Ganglia Toolkit

Ganglia Development Team

The latest version of this document can be found on the ganglia documentation page (http://ganglia.sourceforge.net/docs/).

Ganglia grew out of UC Berkeley Computer Science clustering research: The Millennium Cluster Project (http://www.millennium.berkeley.edu/) in collaboration with the NPACI Rocks (http://rocks.npaci.edu/) Cluster Group.

Ganglia (http://ganglia.sourceforge.net/) provides a complete real-time monitoring and execution environment that is in use by hundreds of universities, private and government laboratories and commercial cluster implementors around the world. Whether you want to monitor hundreds of computers in real-time across a university campus or around the world, ganglia is for you.

Component	Description	Author	Version
Ganglia Monitor Daemon (gmond)	This daemon must be running on all nodes you want to monitor. it multicasts its present for other gmonds, collects host metrics, multicasts the state of the host on the multicast channel and exports the data for itself and other nodes via XML	Matt Massie, Preston Smith (FreeBSD and AIX port), Steve Wagner (Solaris and Tru64 port), Federico Sacerdoti, Matt Rice (Windows port)	2.5.0

Component	Description	Author	Version
Ganglia Meta Daemon (gmetad)	The Ganglia Meta Daemon merges and data from multiple gmond sources over the wide area, stores the volatile numerical data in local round-robin databases, concatenates and presents a single XML image of all clusters which are being monitored.	Matt Massie	2.5.0
Ganglia Metad Web Interface	The Ganglia Meta Daemon Web Interface uses a PHP-enabled (http://www.php.net/) web server to present the data collected by gmetad to a web browser. For an example of the output, visit `http://ganglia.sourceforge.net/demo/`	Federico Sacerdoti and Matt Massie	2.5.0
Python Class and Client	a Python class for sorting and classifying large clusters using the monitoring core	Mason Katz and Greg Bruno	2.0

Table of Contents

1. Introduction

Jeremy S. Anderson

There are two major products that come out of Berkeley: LSD and UNIX. We don't believe this to be a coincidence.

Ganglia (http://ganglia.sourceforge.net/) provides a complete real-time monitoring and execution environment that is in use by hundreds of universities, private and government laboratories and commercial cluster implementors around the world. Ganglia is as simple to install and use on a 16-node cluster as it is to use on a 512-node cluster as has been proven by its use on multiple 500+ node clusters.

Ganglia was initially developed at the University of California, Berkeley Computer Science Division as way to link clusters across the Berkeley campus together in a logical way. Since it was developed at a university, it is completely open-source and has no proprietary components. All data is exchanged in well-defined XML and XDR to ensure maximum extensibility and portability.

The monitoring core allows you to monitor any number of host metrics in real-time. At present, the monitoring core runs on Linux, FreeBSD, Solaris, AIX, Tru64 and IRIX. There is a Windows port as well which is in beta.

Ganglia is not just a way to link nodes in a cluster together in a logical way but also a way to link clusters to other clusters. Ganglia blurs the line between clustering and distributed computing by providing for Cluster to Cluster (C2C) data exchanges which link disparate cluster resources together into a single logical framework.

1.1. Ganglia Monitoring Daemon (gmond)

Last Updated: $Date: 2002/09/16 20:12:49 $

Gmond is a multi-threaded daemon which runs on each cluster node you want to monitor. Installation is easy. You don't have to have a common NFS filesystem or a database, install special accounts, maintain configuration files or other annoying hassles. Gmond is its own redundant, distributed database.

Gmond has four main responsibilities: *monitor* changes in host state, *multicast* relevant changes, *listen* to the state of all other ganglia nodes via a multicast channel and *answer* requests for an XML description of the cluster state.

Gmond has threads which listen to the multicast channel and write the data collected to a fast, in-memory hash table. All metric data for each cluster node is processed and saved. You might think this would require large amounts of memory but a gmond can maintain the in-memory data in just 16+ 136*n_nodes +364*n_metrics bytes. For example, if you have a massive *1024* node cluster and you are monitoring *25 metrics* on each machine then gmond will only use `16 + 136*1024 + 364*25 = 148380 bytes` or just 144 kilobytes of memory! The hash table is also completely multi-threaded with read/write locks implemented by POSIX thread mutexes and condition variables for each individual host in the cluster. That means that multiple threads can simultaneously read and write to the hash without interfering with each other. Very scalable. Ganglia is used on many 500+ node clusters.

Each gmond transmits in two different ways: multicasting host state in external data representation (XDR) or sending XML over a TCP connection.

Gmond only multicasts a metric that it is monitoring for two reasons: a change in the value of the metric exceeds the value threshold or when gmond hasn't multicast the metric longer than the time threshold. The value threshold ensures that gmond only multicast when it really needs to. If there is no change in state, then no data is multicast. Of course things are always changing and the time threshold ensures that small changes haven't grow into large changes too slowly to notice. These thresholds dramatically reduces chatter on the multicast channel. For example, the number of CPUs on a host is only sent once an hour since it is a constant whereas 1-minute load could be sent every 15 seconds depending on the change in value.

Since all gmonds in your cluster are storing your cluster state, it is important that all nodes have the same cluster image. The gmond self-organizes and knows how to make sure that all gmonds are on the same page.

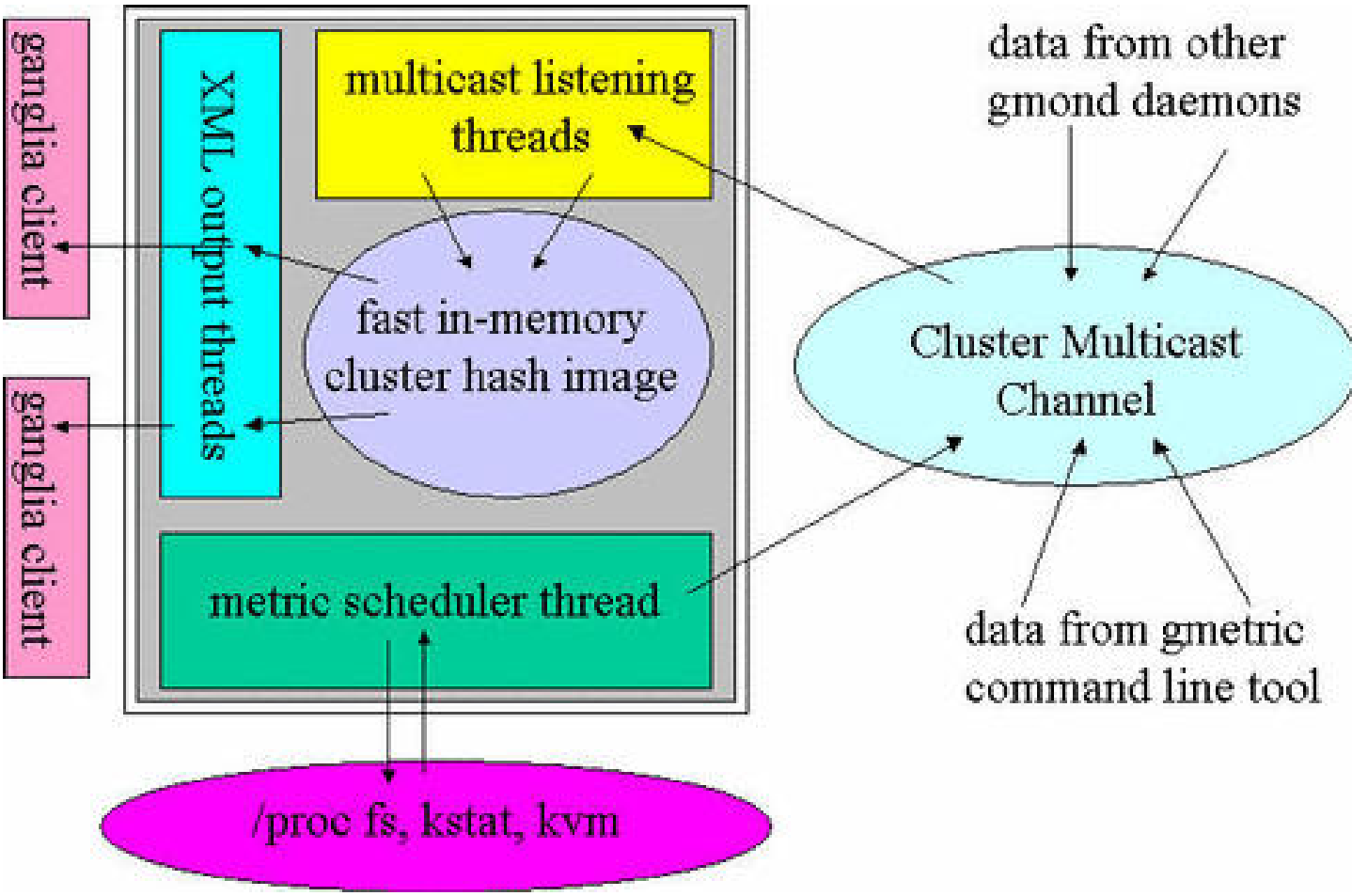
Anytime that gmond gets a multicast packet from a new host, it expires the time threshold for all its metrics. This means that all its metric data will get sent to the new gmond even if it wasn't previously scheduled to be sent. For example, the number of CPUs is only multicast once an hour. If the local gmond did not expire its time threshold then the new gmond wouldn't know the number of CPUs on that machine for up to an hour.

