

MRG Messaging

Installation Guide

1.0



ISBN:
Publication date:

MRG Messaging

This book will show you how to download and install the MRG Messaging component of the Red Hat Enterprise MRG distributed computing platform. To learn how to program MRG Messaging applications, see the MRG Messaging Tutorial.

MRG Messaging: Installation Guide

Copyright © 2008 Red Hat, Inc

Copyright © 2008 Red Hat, Inc. This material may only be distributed subject to the terms and conditions set forth in the Open Publication License, V1.0 or later with the restrictions noted below (the latest version of the OPL is presently available at <http://www.opencontent.org/openpub>).

Distribution of substantively modified versions of this document is prohibited without the explicit permission of the copyright holder.

Distribution of the work or derivative of the work in any standard (paper) book form for commercial purposes is prohibited unless prior permission is obtained from the copyright holder.

Red Hat and the Red Hat "Shadow Man" logo are registered trademarks of Red Hat, Inc. in the United States and other countries.

All other trademarks referenced herein are the property of their respective owners.

The GPG fingerprint of the security@redhat.com key is:

CA 20 86 86 2B D6 9D FC 65 F6 EC C4 21 91 80 CD DB 42 A6 0E

1801 Varsity Drive
Raleigh, NC 27606-2072
USA
Phone: +1 919 754 3700
Phone: 888 733 4281
Fax: +1 919 754 3701
PO Box 13588
Research Triangle Park, NC 27709
USA

| | |
|---|-----|
| Preface | vii |
| 1. Document Conventions | vii |
| 2. We Need Feedback! | ix |
| 1. Installing MRG Messaging | 1 |
| 1. Installing MRG Messaging on Red Hat Enterprise Linux 5 | 1 |
| 2. Installing MRG Messaging on Red Hat Enterprise Linux 4 | 2 |
| 3. Available Packages — RPM | 3 |
| 2. Starting the Broker | 5 |
| 3. Options for Running the Broker | 7 |
| 1. Using Modules with the Broker | 9 |
| 2. Setting the Store Journal Size | 9 |
| 3. Logging Broker Errors | 10 |
| 4. Using Persistence with MRG Messaging | 13 |
| 5. Command Line Utilities for MRG Messaging | 15 |
| 6. Federation in MRG Messaging | 17 |
| 7. User Authentication with MRG Messaging | 19 |
| 8. More Information | 21 |
| A. Revision History | 23 |

Preface

Red Hat Enterprise MRG.

This book contains basic installation information for the MRG Messaging component of Red Hat Enterprise MRG. Red Hat Enterprise MRG is a high performance distributed computing platform consisting of three components:

1. *Messaging* — Cross platform, high performance, reliable messaging using the Advanced Message Queuing Protocol (AMQP) standard.
2. *Realtime* — Consistent low-latency and predictable response times for applications that require microsecond latency.
3. *Grid* — Distributed High Throughput (HTC) and High Performance Computing (HPC).

All three components of Red Hat Enterprise MRG are designed to be used as part of the platform, but can also be used separately.

MRG Messaging.

MRG Messaging is an open source, high performance, reliable messaging distribution that implements the Advanced Message Queuing Protocol (AMQP) standard. MRG Messaging is based on [Apache Qpid¹](#), but includes persistence options, additional components, Linux kernel optimizations, and operating system services not found in the Qpid implementation. We have worked closely with companies that rely heavily on high performance messaging, and created a system to meet their real-world needs.

This guide shows you how to install MRG Messaging and start the broker, and explains the basic options available. If you want to write your own applications for use with MRG Messaging, you should also look at the *MRG Messaging Tutorial*.

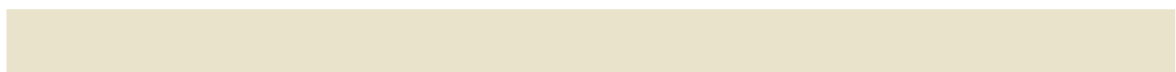
1. Document Conventions

Certain words in this manual are represented in different fonts, styles, and weights. This highlighting indicates that the word is part of a specific category. The categories include the following:

Courier font

Courier font represents commands, file names and paths, and prompts.

