



# vFabric Hyperic 4.5

## Installation and Configuration Guide

## Table of Contents

1. vFabric Hyperic Installation Guide .....	3
1.1. Installation Requirements .....	4
1.2. Set Up Hyperic Database .....	9
1.3. Choose and Download an Installation Package .....	27
1.4. Installing Hyperic .....	29
1.5. Upgrade Hyperic Components .....	51
2. Configure and Run the Hyperic Agent .....	59
2.1. Understand Agent Environment and Operation .....	60
2.2. Configure Agent - Server Communication .....	67
2.3. Start, Stop, and Other Agent Operations .....	79
2.4. Configure Auto-Discovery Scanning and Reporting .....	83
2.5. Configure Agent Logging .....	84
2.6. Configure Plugin Loading .....	87
2.7. Deploying Multiple Hyperic Agents .....	88
2.8. Tweak the Agent to Enable a Resource Plugin .....	90
2.9. Manage the Hyperic Agent .....	99
2.10. Agent Properties .....	109
3. Configure and Run the Hyperic Server .....	125
3.1. Start and Stop Hyperic Server .....	126
3.2. Configure Hyperic Server Help and Announcement Behavior .....	127
3.3. Configure Metric Baselineing and Alert Processing Behavior .....	128
3.4. Managing the HQ Database .....	131
3.5. Scaling and Tuning Hyperic Performance .....	143
3.6. Integrate HQ Server with Other Systems .....	146
3.7. Clustering HQ Servers for Failover .....	155
3.8. Hyperic Server Properties .....	159
3.9. HQ Database Table Schemas .....	164
4. Troubleshoot Agent and Server Problems .....	174
4.1. Looking for Clues .....	175
4.2. Hyperic Server Problems .....	178
4.3. Agent Startup Problems .....	179
4.4. Invalid or Unknown Availability .....	181
4.5. Slow User Interface .....	183
4.6. Warning Messages in the Agent Log .....	184

# 1. vFabric Hyperic Installation Guide

These sections have instructions for setting up an external Hyperic database and installing the Hyperic Server and Hyperic Agents.

- [Section 1.1, “Installation Requirements”](#)
- [Section 1.2, “Set Up Hyperic Database”](#)
- [Section 1.3, “Choose and Download an Installation Package”](#)
- [Section 1.4, “Installing Hyperic”](#)
- [Section 1.5, “Upgrade Hyperic Components”](#)

## 1.1. Installation Requirements

Topics marked with\* relate to features available only in vFabric Hyperic.

- [Section 1.1.1, “Hyperic Server Requirements”](#)
  - [Hyperic Server JRE Requirements](#)
  - [Hyperic Server Resource Requirements](#)
  - [Hyperic Server Operating System Requirements](#)
  - [X Libraries on Unix-Based Platforms](#)
  - [Hyperic Server Supported Databases](#)
  - [Supported Browsers](#)
- [Section 1.1.2, “Hyperic Database Requirements”](#)
- [Section 1.1.3, “Hyperic Agent Requirements”](#)
  - [Agent System Resource Requirements](#)
  - [Agent Operating System Requirements](#)
  - [Agent JRE Requirements](#)
  - [Drivers for Monitoring Databases](#)

### 1.1.1. Hyperic Server Requirements

Hyperic supports only one Hyperic Server on a single host. The host must have a static IP address for server communications.

#### Hyperic Server JRE Requirements

Hyperic Server 4.5 requires a 1.6 JRE.

Use the no-JRE server installer if you prefer to use an existing JRE in your environment, or if no architecture-specific HQ Server package is available.

HQ Server can run with either a 32-bit or 64-bit JRE, except on Windows, as noted below.

##### Note for Windows Environments

HQ Server does **not** support 64-bit JREs under Windows. Use the 32-bit installer package for Windows, or the platform-independent installer if you prefer to use a pre-existing JRE in your environment.

If you do use a JRE of your own, see [HQ Components and JRE Location](#) for information about how to ensure that HQ Server uses the right JRE.

##### Note for Solaris Sparc Environment

There is *no* platform-specific HQ Server package for Solaris SPARC. Install a no-JRE HQ Server package in Solaris SPARC environments. For information about how to ensure that HQ Server can determine what JRE to use, see [HQ Components and JRE Location](#).