Gmond is also good at handling failures. It's inevitable that nodes in your cluster will go down occasionally and (heaven-forbit!) gmond may even fail. Recovering gracefully from those temporary failures is important. The payload for a heartbeat message is the timestamp of when gmond was started on a node. If ever that timestamp changes then gmond is alerted of a gmond restart on another node. This alert will trigger gmond to expire the time threshold on all its metrics in order to quickly update the new gmond of it's state.

Each gmond processes its own multicast data locally via loopback. That means that it saves in-memory the very data that it's sending to remote gmonds.

When asked, gmond will write out the complete cluster state in XML including its DTD. However, gmond will only share that data with hosts in its in-memory cluster hash OR a host specified with the trusted_hosts list in its configuration file (/etc/gmond.conf by default).

You can send custom metrics on the ganglia multicast channel as well. Gmond monitors dozens of metrics right out of the box but gmond doesn't assume that these are the only only metrics that you want to monitor. To expand the list of metrics you are monitoring, use the gmetric tool.



The grey box represents the Ganglia Monitoring Daemon (gmond) with all its components inside: the *metric scheduler thread*, *multicast listening threads*, *fast in-memory hash* and the *XML output threads*.

The *metric scheduler thread* checks the state of the host that gmond is running on and multicasts any relevant changes. Gmond decides what is relevant by using *value and time thresholds*. The metric scheduler remembers what value it last multicast and when it multicast it. If the difference between the last multicast value and the new value is greater than the value threshold, then the metric scheduler will multicast the value. Also, if the time elapsed since the metric value has been multicast is greater than the time threshold it will be sent regardless of its value. The *value and time thresholds* are metric-specific and set based on the metric characteristics.

The *multicast listening threads* listen on the ganglia multicast channel for incoming messages including messages from itself (via loopback). All data is stored in the *fast in-memory hash*. This hash holds the data for all hosts sending data on the cluster multicast channel via gmond or gmetric.

The *XML output threads* are responsible for processing incoming connection requests. When I request for XML is made the output threads checks if the client is 127.0.0.1, searches the hash of known hosts on the ganglia multicast channel and then the *trusted_hosts* list (in /etc/gmond.conf). If the host is not found, it closes the connection; otherwise, a complete XML description of the state of all hosts multicasting on its local multicast channel. You use telnet if you want to see this description...

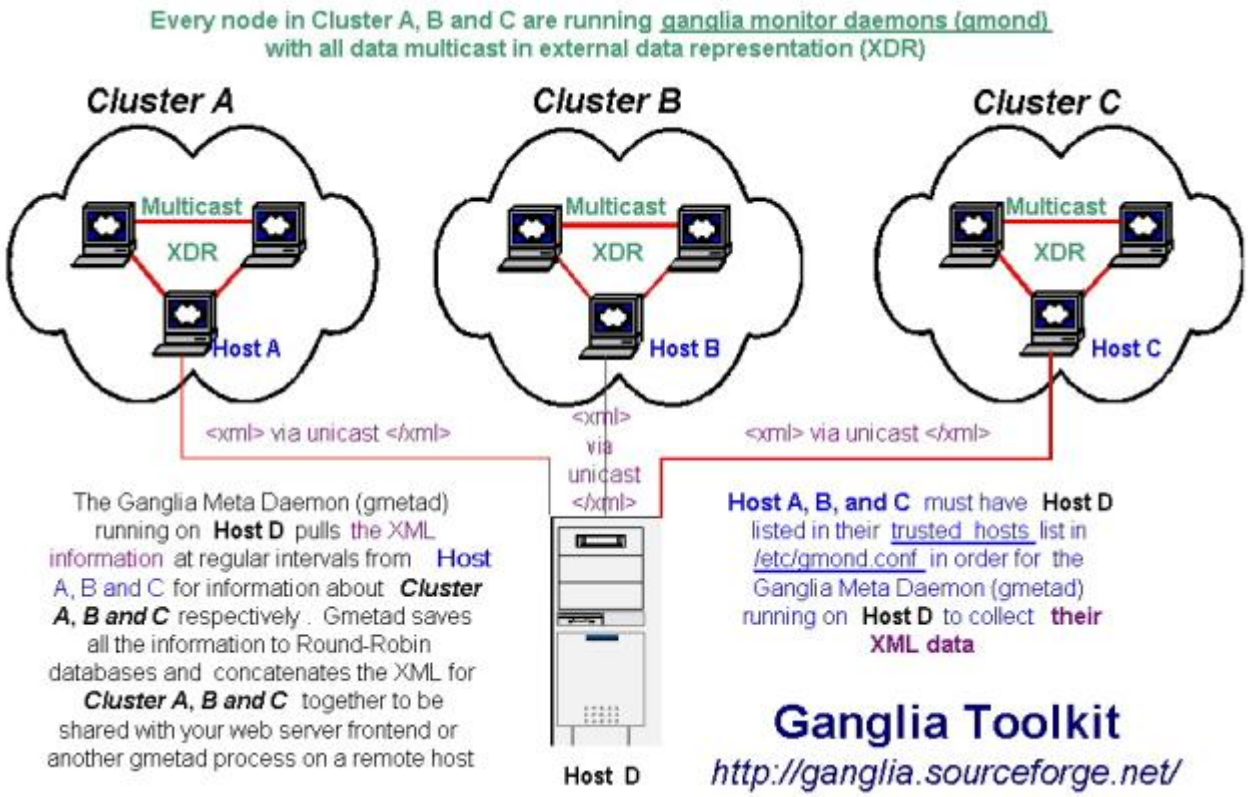
```
telnet localhost 8649
```

Port 8649 (U*N*I*X on a phone keypad :)) is the default port for the XML threads to listen on but it can be changed at run time.