When shown as below, it indicates computer output:



¹ <http://cwiki.apache.org/qpid/>

| | | | |
|---------|-------------|------|--------------------|
| Desktop | about.html | logs | paulwesterberg.png |
| Mail | backupfiles | mail | reports |

Courier font

Bold Courier font represents text that you are to type, such as: `service jonas start`

If you have to run a command as root, the root prompt (#) precedes the command:

```
# gconftool-2
```

Courier font

Italic Courier font represents a variable, such as an installation directory:

```
install_dir/bin/
```

font

Bold font represents **application programs** and **text found on a graphical interface**.

When shown like this: **OK**, it indicates a button on a graphical application interface.

Additionally, the manual uses different strategies to draw your attention to pieces of information. In order of how critical the information is to you, these items are marked as follows:



Note

A note is typically information that you need to understand the behavior of the system.



Tip

A tip is typically an alternative way of performing a task.



Important

Important information is necessary, but possibly unexpected, such as a configuration change that will not persist after a reboot.



Caution

A caution indicates an act that would violate your support agreement, such as recompiling the kernel.



Warning

A warning indicates potential data loss, as may happen when tuning hardware for maximum performance.

2. We Need Feedback!

If you find a typographical error in this manual, or if you have thought of a way to make this manual better, we would love to hear from you! Please submit a report in Bugzilla:

<http://bugzilla.redhat.com/bugzilla/> against the product **Red Hat Enterprise MRG**.

When submitting a bug report, be sure to mention the manual's identifier:

Messaging_Installation_Guide

If you have a suggestion for improving the documentation, try to be as specific as possible when describing it. If you have found an error, please include the section number and some of the surrounding text so we can find it easily.

Installing MRG Messaging

In order to install MRG Messaging you will need to have registered your system with [Red Hat Network](#)¹. This table lists the Red Hat Enterprise MRG channels available on Red Hat Network for MRG Messaging.

| Channel Name | Operating System | Architecture |
|----------------------------|------------------|--------------|
| Red Hat MRG Messaging | RHEL-4 AS | 32bit, 64bit |
| Red Hat MRG Messaging | RHEL-4 ES | 32bit, 64bit |
| Red Hat MRG Messaging | RHEL-5 Server | 32bit, 64bit |
| Red Hat MRG Messaging | Non-Linux | 32bit |
| Red Hat MRG Messaging Base | RHEL-4 AS | 32bit, 64bit |
| Red Hat MRG Messaging Base | RHEL-4 ES | 32bit, 64bit |
| Red Hat MRG Messaging Base | RHEL-5 Server | 32bit, 64bit |

Table 1.1. Red Hat Enterprise MRG Channels Available on Red Hat Network



Important

Before you install Red Hat Enterprise MRG check that your hardware and platform is supported. A complete list is available on the [Red Hat Enterprise MRG Supported Hardware Page](#)².

1. Installing MRG Messaging on Red Hat Enterprise Linux 5

1. Install the MRG Messaging group using the `yum` command.

```
# yum groupinstall "MRG Messaging"
```

¹ <https://rhn.redhat.com/help/about.pxt>

2. You can check the installation location and that the components have been installed successfully by using the `rpm -ql` command with the name of the package you installed. For example:

```
# rpm -ql rhm
/usr/lib/qpiddlibdbstore.so.0
/usr/lib/qpidd/libdbstore.so.0.1.0
/usr/share/doc/rhm-0.2
/usr/share/doc/rhm-0.2/COPYING
/usr/share/doc/rhm-0.2/README
/var/rhm
```



Note

If you find that yum is not installing all the dependencies you require, make sure that you have registered your system with [Red Hat Network³](#).

2. Installing MRG Messaging on Red Hat Enterprise Linux 4

1. Install the MRG Messaging components using the `up2date` command.

```
# up2date python-qpidd qpidd-java-client qpidd-devel rhm rhm-docs
```

2. You can check the installation location and that the components have been installed successfully by using the `rpm -ql` command with the name of the package you installed. For example:

```
# rpm -ql rhm
/usr/lib/qpiddlibdbstore.so.0
/usr/lib/qpidd/libdbstore.so.0.1.0
/usr/share/doc/rhm-0.2
/usr/share/doc/rhm-0.2/COPYING
/usr/share/doc/rhm-0.2/README
/var/rhm
```

**Note**

If you find that `up2date` is not installing all the dependencies you require, make sure that you have registered your system with [Red Hat Network](#)⁴.