A platform-specific HQ Agent package with a bundled JRE is available for SPARC Solaris in a tarball.

**Note:** You **must** use GNU Tar to unpack HQ tarballs.

#### Hyperic Server Resource Requirements

For small to medium deployments, up to 50 managed platforms:

- 1 GHz or higher Pentium 4, or equivalent (2 x 2.4GHz Pentium Xeon or equivalent recommended)
- 1 GB RAM (4 or more GB recommended)
- 1-5 GB Free Disk Space

#### Hyperic Server Operating System Requirements

- Linux
- Solaris 10 or higher
- Mac OS X (Intel x86)
- Windows 2003 Server
- Windows 2008

- Windows Vista
- Windows 7

**Notes:**

Although Hyperic does not support Hyperic Server running under Windows XP in production environments, the configuration works; you can run small evaluation deployments under Windows XP.

For Unix-based platforms, the libXp.so.6 X library is required. See below.

## X Libraries on Unix-Based Platforms

On Unix-based platforms, Hyperic Server requires the libXp.so.6 X library to create charts and other graphics in the Hyperic user interface. The location of this library varies by version and provider:

- Enterprise Linux---As of Red Hat Enterprise Linux 4 and CentOS 4, libXp.so.6 is in the xorg-x11-deprecated-libs RPM.
- Debian---install the libxp6, libxt6, libxtst6, and libx11-6 packages.
- Fedora Core 5---as of Fedora Core 5, the libXp.so library has been split out to its own package; install the libXp RPM.
- Other distributions---The required libraries can be found in either the XFree86-libs or the xorg-x11-libs package.

## Hyperic Server Supported Databases

Hyperic is packaged with a built-in PostgreSQL V8.2.5 database, which is suitable for evaluation or very small Hyperic deployments.

Hyperic does not support deployments that use the built-in database for production deployments with more than 25 managed platforms.

For production deployments, Hyperic recommends running the Hyperic database MySQL or Oracle. For large deployments (100+ platforms) Hyperic requires the database to be located on a different host than the Hyperic Server and not be shared.

Hyperic supports the use of these databases for an external Hyperic database:

- MySQL
  - MySQL Enterprise Server and MySQL Community Server, v5.1.x (recommended)
  - MySQL Enterprise Server and MySQL Community Server, v5.0.45 or higher
  - Hyperic does not work with MySQL Versions other than 5.0.x or 5.1.x.
- Oracle:
  - 10g or 11g
- PostgreSQL
  - PostgreSQL 8.3
  - **Exception:**

- Hyperic does not support the use of PostgreSQL running under Windows in production deployments, and for production environments recommends the use of MySQL or Oracle.
- Hyperic does not support deployments that use the built-in database for production deployments with more than 25 managed platforms.

## Supported Browsers

Hyperic supports these browser versions:

- Firefox 2.x
- Firefox 3.x
- Internet Explorer 7
- Internet Explorer 8, except if running on Windows 2008

### Firefox Skype plugin causes problems

The Skype plugin for Firefox causes unexpected behavior in the Hyperic user interface. Disable the plugin to work around this problem.

Firefox is recommended.

## 1.1.2. Hyperic Database Requirements

- 1 GHz or higher Pentium 4, or equivalent (2 x 2.4GHz Pentium Xeon or equivalent recommended)
- 2 GB RAM (4 or more GB recommended)
- 25 GB Free Disk Space

## 1.1.3. Hyperic Agent Requirements

Hyperic supports only one Hyperic Agent on a host.

### Agent System Resource Requirements

- 500 MHz Celeron or higher, or equivalent
- 256 MB RAM
- 500 MB Free Disk Space

### Agent Operating System Requirements

- Linux
- Windows XP
- Windows XP Pro

- Windows 2000
- Windows 2003 Standard Edition
- Windows 2003 Enterprise Edition
- Windows 2008 Standard Edition
- Windows 7
- Solaris 10 or higher
- Mac OS X 10.5 or higher
- HP-UX 11.11 or higher
- AIX 5.3, and AIX 6.1 or higher
- FreeBSD 8.1

Host operating systems should employ a method of time sync (NTP). This is required in order to ensure accuracy of metric data reporting and alerts.