1.2. Ganglia Meta Daemon (gmetad)

Last Updated: $Date: 2002/10/16 17:55:13 $

This graphical representation show how the Ganglia Meta Daemon (gmetad) works with Ganglia Monitoring Daemons (gmond) to allow you to easily monitor cluster over unicast routes. While gmond uses multicast channels in a peer-to-peer way, gmetad pulls the XML description from ganglia data sources (either gmond or another gmetad) via XML over unicast routes.



Configuring Gmetad is simple. Gmetad behavior is controlled by a single configuration file (/etc/gmetad.conf) by default. This configuration file contains instructions on its use.

Gmetad is the backend for the ganglia web frontend. Gmetad stores historical information to Round-Robin databases and exports summary XML which the web frontend uses to present useful snapshots and trends for all hosts monitored by ganglia.

1.3. Ganglia Web Frontend

Last Updated: $Date: 2002/09/10 20:01:17 $

The Ganglia (http://ganglia.sourceforge.net/) web frontend provides a view of the gathered information via real-time dynamic web pages. Most importantly, it displays Ganglia data in a meaningful way for system administrators and computer users. Although the web frontend to ganglia started as a simple HTML view of the XML tree, it has evolved into a system that keeps a colorful history of all collected data.

The Ganglia web frontend caters to system administrators and users. For example, one can view the CPU utilization

over the past hour, day, week, month, or year. The web frontend shows similar graphs for Memory usage, disk usage, network statistics, number of running processes, and all other Ganglia metrics.

The web frontend depends on the existence of the Gmetad which provides it with data from several Ganglia sources. Specifically, the web frontend will open the local port `8651` (by default) and expects to receive a Ganglia XML tree. The web pages themselves are highly dynamic; any change to the Ganglia data appears immediately on the site. This behavior leads to a very responsive site, but requires that the full XML tree be parsed on every page access. Therefore, the Ganglia web frontend should run on a fairly powerful, dedicated machine if it presents a large amount of data. Currently, a Ganglia Grid containing ~15 clusters with over 500 hosts and 900 CPUs is running succesfully on a dual 600Mhz Pentium III Linux server, with all RRD graph files on a ramdisk. See this page at meta.rocksclusters.org (http://meta.rocksclusters.org/).

The Ganglia web frontend is written in the PHP (http://www.php.net/) scripting language, and uses graphs generated by Gmetad to display history information. It has been tested on many flavours of Unix (primarily Linux) with the Apache webserver and the PHP 4.1 module.

2. Installation

> Albert Einstein
>
> Make things as simple as possible, but not simpler.

2.1. Ganglia Monitoring Core Installation (gmetad, gmond, gstat, gmetric)

Last Updated: $Date: 2002/10/16 18:04:26 $

2.1.1. Installation from Source

Starting in version 2.5.1, the gmond and gmetad services reside in separate packages. We made this change to simplify service management: when you install the gmetad RPM package, the gmetad daemon is automatically started, the same for gmond.

1. Your machine and network must be multicast-enabled to run ganglia. If your machines are all on the same switch then you are in luck: gmond is ready for use without any configuration tweaks. However, if the machines in your cluster are seperated by a router, then you will need to set the *mcast_ttl* option in /etc/gmond.conf to be higher than the default of 1. Set the multi-cast Time-To-Live (TTL) to be one greater than the number of hops

(routers) between the hosts. Also, you will also have to make sure that the routers are configured to pass along the multicast traffic.

> **Note:** Some users have reported problems on machines with no "default" route. Gmond reports the error *mcast_join() setsockopt() error: No such device* and then exits at startup. The workaround is to create a static route for the ganglia multicast channel.
>
> ```
> prompt> route add -host 239.2.11.71 dev eth0
> ```
>
> Replace *eth0* above with the name of the network interface you want to send ganglia multicast traffic on

2. Download the latest source tarball from http://ganglia.sourceforge.net/downloads.php
3. Unzip the distribution

```
prompt> gunzip < ganglia-monitor-core-2.5.0.tar.gz | tar -xvf -
ganglia-monitor-core-2.5.0/
ganglia-monitor-core-2.5.0/Makefile.in
ganglia-monitor-core-2.5.0/README
ganglia-monitor-core-2.5.0/stamp-h.in
ganglia-monitor-core-2.5.0/AUTHORS
ganglia-monitor-core-2.5.0/COPYING
ganglia-monitor-core-2.5.0/ChangeLog
ganglia-monitor-core-2.5.0/INSTALL
ganglia-monitor-core-2.5.0/Makefile.am
ganglia-monitor-core-2.5.0/NEWS
ganglia-monitor-core-2.5.0/acconfig.h
ganglia-monitor-core-2.5.0/acinclude.m4
ganglia-monitor-core-2.5.0/aclocal.m4
ganglia-monitor-core-2.5.0/config.h.in
...
```

4. Jump into the distribution directory

```
prompt> cd ganglia-monitor-core-2.5.0
```

5. Run the configuration script

> **Note:** You must add the *--with-gmetad* configure flag if you wish to compile and install the Ganglia Meta Daemon (gmetad). To compile gmetad, you must have the Round-Robin Database library and header files (http://www.rrdtool.com/download.html) installed (librrd.a and rrd.h).

Note: You must add the *--disable-shared* and *--enable-static* configure flags if you running on AIX

```
prompt> ./configure
creating cache ./config.cache
checking for a BSD compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking whether make sets ${MAKE}... yes
checking for working aclocal... found
checking for working autoconf... found
checking for working automake... found
checking for working autoheader... found
checking for working makeinfo... found
checking host system type... i686-pc-linux-gnu
checking for gcc... gcc
...
creating ./config.status
creating Makefile
creating lib/Makefile
creating gmond/Makefile
creating gmetric/Makefile
creating ganglia.spec
creating config.h
linking ./gmond/machines/linux.c to gmond/machine.c
```

6. Run make

```
prompt> make
make  all-recursive
make[1]: Entering directory ‘/tmp/mas/ganglia-monitor-core-2.5.0’
Making all in lib
make[2]: Entering directory ‘/tmp/mas/ganglia-monitor-core-2.5.0/lib’
/bin/sh ../libtool --mode=compile gcc -DHAVE_CONFIG_H -I. -I. -I..    -I. -O2  -D_REENTRANT -

Wall -Wshadow -Wpointer-arith -Wcast-align -Wstrict-prototypes -D_GNU_SOURCE -c daemon_inetd.
c
mkdir .libs
gcc -DHAVE_CONFIG_H -I. -I. -I.. -I. -O2 -D_REENTRANT -Wall -Wshadow -Wpointer-arith -
Wcast-a
lign -Wstrict-prototypes -D_GNU_SOURCE -c daemon_inetd.c  -fPIC -DPIC -o .libs/daemon_inetd.l
o
```

```
gcc -DHAVE_CONFIG_H -I. -I. -I.. -I. -O2 -D_REENTRANT -Wall -Wshadow -Wpointer-arith -
Wcast-a
lign -Wstrict-prototypes -D_GNU_SOURCE -c daemon_inetd.c -o daemon_inetd.o >/dev/null 2>&1
mv -f .libs/daemon_inetd.lo daemon_inetd.lo
...
```

7. Once everything is built, install it

```
prompt> make install
Making install in lib
make[1]: Entering directory ‘/tmp/mas/ganglia-monitor-core-2.5.0/lib’
make[2]: Entering directory ‘/tmp/mas/ganglia-monitor-core-2.5.0/lib’
/bin/sh ../config/mkinstalldirs /usr/lib
/bin/sh ../libtool  --mode=install /usr/bin/install -c libganglia.la /usr/lib/libganglia.la
...
```