3. Available Packages — RPM

This section lists the RPM packages available for MRG Messaging.

| RPM Package Name | Description | Language | Architecture |
|---------------------------|--|----------|--------------------------|
| <code>qpidd</code> | MRG Messaging broker (Apache Qpid binaries for i386). | C++ | x86 (AMD or Intel 32bit) |
| <code>rhm</code> | MRG Messaging libraries, providing guaranteed message delivery. | C++ | x86 (AMD or Intel 32bit) |
| <code>qpiddc</code> | MRG Messaging client libraries. Required to run the broker. | C++ | x86 (AMD or Intel 32bit) |
| <code>qpiddc-devel</code> | C++ client libraries, including header files, developer documentation, and symbolic links to shared libraries. | C++ | x86 (AMD or Intel 32bit) |

Table 1.2. MRG Messaging Packages - i386

| RPM Package Name | Description | Language | Architecture |
|---------------------|---|----------|-------------------------|
| <code>qpidd</code> | MRG Messaging broker (Apache Qpid binaries for x86). | C++ | x86 (AMD64 or Intel 64) |
| <code>rhm</code> | MRG Messaging libraries, providing guaranteed message delivery. | C++ | x86 (AMD64 or Intel 64) |
| <code>qpiddc</code> | MRG Messaging | C++ | x86 (AMD64 or Intel |

| RPM Package Name | Description | Language | Architecture |
|------------------|--|----------|-------------------------|
| | client libraries. Required to run the broker. | | 64) |
| qpidd-devel | C++ client libraries, including header files, developer documentation, and symbolic links to shared libraries. | C++ | x86 (AMD64 or Intel 64) |

Table 1.3. MRG Messaging Packages - x86_64

| RPM Package Name | Description | Language | Architecture |
|-------------------|---|----------|--------------------------|
| qpidd-java-client | Java client library including JMS implementation | Java | Architecture Independent |
| qpidd-java-common | Java client and broker (to be released) common library. | Java | Architecture Independent |

Table 1.4. MRG Messaging Packages - Java

| RPM Package Name | Description | Language | Architecture |
|------------------|--|----------|--------------------------|
| amqp | AMQP specification (used internally by the Python client). | Python | Architecture Independent |
| python-qpidd | Python client libraries and command line utilities. | Python | Architecture Independent |

Table 1.5. MRG Messaging Packages - Python

| RPM Package Name | Description | Language | Architecture |
|------------------|--|------------|--------------------------|
| rhm-docs | Installation Guide, Tutorial and source code examples. | US English | Architecture Independent |

Table 1.6. MRG Messaging Packages - Documentation

Starting the Broker

Starting the Broker

1. By default, the broker is installed in `/usr/sbin/`. If this is not on your path, you will need to type the whole path to start the broker:

```
# /usr/sbin/qpidd -t
[date] [time] info Loaded Module: libbdbstore.so.0
[date] [time] info Locked data directory: /var/lib/qpidd
[date] [time] info Management enabled
[date] [time] info Listening on port 5672
```

The `-t` or `--trace` option enables debug tracing, printing messages to the terminal.

2. To stop the broker, type **CTRL+C** at the shell prompt

```
[date] [time] notice Shutting down.
[date] [time] info Unlocked data directory: /var/lib/qpidd
```

3. For production use, MRG Messaging is usually run as a service. To start the broker as a service, run the following command as the root user:

```
# service qpidd start
Starting qpidd daemon:      [ OK ]
```

4. You can check on the status of the service using the `service status` command and stop the broker with `service stop`.

```
# service qpidd status
qpidd (pid PID) is running...

# service qpidd stop
Stopping qpidd daemon:      [ OK ]
```


Options for Running the Broker

The broker can be run with a number of options. An overview of the most common options are given here.