## Agent JRE Requirements

The Hyperic Agent can run with either a 1.5 or 1.6 JRE. Hyperic recommends a 1.5 JRE, which is included in platform-specific agent installers.

Use the platform-independent agent installer if you prefer to use an existing JRE in your environment.

## Drivers for Monitoring Databases

In the open source version of Hyperic, the plugins packaged with the Hyperic Agent for MSSQL, Oracle, Informix, DB2, and Sybase do not include the database vendor's JDBC plugin. After installing Hyperic HQ you must download and install the vendor-provided JDBC drivers for these plugins to work.

**Note:** The database plugins in vFabric Hyperic include the JDBC drivers.



## 1.2. Set Up Hyperic Database

This page has links to instructions for setting up an external Hyperic database. Follow the instructions for your database type.

- [Section 1.2.1, “Set Up MySQL”](#)
- [Section 1.2.2, “Set Up Oracle”](#)
- [Section 1.2.3, “Set Up PostgreSQL”](#)

## 1.2.1. Set Up MySQL

Topics marked with\* relate to features available only in vFabric Hyperic.

- [Set Up Hyperic Database on MySQL](#)
  - [Step 1: Create a MySQL Database Instance](#)
  - [Step 2: Configure MySQL Startup Options and System Variables](#)
  - [Step 3 :Install the Hyperic Server](#)
  - [Step 4: Tune the Batch Aggregate Inserter for MySQL](#)
- [Solve Problems with MySQL Configuration](#)
- [MySQL Maintenance Examples](#)

### Set Up Hyperic Database on MySQL

This section provides instructions for setting up MySQL as your external Hyperic database. It is assumed that you have already installed MySQL and are either familiar with MySQL or have the support of someone who is.

**Note:** If you are installing Hyperic for evaluation, you can use Hyperic's built-in PostgreSQL database, rather than set up an external database.

If you are new to MySQL, the introduction to MySQL at [http://dev.mysql.com/tech-resources/articles/mysql\\_intro.html](http://dev.mysql.com/tech-resources/articles/mysql_intro.html) may be of interest.

#### Step 1: Create a MySQL Database Instance

Run these commands at the mysql prompt, as the root user:

```
mysql> create user 'hqadmin'@'<hq_server_host>' identified by '<passwd>';
mysql> create database HQ CHARACTER SET utf8 COLLATE utf8_bin;
mysql> grant all on HQ.* to 'hqadmin'@'<hq_server_host>';
```

UTF8 is required for encoding.

#### Step 2: Configure MySQL Startup Options and System Variables

In this step, you configure the MySQL database, by editing the settings in its configuration file. In Unix and Linux, the file is /etc/my.cnf. In Windows the file is my.ini, located in the MySQL installation base directory.

For more information about InnoDB startup options and system variables, see <http://dev.mysql.com/doc/ref-man/5.0/en/innodb-parameters.html>

1. Enable the full query log. Every query (even ones with incorrect syntax) that the database server receives will be logged. This is useful for debugging, but it is usually disabled in production use. Be sure to change the paths given here to match your environment.

```
[mysqld]
log-error = /home/mysql/log/mysqld.err
log = /home/mysql/log/mysql_general.log
```

2. Print warnings to the error log file. If you have any problem with MySQL, you should enable logging of warnings and examine the error log for possible explanations.

```
log_warnings  
server-id      = 1
```

3. Configure buffer pool size. The size of the MySQL buffer pool is has a significant impact on MySQL performance. If your database is on a dedicated machine, make the buffer pool about 80% of total memory.

```
innodb_buffer_pool_size = 256M
```

4. Configure the frequency with which the log buffer is written to the log, and the the point at which the log is flushed to the disk. Setting `innodb_flush_log_at_trx_commit` to 0 dramatically increases MySQL performance, but with this setting, you are likely to lose data in the event of a server crash. If loss of data is unacceptable, set `innodb_flush_log_at_trx_commit` to 2. Hyperic does not recommend setting the value to 1.

```
innodb_flush_log_at_trx_commit = 2  
innodb_log_buffer_size = 64M  
innodb_log_file_size = 256M
```