8. Start up gmond and make sure it is started at reboot. Activate gmetad too if necessary.

> **Note:** These instructions are Linux specific. Each operating system has a unique way of making sure that daemons are started at system boot.

```
prompt> cp ./gmond/gmond.init /etc/rc.d/init.d/gmond
prompt> chkconfig --add gmond
prompt> chkconfig --list gmond
gmond           0:off 1:off 2:on 3:on 4:on 5:on 6:off
prompt> /etc/rc.d/init.d/gmond start
Starting GANGLIA gmond:                                    [  OK  ]
```

If you are installing the monitoring core and a web server you plan to install the web frontend on, then you will need to activate gmetad as well in much the same way. If you have specified a different location to store your RRDs than the default (/var/lib/ganglia/rrds) then substitute the new directory in the following example...

```
prompt> mkdir -p /var/lib/ganglia/rrds
prompt> chown -R nobody /var/lib/ganglia/rrds
prompt> cp ./gmetad/gmetad.init /etc/rc.d/init.d/gmetad
prompt> chkconfig --add gmetad
prompt> chkconfig --list gmetad
gmond             0:off 1:off 2:on  3:on  4:on  5:on  6:off
prompt> /etc/rc.d/init.d/gmetad start
Starting GANGLIA gmetad:                                    [  OK  ]
```

9. Test your gmond installation

   ```
   prompt> telnet localhost 8649
   ```

   You should see XML that conforms to the ganglia XML spec

   Optionally, you can test you gmetad installation

   ```
   prompt> telnet localhost 8651
   ```

2.1.2. Installation Using RPM (Linux-specific)

1. Your machine and network must be multicast-enabled to run ganglia (stock kernel almost always have multicast). If your machines are all on the same switch then you are in luck: gmond is ready for use without any configuration tweaks. However, if the machines in your cluster are seperated by a router, then you will need to set the *mcast_ttl* option in /etc/gmond.conf to be higher than the default of 1. Set the multi-cast Time-To-Live (TTL) to be one greater than the number of hops (routers) between the hosts. Also, you will also have to make sure that the routers are configured to pass along the multicast traffic.

   > **Note:** Some users have reported problems on machines with no "default" route. Gmond reports the error *mcast_join() setsockopt() error: No such device* and then exits at startup. The workaround is to create a static route for the ganglia multicast channel.
   >
   > ```
   > prompt> route add -host 239.2.11.71 dev eth0
   > ```
   >
   > Replace *eth0* above with the name of the network interface you want to send ganglia multicast traffic on

2. Download the latest monitoring core RPMs: gmond (http://prdownloads.sourceforge.net/ganglia/ganglia-monitor-core-gmond-2.5.0-1.i386.rpm) | gmetad (http://prdownloads.sourceforge.net/ganglia/ganglia-monitor-core-gmetad-2.5.0-1.i386.rpm).
3. Install the Gmond RPM on every machine that you want to monitor with ganglia

   ```
   prompt> rpm -Uvh ganglia-monitor-core-gmond-2.5.0-1.i386.rpm
   Preparing...                ########################################### [100%]
      1:ganglia-monitor-core   ########################################### [100%]
   Starting GANGLIA gmond: [  OK  ]
   ```

   This package will install all the necessary startup files and activate gmond for startup at system boot and start the Ganglia Monitor Daemon (gmond).

4. Install the Gmetad RPM on a frontend or monitoring machine. This machine can be a compute node, but generally the gmetad service will run on a dedicated webserver.

   ```
   prompt> rpm -Uvh ganglia-monitor-core-gmetad-2.5.0-1.i386.rpm
   Preparing...                ########################################### [100%]
      1:ganglia-monitor-core   ########################################### [100%]
   Starting GANGLIA gmond: [  OK  ]
   ```

   This package will install all the necessary startup files and activate gmetad for startup at system boot and start the Ganglia Meta Daemon (gmetad). You will need to define the "data sources" in the `/etc/gmetad.conf` configuration file, to tell Gmetad which Gmond daemons to monitor.

   ```
   ## /etc/gmetad.conf ##
   #
   # The data_source tag must immediately be followed by a unique
   # string which identifies the source then a list of machines
   # which service the data source in the format ip:port, or name:port.
   # If a # port is not specified then 8649 (the default gmond port) is
   # assumed.
   # default: There is no default value

   data_source "my cluster" localhost
   ```

2.2. Ganglia Web Frontend

Last Updated: $Date: 2002/09/17 17:50:55 $

2.2.1. Installation from Source

1. Download the latest source tarball from http://ganglia.sourceforge.net/downloads.php
2. Unzip the webfrontend distribution in your website tree. This is often under the directory `/var/www/html`, however look for the variable `DocumentRoot` in your Apache configuration files to be sure. All the PHP script files use relative URLs in their links, so you may place the `gmetad-webfrontend/` directory anywhere convenient. I like to unzip *tar.gz files with one *tar* command:

   ```
   prompt> cd /var/www/html
   prompt> tar xvzf gmetad-webfrontend-2.5.0.tar.gz
   ```

3. Ensure your webserver understands how to process PHP (http://www.php.net/) script files. Currently, the web frontend contains certain php language that requires PHP version 4 or greater. Processing PHP script files usually requires a webserver module, such as the *mod_php* for the popular Apache webserver. In RedHat Linux, the RPM package that provides this module is called simply "php".

   For Apache, mod_php module must be enabled. The following lines should appear somewhere in Apache's *conf files. This example applies to RedHat and Mandrake Linux. The actual filenames may vary on your system. If you installed the php module using an RPM package, this work will have been done automatically.

   ```
   <IfDefine HAVE_PHP4>
   LoadModule php4_module    extramodules/libphp4.so
   AddModule mod_php4.c
   </IfDefine>

   AddType  application/x-httpd-php         .php .php4 .php3 .phtml
   AddType  application/x-httpd-php-source  .phps
   ```

4. The webfrontend requires the existance of the gmetad package on the webserver. Follow the installation instructions on the gmetad page. Specifically, the webfrontend requires the *rrdtool* and the `rrds/` directory from gmetad. If you are a power user, you may use NFS to simulate the local existance of the rrds.

5. Test your installation. Visit the URL:

   `http://localhost/gmetad-webfrontend/`

   With a web-browser, where *localhost* is the address of your webserver.

2.2.2. Installation Using RPM

1. Installation from the RPM should be easy, however they currently only work on Linux i386 machines. Download gmetad-webfrontend-2.5.0-1.i386.rpm (http://prdownloads.sourceforge.net/ganglia/gmetad-webfrontend-2.5.0-1.i386.rpm), the latest web frontend RPM package.

2. Install the gmetad-webfrontend RPM on your webserver machine.

   ```
   prompt> rpm -Uvh gmetad-webfrontend-2.5.0-1.i386.rpm
   Preparing...                ########################################### [100%]
      1:gmetad-webfrontend     ########################################### [100%]
   ```

3. Test your installation. Visit the URL:

```
http://localhost/gmetad-webfrontend/
```

With a web-browser, where *localhost* is the address of your webserver.

3. Configuration

Ethiopian proverb

When spiders unite, they can tie down a lion.

3.1. Gmetad

The behavior of the Ganglia Meta Daemon is completely controlled by a single configuration file which is by default /etc/gmetad.conf. For gmetad to do anything useful you much specify at least one data_source in the configuration. The format of the data_source line(s) is as follows

```
data_source "Cluster A" 127.0.0.1  1.2.3.4:8655  1.2.3.5:8625
data_source "Cluster B" 1.2.4.4:8655
```

In this example, there are two unique data sources: "Cluster A" and "Cluster B". The Cluster A data source has three redundant sources. If gmetad cannot pull the data from the first source, it will continue trying the other sources in order.

> **Note:** If you do not specify a port for a source, gmetad will assume you want to connect to the default gmond port of 8649.