Setting Options Using the Configuration File

1. For options that persist across sessions, you can put the options in the configuration file. Open the `/etc/qpidd.conf` file in your preferred text editor to make the necessary changes.
2. This example uses the configuration file to store any output to the `/var/log/qpidd.log` file, and enable debug tracing. Changes will take effect from the next time the broker is started and will be used in every subsequent session.

```
# Configuration file for qpidd
log-output=/var/log/qpidd.log
trace=1
```

3. If you are running the broker as a service, you will need to restart the service once you have made the changes.

```
# service qpidd restart
Stopping qpidd daemon:          [ OK ]
Starting qpidd daemon:         [ OK ]
```

4. If you are *not* running the broker as a service, you can now start the broker as normal.

```
# /usr/sbin/qpidd -t
[date] [time] info Locked data directory: /var/lib/qpidd
[date] [time] info Management enabled
[date] [time] info Listening on port 5672
```

Setting Options Using the Command Line

To set options for a single instance, add the option to the command line when you start the broker. You will need to be the root user.

1. This example uses command line options to start the broker with debug tracing and store any output to the `/var/log/qpidd.log` file. These options will need to be explicitly stated every time the broker is run.

```
# /usr/sbin/qpidd -t --log-output /var/log/qpidd.log
```

Common Options.

For more options, type `# man qpidd` or `# /usr/sbin/qpidd --help` at the shell prompt.

| General options for running the broker | |
|--|---|
| -t | This option enables debug tracing, with output printed to the screen. |
| -p <Port_Number> | Instructs the broker to use the specified port. Defaults to port 5672. It is possible to run multiple brokers simultaneously by using different port numbers. |
| -v | Displays the installed version. |
| -h | Displays the help message. |

Table 3.1. General Broker Options

| Options for running the broker as a service | |
|---|---|
| -d | This option instructs rhmd to run in the background as a daemon. Messages retrieved using a consumer are displayed, but any output usually displayed by the broker is suppressed. |
| -q | When the broker is running as a daemon this command shuts the broker down politely; that is, by closing the child processes, followed by the parent processes. |
| -c | This command checks if the daemon is already running. If it is running, it returns the process ID number. |
| -d --wait=<seconds> | This sets the maximum wait time (in seconds) for the daemon to initialize. If the daemon has not successfully completed initialization within this time, an error is returned. This option must be used in conjunction with the -d option, or it will be ignored. |

Table 3.2. Options for running the broker as a service (daemon)

1. Using Modules with the Broker

| Options for using modules with the broker | |
|--|---|
| <code>--load-module <i>MODULENAME</i></code> | Instructs the broker to use the specified module as a plug-in. |
| <code>--module-dir <DIRECTORY></code> | Causes the broker to use a different module directory. |
| <code>--no-module-dir</code> | Causes the broker to ignore module directories. |
| <code>--data-dir <i>DIRECTORY</i></code> | Disables storage of configuration information and other data by the broker. If the default data directory (<code>/var/lib/qpidd</code>) exists, it will be ignored. |
| <code>--no-data-dir</code> | Disables persistent storage. If a data directory exists, it will be ignored. |

Table 3.3. Options for using modules with the broker

Getting Help with Modules.

To see the help text for any module, you will need to load the module in order to use the `--help` command:

```
# /usr/sbin/qpidd --load-module MODULENAME --help
```

2. Setting the Store Journal Size


It is important to set the store journal size to match the anticipated persistent message queue depth. Since the store uses a fixed-size circular file buffer, it is possible for the store to run out of space to enqueue messages if consumers are slow or messages are too large.

Each queue that is marked persistent will cause the broker to create an instance of the store together with its files. The broker will stop accepting persistent messages when approximately 80% of the journal capacity is reached. Consuming of messages (dequeuing), however will be allowed to continue. Once the dequeued messages have cleared sufficient space, enqueues will continue normally. Allowing messages to dequeue will free space in the journal. It is particularly important to dequeue the earliest messages in the journal as early as possible, as these will hold up the write process in the buffer.