5. Configure innodb as the default storage engine. (Required.)

```
default-storage_engine=innodb  
bulk_insert_buffer_size = 32M  
join_buffer_size = 8M  
max_heap_table_size = 256M  
tmp_table_size = 256M  
max_tmp_tables = 48  
myisam_sort_buffer_size = 256M
```

6. Configure the sort buffer size. MySQL recommends a `sort_buffer_size` larger than the one suggested her.

```
sort_buffer_size = 64K
```

An article on experimenting with sort buffer size is available at <http://www.mysqlperformanceblog.com/2007/08/18/how-fast-can-you-sort-data-with-mysql>.

7. Configure the read buffer size. Because Hyperic does a significant volume of sequential reads, a large read buffer improve performance.

```
read_buffer_size = 1M  
read_rnd_buffer_size = 10M  
table_cache = 2048  
set-variable = max_connections=400  
key_buffer_size = 256M  
thread_cache_size = 32
```

8. Configure the number of threads that can run in the InnoDB kernel. A starting point for setting this value is to to set a value equal to 2 times the number of CPUs times the number of disks.

```
innodb_thread_concurrency = 8
```

9. Set the method that is used to flush data and log files.\*= For battery-backed-up storage with write-back cache mode on Linux OSs, the `O_DIRECT` flush method is good. Learn about [other innodb flush methods](#).

```
innodb_flush_method=O_DIRECT  
innodb_rollback_on_timeout=1
```

- In this situation, tune your Linux OS (version 2.6 or higher) to favor the use of main memory rather than file caches with either:

```
# sysctl -w vm.swappiness=30
```

- or

```
# echo 30 >/proc/sys/vm/swappiness
```

10. Set query cache size. Generally, the higher this value, the better the performance. However, in MySQL versions older than 5.0.50, beware of setting this variable too high, as it may cause the database to pause. For more information, see the bug description at <http://bugs.mysql.com/bug.php?id=21074>.

```
query_cache_size = 0
```

11. Set query cache limit. The default value here is 1M. If the `qcache_hits-to-qcache_inserts` ratio is low, raise this value.

```
query_cache_limit = 8M
```

12. Set character encoding. Hyperic requires a char encoding of utf-8\*.

```
default-character-set=utf8
collation_server=utf8_bin
```

### Step 3 :Install the Hyperic Server

For instructions, see [Section 1.4, “Installing Hyperic”](#).

### Step 4: Tune the Batch Aggregate Inserter for MySQL

**NOTE:** Perform these steps only after installing the Hyperic Server.

These tuning recommendations are based on a performance tuning exercise in an environment with 700 Hyperic Agents reporting to an Hyperic Server on an 8 way / 16 GB host with an MySQL database running on an 8 way / 8 GB host, each running CentOS 5, with

- Workers: 4
- QueueSize: 4000000
- BatchSize: 2000

With 7 hours of backfilled data the server peaked out at 2.2 million rows inserted.

This intent of the strategy was to keep the Batch Aggregate Inserter (BAI) on "cruise control", instead throwing threads at the queued metrics all at once and causing CPU spikes.

It was found that the BAI workers had no trouble keeping up with the "normal" incoming load, and in a catchup scenario (after backfilling) the high Queue Size allowed them plenty of time to catch up.

For a smaller deployment, consider only tweaking the number of workers down to 1 or 2. This will ease random CPU spikes and MySQL should have no problem keeping up with the incoming traffic.

Please NOTE these settings may not be applicable to PostgreSQL and Oracle since MySQL handles catchup scenarios much more gracefully.