Here is a sample gmetad configuration file:

```
# This is an example of a Ganglia Meta Daemon configuration file
#               http://ganglia.sourceforge.net/
#
# $Id: gmetad.conf,v 1.3 2002/09/19 18:56:42 sacerdoti Exp $
#
# Setting the debug_level above zero will make gmetad output
# debugging information and stay in the foreground
# default: 0
```

```
# debug_level 10
#
# The data_source tag must immediately be followed by a unique
# string which identifies the source then a list of machines
# which service the data source in the format ip:port, or name:port.
# If a # port is not specified then 8649 (the default gmond port) is
# assumed.
# default: There is no default value
# data_source "my box" localhost  my.machine.edu:8655  1.2.3.5:8655
# data_source "another source"  1.3.4.7:8655  1.3.4.8
#
# List of machines this gmetad will share XML with
# default: There is no default value
# trusted_hosts 127.0.0.1 169.229.50.165
#
# If you don't want gmetad to setuid then set this to off
# default: on
# setuid off
#
# User gmetad will setuid to (defaults to "nobody")
# default: "nobody"
# setuid_username "nobody"
#
# The port gmetad will answer requests for XML
# default: 8651
# xml_port 8651
#
# The number of threads answering XML requests
# default: 2
# server_threads 4
#
# Where gmetad stores its round-robin databases
# default: "/var/lib/ganglia/rrds"
# rrd_rootdir "/some/other/place"
```

3.2. Gmond

While the default options for gmond will work for most clusters, gmond is very flexible and can be customize with the configuration file: /etc/gmond.conf.

/etc/gmond.conf is not required as its absence will only cause gmond to start in a default configuration. Here is a sample of a gmond.conf configuration file...

```
# $Id: gmond.conf,v 1.2 2002/09/19 00:37:18 sacerdoti Exp $
# This is the configuration file for the Ganglia Monitor Daemon (gmond)
# Documentation can be found at http://ganglia.sourceforge.net/docs/
#
# To change a value from it's default simply uncomment the line
# and alter the value
######################
#
# The name of the cluster this node is a part of
# default: "unspecified"
# name  "My Cluster"
#
# The owner of this cluster. Represents an administrative
# domain. The pair name/owner should be unique for all clusters
# in the world.
# default: "unspecified"
# owner "My Organization"
#
# The latitude and longitude GPS coordinates of this cluster on earth.
# Specified to 1 mile accuracy with two decimal places per axis in Decimal
# DMS format: "N61.18 W130.50".
# default: "unspecified"
# latlong "N32.87 W117.22"
#
# The URL for more information on the Cluster. Intended to give purpose,
# owner, administration, and account details for this cluster.
# default: "unspecified"
# url "http://www.mycluster.edu/"
#
# The location of this host in the cluster. Given as a 3D coordinate:
# "Rack,Rank,Plane" that corresponds to a Euclidean coordinate "x,y,z".
# default: "unspecified"
# location "0,0,0"
#
# The multicast channel for gmond to send/receive data on
# default: 239.2.11.71
# mcast_channel 239.2.11.71
#
# The multicast port for gmond to send/receive data on
# default: 8649
# mcast_port    8649
#
# The multicast interface for gmond to send/receive data on
# default: the kernel decides based on routing configuration
# mcast_if  eth1
```

```
#
# The multicast Time-To-Live (TTL) for outgoing messages
# default: 1
# mcast_ttl  1
#
# The number of threads listening to multicast traffic
# default: 2
# mcast_threads 2
#
# Which port should gmond listen for XML requests on
# default: 8649
# xml_port     8649
#
# The number of threads answering XML requests
# default: 2
# xml_threads   2
#
# Hosts ASIDE from "127.0.0.1"/localhost and those multicasting
# on the same multicast channel which you will share your XML
# data with.  Multiple hosts are allowed on multiple lines.
# Can be specified with either hostnames or IP addresses.
# default: none
# trusted_hosts 1.1.1.1 1.1.1.2 1.1.1.3 \
# 2.3.2.3 3.4.3.4 5.6.5.6
#
# The number of nodes in your cluster.  This value is used in the
# creation of the cluster hash.
# default: 1024
# num_nodes  1024
#
# The number of custom metrics this gmond will be storing.  This
# value is used in the creation of the host custom_metrics hash.
# default: 16
# num_custom_metrics 16
#
# Run gmond in "mute" mode.  Gmond will only listen to the multicast
# channel but will not send any data on the channel.
# default: off
# mute on
#
# Run gmond in "deaf" mode.  Gmond will only send data on the multicast
# channel but will not listen/store any data from the channel.
# default: off
# deaf on
#
```

```
# Run gmond in "debug" mode.  Gmond will not background.  Debug messages
# are sent to stdout.  Value from 0-100.  The higher the number the more
# detailed debugging information will be sent.
# default: 0
# debug_level 10
#
# If you don't want gmond to setuid, set this to "on"
# default: off
# no_setuid  on
#
# Which user should gmond run as?
# default: nobody
# setuid     nobody
#
# If you do not want this host to appear in the gexec host list, set
# this value to "on"
# default: off
# no_gexec   on
#
# If you want any host which connects to the gmond XML to receive
# data, then set this value to "on"
# default: off
# all_trusted on
```

If you want to customize the operation of gmond, simply edit this file and save it to /etc/gmond.conf. You can create multiple gmond configurations by writing the configuration file to a different file, say /etc/gmond_test.conf, and the using the --config option of gmond to specify which configuration file to use.

```
# gmond --conf=/etc/gmond_test.conf
```

would start gmond with the settings in /etc/gmond_test.conf

3.3. Ganglia Web Frontend

Last Updated: $Date: 2002/09/10 20:01:17 $

Most configuration parameters reside in the `gmetad-webfrontend/conf.php` file. Here you may alter the template, gmetad location, RRDtool location, and set the default time range and metrics for graphs.

The static portions of the Ganglia website are themable. This means you can alter elements such as section lables, some links, and images to suit your individual tastes and environment. The `$template_name` variable names a directory containing the current theme. Ganglia uses TemplatePower (http://templatepower.codocad.com/) to implement themes. A user-defined skin must conform to the template interface as defined by the `default` theme.

Essentially, the variable names and START/END blocks in a custom theme must remain the same as the default, but all other HTML elements may be changed.

Other configuration variables in `conf.php` specify the location of gmetad's files, and where to find the rrdtool program. These locations need only be changed if you do not run gmetad on the webserver. Otherwise the default locations should work fine. The `$default_range` variable specifies what range of time to show on the graphs by default, with possible values of hour, day, week, month, year. The `$default_metric` parameter specifies which metric to show on the cluster view page by default.

4. Commandline Tools

Last Updated: $Date: 2002/10/16 18:35:13 $

4.1. Ganglia Metric Tool (gmetric)

The *Ganglia Metric Tool* (gmetric) allows you to easily monitor any arbitrary host metrics that you like expanding on the core metrics that gmond measures by default.