As a rule of thumb, the journal capacity should be about 25% greater than the total persistent message size expected to be stored on the disk at any one moment in time.

| Options for setting the store journal size | |
|--|---|
| <code>--num-jfiles</code> | Set the number of files for each instance of the persistence journal. The default is 8. |
| <code>-jfile-size-pgs</code> | Set the size of each journal file in multiples of 64KB. The default is 24. |

Table 3.4. Options for setting the journal size



Warning

A totally full condition on the journal (in which there is no more write space in the circular buffer) is a fatal condition. It is possible to read the messages in a full journal, but not to dequeue them (as dequeuing requires the ability to write dequeue records). For this reason further enqueues are disabled at 80% capacity.

3. Logging Broker Errors

Logging is enabled in the broker by default for all errors. Output is sent to `stderr` - the standard output error stream (usually this is the shell prompt). You can choose to log using `syslog`, which allows you to store logs to either a local file or to a remote logging server.

| Options for logging with <code>syslog</code> | |
|--|---|
| <code>--log-output FILE</code> | <p>Send log output to the specified filename. <code>FILE</code> can also be one of the special values:</p> <ul style="list-style-type: none"> • <code>stderr</code> Standard output error stream • <code>stdout</code> Standard output • <code>syslog</code> Use <code>syslog</code> for log output <p>The default is <code>stderr</code>.</p> |
| <code>--syslog-name NAME</code> | Specify the name to use in <code>syslog</code> messages. The default is <code>qpidd</code> . |
| <code>--syslog-facility LOG_XXX</code> | Specify the facility to use in <code>syslog</code> messages. The default is <code>LOG_DAEMON</code> . |

Table 3.5. Logging Options

Using Persistence with MRG Messaging

A persistence library allows MRG Messaging to store messages and queue configuration, ready to be reloaded in the event of machine or network failure. When the persistent store module is loaded, it allows messages and other persistent state information to be recovered when a broker is restarted.

In order for messages to be stored the persistence store must be loaded. By default, the persistence store is loaded if the `libqdbstore.so` library is located in `/usr/lib/qpidd`. See [Table 4.1, “Persistence Options”](#) for options on how to change this behaviour.

In addition to loading the persistent store, queues and messages also need to be identified as durable. This can be done in the client application or by using the `qpidd-config` command line tool. See the *MRG Messaging Tutorial* for more information about creating client applications.



Important

If the persistence module is not loaded, messages and the broker state will not be stored to disk, even if the queue and messages sent to it are marked persistent.

Persistence Options

| | |
|---|---|
| <code>--store-directory</code> <i>DIRECTORY</i> | Specifies the directory used for storage of persistence configuration information. The default is <code>/var/lib/qpidd</code> . |
|---|---|

Table 4.1. Persistence Options



Important

Only one running broker can access a data directory at a time. If another broker attempts to access the data directory it will fail with an error stating “Exception: Data directory is locked by another process.”

Command Line Utilities for MRG Messaging

MRG Messaging contains a number of command line utilities for monitoring and configuring messaging brokers.

`qpuid-config`

Display and configure exchanges, queues, and bindings in the broker

`qpuid-route`

Display and configure broker federation, including routing and links between brokers

`qpuid-tool`

Access configuration, statistics, and control within the broker

`qpuid-queue-stats`

Monitor the size and enqueue/dequeue rates of queues in a broker

Install the command line utilities

1. The command line utilities are included in the python client library package. Follow the installation instructions in [Chapter 1, Installing MRG Messaging](#) and install the `python-qpuid` and `amqp` packages.
2. To start each of the tools, type its name at the command prompt:

```
qpuid-config
```

```
# qpuid-config
```

```
qpuid-route
```

```
# qpuid-route
```

```
qpuid-tool
```