To update the Batch Aggregate Inserter settings for MySQL run these commands at the mysql prompt as the hqadmin user:

```
mysql> update HQ.EAM_CONFIG_PROPS set propvalue = 4 where propkey =
'BATCH_AGGREGATE_WORKERS';
mysql> update HQ.EAM_CONFIG_PROPS set propvalue = 2000 where propkey =
'BATCH_AGGREGATE_BATCHSIZE';
mysql> update HQ.EAM_CONFIG_PROPS set propvalue = 4000000 where propkey =
'BATCH_AGGREGATE_QUEUE';
```

## Solve Problems with MySQL Configuration

If MySQL fails to start and issues a message similar to this:

```
InnoDB: Error: log file ./ib_logfile0 is of different size 0 5242880 bytes
InnoDB: than specified in the .cnf file 0 268435456 bytes!
080403 8:06:13 ERROR Default storage engine (InnoDB) is not available
080403 8:06:13 ERROR Aborting
```

the actual log size does not match the configured log size.

Delete the log files in /var/lib/mysql/ and restart MySQL.

## MySQL Maintenance Examples

Here are examples of regular maintenance for mysql

### 1. Simple MySQL Backup Script

```
#!/bin/sh

START=`date '+%A %Y/%m/%d %H:%M:%S'`
DAY=`date +%A`
MYSQLADMIN="/usr/bin/mysqladmin"
MYSQLDUMP="/usr/bin/mysqldump"
USER="root"
PASSWORD="mysql"
DBNAME="hqdb"
DEST="/home/mysql/dumps/$DBNAME-$DAY.sql.gz"
flushCmd="$MYSQLADMIN -u $USER -p$PASSWORD flush-logs"
dumpCmd="$MYSQLDUMP -u $USER -p$PASSWORD --quick --single-transaction $DBNAME"
gzip="gzip"
echo "Starting backup: $START"
echo "$flushCmd && $dumpCmd | $gzip > $DEST"
$flushCmd && $dumpCmd | $gzip > $DEST
END=`date '+%A %Y/%m/%d %H:%M:%S'`
echo "Backup completed: $END"
```

### 2. Simple Log Rollover Scheme. This may be done with error files, log files, etc.

```
cp /path/to/mysql/log/mysqld.err /path/to/mysql/log/mysqld-`date '+%w'`.err ;
cp /dev/null /path/to/mysql/log/mysqld.err
```

### 3. Sample Unix Cron Entries (empty lines will fail in cron, beware)

```
#
#      Field 1: (0-59) minute
#      Field 2: (0-23) hour
#      Field 3: (1-31) day of the month
```

```
#      Field 4: (1-12) month of the year
#      Field 5: (0-6)  day of the week - 1=Monday
# -----
#
0 2 * * * backup.sh
0 1 * * * cp /path/to/mysql/log/mysql.err /path/to/mysql/log/mysql-`date '+%w'`.err ;
cp /dev/null /path/to/mysql/log/mysql.err
```

## 1.2.2. Set Up Oracle

Topics marked with\* relate to features available only in vFabric Hyperic.

This section has instructions for configuring an Oracle database for an HQ installation.

- [Before Installing HQ Server](#)
  - [Installation Requirements](#)
  - [Create an Oracle instance](#)
  - [Create Tablespaces](#)
  - [Create the Database User for HQ](#)
  - [Grant Permissions to the Database User](#)
  - [Obtain Oracle JDBC Driver \(Hyperic HQ only\)](#)
- [Install HQ Server](#)
- [Enable Row Movement for Oracle 10g and 11g](#)
- [Periodic Oracle 10g and 11g Database Maintenance](#)
- [Tuning HQ Database on Oracle for Medium to Large Environments](#)
  - [Create the TS\\_HQDB\\_16K Tablespace](#)
  - [Configure REDO Logs](#)
  - [Configure Initialization Parameters for Oracle](#)
  - [Configure Batch Aggregate Inserter](#)
  - [Move Database Tables](#)
  - [Rebuild Indexes for Moved Tables](#)
  - [Configure Tables for High Concurrency](#)
  - [Restart HQ Server](#)

## Before Installing HQ Server

Before installing HQ Server, you create the HQ database in Oracle, configure the database, and create the database user that HQ will use to access the database. The sections that follow have instructions.

### Installation Requirements

Hyperic supports:

- Oracle 10g 10.2.0.1 or higher patch level. Preferred version is 10.2.0.4
- Oracle 11g

Hyperic recommends that the Oracle server for the Hyperic database run a dedicated system, with at least 8 GB RAM allocated to SGA.