If you want help with the gmetric sytax, simply use the "help" commandline option

```
prompt> gmetric --help
gmetric 2.5.0

Purpose:
  The Ganglia Metric Client (gmetric) announces a metric
  value to all Ganglia Monitoring Daemons (gmonds) that are listening
  on the cluster multicast channel.

Usage: ganglia-monitor-core [OPTIONS]...
   -h         --help                  Print help and exit
   -V         --version               Print version and exit
   -nSTRING   --name=STRING           Name of the metric
   -vSTRING   --value=STRING          Value of the metric
   -tSTRING   --type=STRING           Either string|int8|uint8|int16|uint16|int32|uint32|float|doubl
   -uSTRING   --units=STRING          Unit of measure for the value e.g. Kilobytes, Celcius
   -sSTRING   --slope=STRING          Either zero|positive|negative|both (default='both')
   -xINT      --tmax=INT              The maximum time in seconds between gmetric calls (default=60)
   -cSTRING   --mcast_channel=STRING  Multicast channel to send/receive on (default='239.2.11.71')
   -pINT      --mcast_port=INT        Multicast port to send/receive on (default=8649)
   -iSTRING   --mcast_if=STRING       Network interface to multicast on e.g. 'eth1' (de-
fault='kernel decides')
```

```
   -lINT      --mcast_ttl=INT         Multicast Time-To-Live (TTL) (default=1)
```

The gmetric tool formats a special multicast message and sends it to all gmonds that are listening.

All metrics in ganglia have a *name*, *value*, *type* and optionally *units*. For example, say I wanted to measure the temperature of my CPU (something gmond doesn't do) then I could multicast this metric with *name*="temperature", *value*="63", *type*="int16" and *units*="Celcius".

Assume I have a program called `cputemp` which outputs in text the temperature of the CPU

```
prompt> cputemp
63
```

I could easily send this data to all listening gmonds by running

```
prompt> gmetric --name temperature --value ‘cputemp‘ --type int16 \
--units Celcius
```

Check the exit value of gmetric to see if it successfully sent the data: 0 on success and -1 on failure.

To constantly sample this *temperature* metric, you just need too add this command to your cron table.

4.2. Ganglia Cluster Status Tool (gstat)

The Ganglia Cluster Status Tool (gstat) is a commandline utility that allows you to get status report for your cluster. With time, it will be a more flexible way to query a gmond running locally or remotely.

Commandline Options

To get the commandline options simply run...

```
prompt> gstat --help
gstat 2.5.0

Purpose:
  The Ganglia Status Client (gstat) connects with a
  Ganglia Monitoring Daemon (gmond) and output a load-balanced list
  of cluster hosts

Usage: gstat [OPTIONS]...
   -h         --help             Print help and exit
   -V         --version          Print version and exit
   -a         --all              List all hosts.  Not just hosts running gexec (default=off)
   -d         --dead             Print only the hosts which are dead (default=off)
   -m         --mpifile          Print a load-balanced mpifile (default=off)
   -1         --single_line      Print host and information all on one line (default=off)
```

```
   -l         --list             Print ONLY the host list (default=off)
   -iSTRING   --gmond_ip=STRING  Specify the ip address of the gmond to query (default='127.0.0.1')
   -pINT      --gmond_port=INT   Specify the gmond port to query (default=8649)
```

Running gstat without any parameters will cause it print a load-balanced (least-loaded host first) list of all the hosts running gmond along with the process, load, and CPU information. If you want to see which hosts are down in your cluster, use the `--dead` gstat option. You can also have gstat produce a dynamic load-balanced mpimachine file with the `--mpifile` option.

Gstat Examples

Get a load-balanced list of hosts that are up...

```
prompt> gstat
CLUSTER INFORMATION
       Name: unspecified
      Hosts: 97
Gexec Hosts: 73
 Dead Hosts: 0
  Localtime: Mon Apr 22 16:58:43 2002

CLUSTER HOSTS
Hostname                     LOAD                         CPU              Gexec
 CPUs (Procs/Total) [     1,     5, 15min] [  User,  Nice, System, Idle]

mm92.millennium.berkeley.edu
    4 (    1/   97) [  1.10,  1.19,  0.99] [   5.9,   0.0,   0.5, 100.0] ON
mm98.Millennium.Berkeley.EDU
    4 (    0/   80) [  1.16,  1.67,  1.25] [   4.1,   0.0,   0.2,  98.5] ON
mm91.Millennium.Berkeley.EDU
    4 (    1/   87) [  1.67,  1.78,  1.69] [  25.0,   0.0,   0.7,  74.9] ON
mm75.millennium.berkeley.edu
    4 (    3/  103) [  1.85,  2.54,  1.83] [  72.6,   0.0,   0.2,  50.3] ON
mm67.millennium.Berkeley.EDU
    4 (    4/  112) [  1.89,  2.08,  1.38] [  81.4,   0.0,   0.1,  38.5] ON
mm87.millennium.berkeley.edu
    4 (    4/  112) [  1.95,  1.67,  1.27] [   3.2,   0.0,   0.4,  96.4] ON
mm83.millennium.Berkeley.EDU
    4 (    1/  120) [  2.00,  2.59,  2.24] [  25.0,   0.0,   0.0,  75.0] ON
mm10.millennium.Berkeley.EDU
    2 (    0/   77) [  0.00,  0.06,  0.07] [   0.2,   0.0,   0.0,  99.9] ON
...
```

To get create a dynamic load-balanced mpifile list

```
prompt> gstat --mpifile
```

```
mm56.Millennium.Berkeley.EDU:4
mm44.Millennium.Berkeley.EDU:4
mm31.Millennium.Berkeley.EDU:2
mm43.Millennium.Berkeley.EDU:4
mm15.Millennium.Berkeley.EDU:2
...
```

5. Getting Help

Jack Handey

The tired and thirsty prospector threw himself down at the edge of the watering hole and started to drink. But then he looked around and saw skulls and bones everywhere. "Uh-oh," he thought. "This watering hole is reserved for skeletons."

If you need help, a good starting point is the latest ganglia documentation (http://ganglia.sourceforge.net/docs/). If the manual doesn't answer your question, then the next place to look is the mailing list archives (https://sourceforge.net/mail/?group_id=43021). It is very possible that another ganglia user has had the same problem as you. If you still haven't found an answer, then send mail to the ganglia-general@lists.sourceforge.net.

To report bugs, submit patches or ask questions of the ganglia developers, send an email to ganglia-developers@lists.sourceforge.net.

6. ChangeLog

2002-08-23 Federico Sacerdoti
  * Increased maximum gmetric value size to 1472 chars from 255.
    (1472 = 1500 Eth MTU - IP and UDP header bytes) Better memory
    efficiency by storing only necessary bytes in hash.

2002-08-21 Matt Massie
  * Added gmetad to the monitoring core distribution

2002-08-16 Federico Sacerdoti
  * Added disk_free, disk_total, and part_max_used disk metrics for Linux.
    They do not double count automounted directories as 'df' sometimes
    does.

* Added LOCATION attribute to HOST tag for physical view of cluster.
* Enabled the use of hostnames in gmond's trusted hosts list.

2002-08-14 Matt Massie
* Updated gmetric to send the SLOPE and TMAX attributes to gmond
* Making the XML port more bullet-proof when a remote client closes its connection prematurely
* Changed the XML DTD to handle cases where a CLUSTER has no HOST or a HOST has no METRICs to report

2002-08-12 Matt Massie
* Added TN (Time Now = number of seconds since last msg received) and TMAX (Time Max = maximum distance between multicast messages) to the HOST and METRIC XML elements. Also added the SLOPE attribute to METRIC which can be set to "zero", "positive", "negative" or "both" in order to make the Round-Robin database structure more efficient

2002-08-08 Preston Smith
* Fixed AIX/libdnet build problems. (Turned out to be header file order in gmond.c and gmetric.c)

2002-08-06 Federico Sacerdoti
* Added a "timestamp" type to gmond and gmetric. Also changed some unit names. Added a OWNER and LATLONG attribute to the CLUSTER tag, both set in the conf file.

2002-07-24 Matt Massie
* Stitched in the libdnet into the ganglia distribution to handle the low-level networking necessary for dynamic routing and interface lookup.

2002-07-05 Preston Smith
* Added a patch from Davide Tachella to get CPU MHz on AIX, along with appropriate autoconf magic to detect it, and tell how to get the libraries that report it if they're not found.