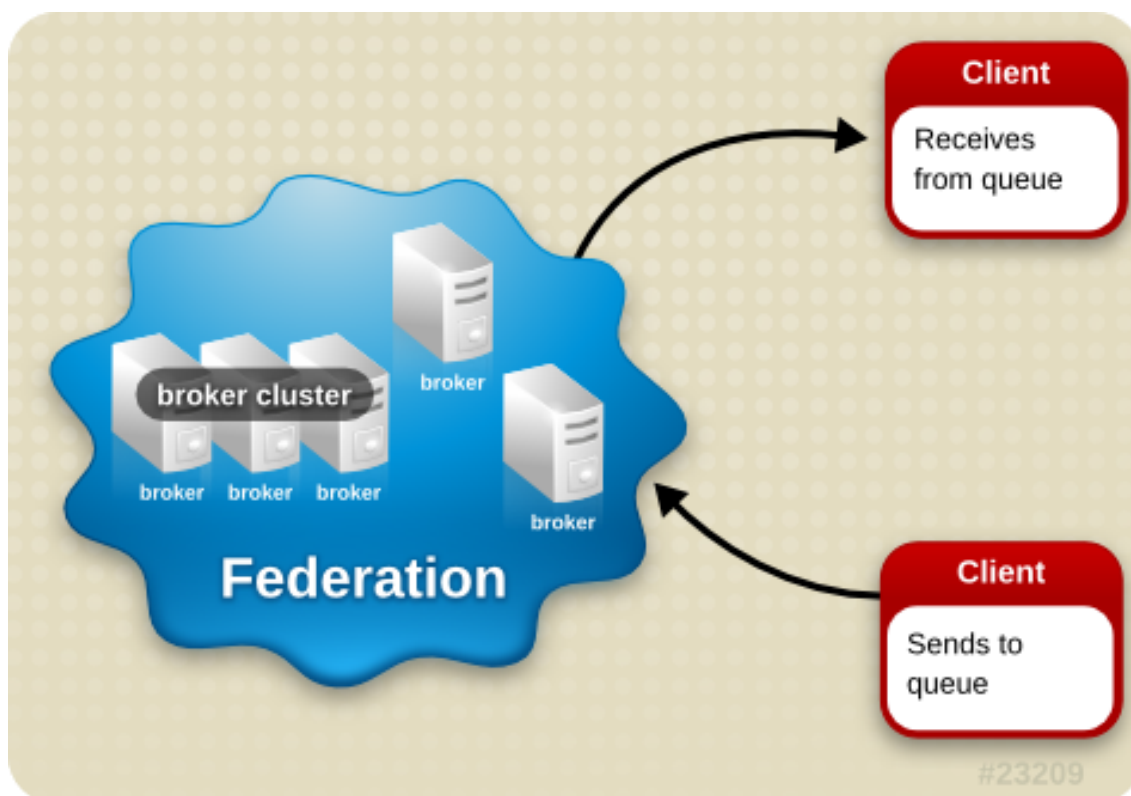
```
# qpuid-tool
```

```
qpuid-queue-stats
```

```
# qpid-queue-stats
```

Federation in MRG Messaging

Federation is used to provide geographical distribution of brokers. A number of individual brokers, or clusters of brokers, can be federated together. This allows client machines to see and interact with the federation as though it were a single broker. Federation can also be used where client machines need to remain on a local network, even though their messages have to be routed out.



A number of individual brokers, or clusters of brokers, can be federated together.

Managing Federation with `qpidd-route`

Federation can be managed from the command line with `qpidd-route`, which can display and configure broker federation, including routing and links between brokers.

1. The command line utilities are included in the python client library package. Follow the installation instructions in [Chapter 1, Installing MRG Messaging](#) and install the `python-qpidd` and `amqp` packages.
2. To view the available commands, use the `--help` option with the `qpidd-route` command:

```
# qpidd-route --help
Usage: qpidd-route [OPTIONS] link add <dest-broker> <src-broker>
       qpidd-route [OPTIONS] link del <dest-broker> <src-broker>
       qpidd-route [OPTIONS] link list <dest-broker>
```

```
qpid-route [OPTIONS] route add <dest-broker> <src-broker>
<exchange> <routing-key> [id] [exclude-list]
qpid-route [OPTIONS] route del <dest-broker> <src-broker>
<exchange> <routing-key>
qpid-route [OPTIONS] route list <dest-broker>
qpid-route [OPTIONS] route flush <dest-broker>
```

| Options for using <code>qpid-route</code> to Manage Federation | |
|--|--|
| <code>-v</code> | Verbose output. |
| <code>-q</code> | Quiet output, will not print duplicate warnings. |
| <code>-d</code> | Make the configuration change durable. |

Table 6.1. Options for using `qpid-route` to Manage Federation

User Authentication with MRG Messaging

MRG Messaging uses Simple Authentication and Security Layer (SASL) for identifying and authorizing incoming connections to the broker, as mandated in the AMQP specification. SASL provides a variety of authentication methods. While MRG Messaging clients primarily implement the “PLAIN” method, the broker uses the *Cyrus SASL library*¹ to allow for a full SASL implementation.