### Create an Oracle instance

Install Oracle on the machine to be used, and create a database.

The database can be created with Oracle Database Configuration Assistant.

Select **New Database** (Includes datafiles = No). To save space, decline to install the Example Schemas.

Because you are running Oracle on a dedicated host, you can select the "Typical Memory" configuration.

Select "OLTP" as the type of database sizing to use. Allocate as high a percentage of system resources as you can afford: 70-90%, ideally in the higher range. See [Configure Initialization Parameters for Oracle](#) for SGA and PGA size.

### Create Tablespaces

1. Create the TEMP\_HQDB temporary tablespace, 2 GB in size
2. Create the TS\_HQDB tablespace, 25 GB in size. This tablespace will be used to store HQ\_METRIC\_DATA\_\*D\_\*S tables.

### Create the Database User for HQ

In this step you create the database user account that HQ will use to access the Oracle database.

There are multiple methods for creating a user in Oracle. To do it using SQL\*Plus, log into the Oracle instance as the system user with SQL\*Plus, and issue the create user command:

```
SQL> CREATE USER _HUSER_ IDENTIFIED BY _HQPASSWORD_ DEFAULT TABLESPACE TS_HQDB;
```

replacing *HQUSER* and *HQPASSWORD* with desired values.

### Grant Permissions to the Database User

You can grant the database user the necessary permissions in SQL\*Plus with the grant command:

```
SQL> GRANT CONNECT, RESOURCE, CREATE VIEW TO HUSER;
```

Verify the permission setting:



```
SQL> SELECT GRANTED_ROLE, DEFAULT_ROLE FROM dba_role_privs WHERE grantee = 'HQUUSER';
```

Make sure that you see the following rows for CONNECT and RESOURCE roles:

```
GRANTED_ROLE DEFAULT_ROLE
CONNECT YES
RESOURCE YES
```

If that is not the case, update the permissions:

```
ALTER USER HQUUSER DEFAULT ROLE RESOURCE, CONNECT;
```

### Obtain Oracle JDBC Driver (Hyperic HQ only)

The HQ Server in Hyperic HQ does not include the Oracle JDBC driver. Perform these steps before installing HQ Server.

To obtain the driver:

1. Go to [http://www.oracle.com/technology/software/tech/java/sqlj\\_jdbc/index.html](http://www.oracle.com/technology/software/tech/java/sqlj_jdbc/index.html)
2. In the **JDBC Driver Downloads** section, select the link for your version of Oracle.
3. On the drivers download page, select the driver for your JDK version.
  - JDK 5.0 — ojdbc5.jar
  - JDK 6.0 — ojdbc6.jar
  - **Note:** JDeveloper 11g JDBC drivers support JDBC 4.0 features. Use ojdbc6.jar with JDK 6.0 for JDBC 4.0 features.
4. On the page that appears accept the licensing agreement.
5. Download the jar file.
6. Copy the driver jar file to `hyperic-hq-installer/installer-4.x.y/lib`.
7. Proceed with the HQ installation process.

### Install HQ Server

Install HQ Server. See [Section 1.4, “Installing Hyperic”](#) for instructions.

**Note:** The HQ database must be up for the HQ Server installation process to succeed.

### Enable Row Movement for Oracle 10g and 11g

After installing HQ Server, run the [enable\\_row\\_movement.sql](#) procedure attached to this page - this is necessary to enable the routine maintenance described in [Periodic Oracle 10g and 11g Database Maintenance](#). It is only necessary to enable row movement once; you do not need to do it on a recurring basis.

### Periodic Oracle 10g and 11g Database Maintenance

Perform the following maintenance after running Hyperic for about a week, and monthly thereafter:

1. Run the [table\\_maintenance.sql](#) procedure attached to this page to compact and shrink space on all HQ tables (except HQ\_METRIC\* tables).
2. Run the [rebuild\\_index.sql](#) procedure attached to this page to rebuild the index on all HQ tables.
3. Run tablespace maintenance as appropriate:
  - a. Perform the following query to determine how fragmented a tablespace is. The query returns the tablespace name and total count of holes in it.

```
select count(*), tablespace_name
from dba_free_space
group by tablespace_name
order by 1,2;
```

- b. If the hole count is in the order of thousands, run **Reorganize**.