2002-06-26 Matt Massie
* Fixed the heap corruption and thread madness on Solaris boxes.

2002-06-24 Preston Smith
* Merged linux-alpha platform specific code into linux.c

(cpu # and cpu MHz are the only different metrics)
Fixed configure.in to reflect this.

2002-06-21 Matt Massie
* Completely rewrote the underlying network library in order to allow
for future growth (e.g. automatic discovery and all network interfaces,
sending/receiving on multiple network interfaces, multicast forwarding)
and to fix known bugs

2002-06-20 Preston Smith
* Tweaked cpu_speed_func to only use the existing code for
i386, ia64, and hp-parisc.
Added code to get cpu clock speed for powerpc
Sparc, s390, and arm don't report their cpu clock rate.
* Added lines to machine_type_func to report arch name for
sparc, alpha, powerpc, m68k, mips, arm, hppa, s390
Couldn't log into an m68k or mips box to check them out, so they may
or may not successfully run gmond.

2002-06-20 Preston Smith
* Added patches to aix.c to support AIX 5.
* Added FreeBSD to the list that has SUPPORT_GEXEC=0 by default
(configure.in)

2002-06-17 Steven Wagner
* Updated autoconf to set SUPPORT_GEXEC flag if building on a supported
platform (use --enable-gexec). gmond should report full IRIX metrics
now, and run "out of the box."

2002-05-31 Matt Massie
* Fixed a bug reading the XML output where some hosts had timestamps newer than
the cluster timestamp and were mistakenly marked as down

2002-05-13 Matt Massie
* Changed pre_process_node() gethostbyaddr() error reporting to be more sane
for environments which nameservice problems
* Updated listen.c to prevent out-of-range multicasted key values from crashing
gmond in debug mode

2002-05-06 Matt Massie

```
  * Added an explicit heartbeat metric to gmond along with a GMOND_STARTED
    HOST attribute which removes the need for a 90 second delay in gmond restarts
  * Updated linux.c to correctly support the ia64 architecture

2002-05-01 Matt Massie
  * Gmond now has a configuration file for its configuration instead of
    passing commandline arguments

2002-04-26 Asaph Zemach
  * Completely rewrote ./gmond/machines/linux.c to be much more efficient by
    removing the need for 2 threads and pthread mutexes.

2002-04-26 Matt Massie
  * Modified net.c setsockopt() functions to be more portable

2002-04-19 Matt Massie
  * Fixed a bug in the .spec file where upgrades failed if gmond was not up
  * Updated autoconf to correctly configure ia64 machines

2002-04-18 Matt Massie
  * Enabled setsockopt SO_REUSEADDR for the mcast_join socket in order allow
    multiple instances of gmond to listen to the same multicast channel

2002-04-17 Matt Massie
  * Added the --list and --single_line option to gstat for output flexibility

2002-04-17 Alan Hagge
  * Added preliminary support for IRIX

2002-04-16 Matt Massie
  * Add the "gexec" metric in preparation for the release of gexec
  * Updated the gexec_cluster() function in libganglia to pull in more data
    from the XML and handle unresolved names and names without domains correctly
  * Updated gstat to print the updated information from gexec_cluster()
  * Added a --no_gexec flag to gmond for hosts that are not part of the
    computation cluster (file servers, frontends, etc)
  * Added a --all_trusted flag where gmond sees ALL hosts as trusted as
    suggested by Martin Knoblauch
  * Removed goto statement in pre_process_node() function to avoid rare looping bug
    thanks to feedback from Mike Snitzer
```

2002-04-12 Chris Elmquist
  * Updated gmetric.c to check if the name of the metric is empty to prevent
    the problem of blank metrics showing up in gmonds

2002-04-08 Matt Massie
  * Fixed a bug in the barrier code used in gmond
  * Fixed mcast_connect() on non-Linux systems related to the IP_MULTICAST_TTL option
  * Added the "--no_setuid" and "--setuid" flags to provide more euid flexibility

2002-04-05 Preston Smith
  * Added support for AIX machines

2002-04-04 Dave Wallace
  * Fixed a bug in gmond which caused big-endian architectures to incorrectly
    store multicast data and therefore misreport on the XML port.
  * Added a key value check of the XDR multicast data in ./gmond/listen.c

2002-04-04 Matt Massie
  * Updated ./lib/gexec_func.c gexec_cluster() function to handle hosts with
    no domainname correctly (as pointed out by David Wallace
  * Updated the program commandline options to obey the GNU Coding Standard
  * Added a --debug_level parameter to gmond to allow debugging the daemon
    without recompiling the daemon
  * Updated the --trusted_host option to allow multiple instances of the option
  * Added a --mcast_ttl gmond option to allow you to modify the
    Time-To-Live (TTL) of the outgoing multicast messages

2002-04-02 Neil Spring
  * Updated the gmond Makefile to cleanup the machine.c link on "make clean"

2002-04-02 Doc Schneider
  * Update the way the number of CPUs are collected in order to workaround a
    bug on AMD-based systems.

2002-03-28 Matt Massie
  * Change distribution tree to fit the ganglia model. All gmond clients are
    in the ./gmond directory and all gregisterd clients in ./gregisterd

2002-03-26 Matt Massie

```
  * Removed the use of streams (via fdopen) on the XML socket and replaced
    it with write()s on the socket descriptor to work around Linux bug under
    high-stress conditions

2002-03-25 Matt Massie
  * Added thread barriers to gmond initialization to ensure listening threads
    exist before the threads which multicast are created

2002-03-24 Matt Massie
  * Use XML_ParseBuffer() in gexec_cluster() in order to avoid
    double copying of buffers (XML input from gmond). Faster.

2002-03-22 Matt Massie
  * Updated gexec_cluster_free() to ensure no memory leaks even when
    cluster.num_nodes and cluster.num_dead_nodes equals zero
  * Used setvbuf to ensure the gmond and libganglia are using line buffering

2002-03-21 Matt Massie
  * Updated net.h to include netinet/in.h
  * Changed sockaddr_in_new function in net.c to plug a potential memory leak
  * Added XML_ParserFree() call to gexec_cluster() lib call to plug up
    memory leak
  * Modified ./gmond/server.c to prevent crashes under heavy stress conditions
  * Added fclose() calls to gexec_cluster() to plug a memory leak
  * Added gexec_cluster_free() call to gstat for good measure

2002-03-15 Preston Smith
  * Patched a wrong sysctl to get free memory for freebsd machines

2002-03-15 Matt Massie
  * Added mute and deaf mode for gmond
  * Created a new ganglia-monitor-core-lib distribution for the
    libganglia library.
  * Moved all the documentation to DocBook, added much more information
    and updated/removed what was there.  Output docs to ./docs directory
    of the distribution in both HTML and PDF form.  Also installed
    Doxygen to document libganglia.
  * Add the Ganglia Status Tool (gstat) which allows you to check the status
    of your cluster from the commandline.  Hosts are sorted with least-loaded
    nodes at the top of the list.
```

2002-03-12 Matt Massie
* Removed the need for the POSIX mutex in pre_process_node() allowing for faster processing of incoming multicast data

2002-03-11 Matt Massie
* Changed gmond to not count itself as a running process when reporting the number of running processes.

2002-03-06 Doc Schneider
* Changed the way ganglia builds RPMs to support non-root builds as suggested by an anonymous SourceForge user

2002-02-28 Matt Massie
* Added a new CLUSTER element to the XML with two attributes: NAME and LOCALTIME. Necessary for monitoring clusters in many different timezones.