Enabling and Using SASL Plain Authentication

To use the default SASL PLAIN authentication mechanism implemented by the MRG Messaging client libraries, either use the default username and password of *guest*, which are included in the database at `/var/lib/qpidd/qpidd.sasl` on installation, or add your own accounts.

1. Add new users to the database by using the `saslpasswd2` command:

```
# saslpasswd2 -f /var/lib/qpidd/qpidd.sasl -u QPIDnew_user_name
```



Warning

Some Red Hat Enterprise MRG tools use the default guest account. If it is removed, those tools will fail to authenticate to the broker.

2. Existing user accounts can be listed by using the `-f` option:

```
# sasldblistusers2 -f /var/lib/qpidd/qpidd.sasl
```



Important

The user database at `/var/lib/qpidd/qpidd.sasl` is readable only by the `qpidd` user. If you start the broker from a user other than the `qpidd` user, you will

¹ <http://cyrusimap.web.cmu.edu/>

need to either modify the configuration file, or turn authentication off.

3. To switch authentication on or off, use the `auth yes|no` option when you start the broker:

```
# /usr/sbin/qpidd --auth yes  
# /usr/sbin/qpidd --auth no
```

You can also set authentication to be on or off by adding the appropriate line to to the `/etc/qpidd.conf` configuration file:

```
auth=no  
auth=yes
```

4. The SASL configuration file is in `/etc/sasl2/qpidd.conf` for Red Hat Enterprise Linux 5 and `/usr/lib/sasl2/qpidd.conf` for Red Hat Enterprise Linux 4.



Note

For information on using a different configuration, use your web browser to view the Cyrus SASL documentation at `/usr/share/doc/cyrus-sasl-lib-2.1.22/index.html` for Red Hat Enterprise Linux 5 or `/usr/share/doc/cyrus-sasl-2.1.19/index.html` for Red Hat Enterprise Linux 4.

More Information

Reporting a Bug.

If you have found a bug in MRG Messaging, follow these instructions to enter a bug report:

1. You will need a [Bugzilla](https://bugzilla.redhat.com/index.cgi)¹ account. You can create one at [Create Bugzilla Account](https://bugzilla.redhat.com/createaccount.cgi)².
2. Once you have a Bugzilla account, log in and click on [Enter A New Bug Report](https://bugzilla.redhat.com/enter_bug.cgi)³.
3. When submitting a bug report, you will need to identify the product (Red Hat Enterprise MRG), the version (1.0), and whether the bug occurs in the software (component = messaging) or in the documentation (component = Messaging_Installation_Guide).

Further Reading.

- Red Hat Enterprise MRG and MRG Messaging Product Information
 - <http://www.redhat.com/mrg>
- MRG Messaging Development Wiki
 - http://rhm.et.redhat.com/page/Main_Page
- For more information on the packages available and getting started with MRG Messaging:
 - http://rhm.et.redhat.com/page/Getting_Started_With_RHM
- MRG Messaging Users Mailing List
 - Subscribe by sending an email to `<rhm-users-request@redhat.com>` with the word “Subscribe” in the subject line.

¹ <https://bugzilla.redhat.com/index.cgi>

² <https://bugzilla.redhat.com/createaccount.cgi>

³ https://bugzilla.redhat.com/enter_bug.cgi

Appendix A. Revision History

Revision History

| | | |
|---|------------------|----------------|
| Revision 1.4 | 5 June, 2008 | Lana Brindley |
| Completed Revision for 1.0 Release | | |
| Revision 1.3 | 9 May, 2008 | Lana Brindley |
| Updated Broker Options and Configuration file information | | |
| Revision 1.2 | 4 February, 2008 | Lana Brindley |
| Updated Installation Instructions | | |
| Revision 1.1 | 16 October, 2007 | Jonathan Robie |
| Major Edit | | |
| Revision 1.0 | 12 October, 2007 | Lana Brindley |
| Preliminary Text, Untested | | |

