log_level -l
Current Log Levels:
  COMPONENT_ALL                   NIV_EVENT
  COMPONENT_CACHE_CONTENT         NIV_EVENT
  COMPONENT_CACHE_INODE           NIV_EVENT
  COMPONENT_CACHE_INODE_GC        NIV_EVENT
  COMPONENT_CLIENT_ID_COMPUTE     NIV_EVENT
  COMPONENT_CONFIG                NIV_EVENT
  COMPONENT_DISPATCH              NIV_EVENT
<output continues ...>
```

List possible components:

```
$ ganesha_log_level -C
Valid Components:
  COMPONENT_ALL
```

```
  COMPONENT_CACHE_CONTENT
  COMPONENT_CACHE_INODE
  COMPONENT_CACHE_INODE_GC
  COMPONENT_CLIENT_ID_COMPUTE
  COMPONENT_CONFIG
  COMPONENT_DISPATCH
<output continues ...>
```

Get the log level for one or more components

```
$ ganesha_log_level -g COMPONENT_LRU
NIV_EVENT
```

3.2 Changing Log Levels with *ganesha_log_level*

Set the log level for one or more components:

```
$ ganesha_log_level -s NIV_DEBUG COMPONENT_LRU
$ ganesha_log_level -g COMPONENT_LRU
NIV_DEBUG
```