2002-02-27 Matt Massie
* Fixed the getopt_long() call in gmond.c thanks to feedback from Meik Hellmund. The getopt_long() parameters didn't match the switch() statement breaking the "trusted_host" option.
* Fixed a bug in gmond where connections from untrusted hosts caused segfaults. Error caused by passing datum_free() a NULL pointer in server_thread() of ./gmond/server.c.
* Changed the way transient nameservice errors are handled by pre_process_node() in ./gmond/listen.c. Previously, transient errors were retried but now they are treated as errors (although gmond will continue trying to resolve the host when it gets a new multicast packet from it)
* Updated the ganglia.spec file to merge gmond and gmetric into a single RPM, fixed some small bugs, and updated the RPM information.
* Changed the gmetric options to also support long options and updated the help output (from -h, --help) to be much more descriptive

2002-02-27 Preston Smith
* Updated the FreeBSD monitoring code to include all metrics which are monitored under Linux except number of running processes, absolute cpu idle time, and shared memory. SMP users may find that freebsd's cp_time sysctls is not completely accurate under FreeBSD stable

meaning CPU%s might be inaccurate. However, it works under FreeBSD-CURRENT.

2002-02-22 Matt Massie
* Added the getopt source to the ganglia library for system that don't have the getopt API available (Solaris, FreeBSD, etc)

2002-02-21 Matt Massie
* Changed the "safe_host" option to "trusted_host" to make it clearer
* Added the "num_nodes" and "num_custom_metrics" commandline options

2002-02-20 Matt Massie
* Completely rewrote the underlying hash library because the original hash functions where over-engineered and had some memory leaks. New hash functions are superlight and fast. Built test program and profiled/traced all memory functions using mpatrol. No leaks.
* Updated code to catch when transient nameservice errors occur and retry. Correctly handle hosts the don't resolve instead of treating as an error.
* Added a patch submitted by Joshua J England for gmond to correctly report the number of CPUs and their speed on alpha architectures
* Added a patch submitted by Eirikur Hallgrimsson and written by Yaroslav Klyukin for gmetric which allows users to chose which network interface gmetric multicasts metric data

2002-02-12 Matt Massie
* Reduced the number of total threads by one by removing the for(;;)pause() spin and having the main thread do server work

2002-02-07 Matt Massie
* created the function my_inet_ntop() function in libganglia to deal with the limitations of inet_ntoa in a multi-threaded environment
* changed the self-organzing behavior of gmond to recognize when a transient error occured on a remote gmond process
* added verbose error checking of gethostbyaddr() in listen.c

2002-02-04 Matt Massie
* fixed a bug in the cpu_num_func() which reported AMD systems as having twice as many CPUs as they really did

2002-02-01 Matt Massie

* increased the speed of the host security check for the XML port
* added commandline options for almost all compile-time opts for gmond
* added a --safe_host option to allow a host outside of the multicast
  channel to connect
* now gmond strips all quotes (") from gmetric data to keep XML well-formed
* improved the self-organizing behavior of the gmonds

2002-01-18 Matt Massie

* changed the name of the distribution from ganglia to
  ganglia-monitor-core to be more descriptive
* modified ./gmond/server.c in order to keep gmond from crashing
  when a client closes the connection prematurely
* added patch to ./gmond/gmond.c from Neil Spring and Brian Youngstrom to
  add command line option and properly setuid to nobody
* added gmond.init.SuSE to the distribution. Contributed by Oliver Mossinger.

7. License

Ganglia is released under the following BSD license.

8. Notes

8.1. Solaris, IRIX, Tru64

Here is an email from Steve Wagner about the state of the ganglia on Solaris, IRIX and Tru64. Steve is to thank for porting ganglia to Solaris and Tru64. He also helped with the IRIX port.

```
State of the IRIX port:

*  CPU percentage stuff hasn't improved despite my efforts.  I fear there
   may be a flaw in the way I'm summing counters for all the CPUs.
*  Auto-detection of network interfaces apparently segfaults.
*  Memory and load reporting appear to be running properly.
*  CPU speed is not being reported properly on multi-proc machines.
*  Total/running processes are not reported.
*  gmetad untested.
*  Monitoring core apparently stable in foreground, background being tested
(had a segfault earlier).

State of the Tru64 port:

*  CPU percentage stuff here works perfectly.
*  Memory and swap usage stats are suspected to be inaccurate.
*  Total/running processes are not reported.
*  gmetad untested.
*  Monitoring core apparently stable in foreground and background.

State of the Solaris port:
*  CPU percentages are slightly off, but correct enough for trending
   purposes.
*  Load, ncpus, CPU speed, breads/writes, lreads/writes, phreads/writes,
   and rcache/wcache are all accurate.
*  Memory/swap statistics are suspiciously flat, but local stats bear
   this out (and they *are* being updated) so I haven't investigated
   further.
*  Total processes are counted, but not running ones.
*  gmetad appears stable

Anyway, all three ports I've been messing with are usable and fairly
stable.  Although there are areas for improvement I think we really can't
keep hogging all this good stuff - what I'm looking at is ready for
release.
```

8.2. Debian Users

Here is an email message from Preston Smith for Debian users

```
 Debian packages for Debian 3.0 (woody) are available at
  http://www.physics.purdue.edu/~psmith/ganglia
 (i386, sparc, and powerpc are there presently, more architectures will
  appear when I get them built.)
 Packages for "unstable" (sid) will be available in the main Debian
  archive soon.

 Also, a CVS note: I checked in the debian/ directory used to create
 debian packages.
```

8.3. Multihomed Machines

Here is an email I sent to a user having problems with a multi-homed machine.

```
i need to add a section in the documentation talking about this since it
seems to be a common question.

when you use...

mcast_if eth1

.. in /etc/gmond.conf that tells gmond to send its data out the "eth1"
network interface but that doesn't necessarily mean that the source
address of the packets will match the "eth1" interface.  to make sure that
data sent out eth1 has the correct source address run the following...

% route add -host 239.2.11.71 dev eth1

... before starting gmond.  that should do the trick for you.

-matt

> I have seen some post related to some issues
> with gmond + multicast running on a dual nic
> frontend.
>
> Currently I am experiencing a weird behavior
>
> I have the following setup:
```

```
>
>   ------------------------
>   | web server + gmetad |
>   ------------------------
>              |
>              |
>              |
>     ----------------------
>     |   eth0 A.B.C.112   |
>     |                    |
>     |  Frontend + gmond  |
>     |                    |
>     | eth1 192.168.100.1 |
>     ----------------------
>              |
>              |
>
>        26 nodes each
>           gmond
>
> In the frontend /etc/gmond.conf I have the
> following statement: mcast_if  eth1
>
> The 26 nodes are correctly reported.
>
> However the Frontend is never reported.
>
> I am running iptables on the Frontend, and I am seing
> things like:
>
> INPUT packet died: IN=eth1 OUT= MAC= SRC=A.B.C.112 DST=239.2.11.71
> LEN=36 TOS=0x00 PREC=0x00 TTL=1 ID=53740 DF PROTO=UDP SPT=41608 DPT=8649
> LEN=16
>
> I would have expected the source to be 192.168.100.1 with mcast_if eth1
>
> Any idea ?
```

8.4. Cisco Catalyst Switches

Perhaps information regarding gmond on networks set up through cisco catalyst switches should be mentioned in the ganglia documentation. I think by default multicast traffic on the catalyst will flood all devices unless

configured properly. Here is a relavent snipet from a message forum, with a link to cisco document.

--
If what you are trying to do, is minimizing the impact on your network due to a multicast application, this link may describe what you want to do:
http://www.cisco.com/warp/public/473/38.html

We set up our switches according to this after a consultant came in and installed an application multicasting several hundred packets per second. This made the network functional again.