ORACLE Enterprise Manager 11g Database Control

Database Instance: orcl.intranet.hyperic.net > Tablespaces > View Tablespace: TS\_HQDB

Actions: Reorganize Go Edit Return

Name: TS\_HQDB  
 Bigfile tablespace: No  
 Status: ReadWrite  
 Type: Permanent  
 Extent Management: Local  
 Encryption: NO

**Storage**  
 Allocation Type: Automatic  
 Segment Space Management: Automatic  
 Enable logging: Yes  
 Compression: Disabled  
 Block Size (B): 8192

Name	Directory	Size (MB)	Used (MB)
ts_hqdb_reorg0	/array/oracle/oradata/orcl/	32,767.98	571.11
ts_hqdb2_reorg0	/array/oracle/oradata/orcl/	25,600.00	518.38

**Tablespace Full Metric Thresholds**

Space Used (%)	Free Space (MB)
This tablespace is using the database default space used thresholds.	This tablespace is using the database default free space thresholds.
Warning (%) 85	Warning (MB) Not Defined
Critical (%) 97	Critical (MB) Not Defined

Actions: Reorganize Go Edit Return

Copyright © 1996, 2007, Oracle. All rights reserved.  
 Oracle, JD Edwards, PeopleSoft, and Retek are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.  
[About Oracle Enterprise Manager](#)

## Tuning HQ Database on Oracle for Medium to Large Environments

If you manage more than 100 platforms, follow the steps in this section to tune your Oracle-hosted HQ database.

### Create the TS\_HQDB\_16K Tablespace

Create the TS\_HQDB\_16K tablespace, 25 GB in size, with 16 K blocksize. Here is an example of the syntax to create a table space with 16 K blocksize, where 'datafile\_name.dbf' is a file where the tablespace data will physically reside:

```
CREATE TABLESPACE TS_HQDB_16K
datafile 'datafile_name.dbf' SIZE 25000M AUTOEXTEND OFF
ONLINE
PERMANENT
EXTENT MANAGEMENT LOCAL UNIFORM SIZE 128K
```

```
BLOCKSIZE 16K
SEGMENT SPACE MANAGEMENT AUTO
;
```

## Configure REDO Logs

Redo logs are transaction journals. Each transaction is recorded in the redo logs. Redo logs are used in a serial fashion with each transaction queuing up in the redo log buffers and being written one at a time into the redo logs.

Configure at least three REDO logs, each 2048 MB in size.

Locate REDO logs on a separate disk spindle from datafiles.

Do not locate REDO logs on a RAID array.

## Configure Initialization Parameters for Oracle

Set following Oracle initialization parameters:

```
DB_WRITER_PROCESS = 4
SGA_MAX = 8G
SGA_TARGET = 7G
SHARED_POOL_SIZE=700M
PGA_AGGREGATE_TARGET=1500M
DB_16K_CACHE_SIZE=1000M
DB_KEEP_CACHE_SIZE = 500M
FILESYSTEMIO_OPTIONS=SetAll
DB_FILE_MULTIBLOCK_READ_COUNT=16
OPEN_CURSORS=300
PROCESSES=500
```

## Configure Batch Aggregate Inserter

1. Make sure that the HQ Server is shut down.
2. Log into the Oracle instance as HQ user with SQL\*Plus, and issue these commands to increase the data aggregate inserter batch size to 8000 and number of workers to 10:

```
UPDATE EAM_CONFIG_PROPS SET PROPVALUE=8000 where PROPKEY='BATCH_AGGREGATE_BATCHSIZE';
UPDATE EAM_CONFIG_PROPS SET PROPVALUE=10 where PROPKEY='BATCH_AGGREGATE_WORKERS';
UPDATE EAM_CONFIG_PROPS SET PROPVALUE=5000000 where
PROPKEY='BATCH_AGGREGATE_QUEUE';
```

## Move Database Tables

While still logged into the Oracle instance as the system user, run these commands to move metric and measurement tables to the TS\_HQDB\_16K tablespace:

```
alter table HQADMIN.HQ_METRIC_DATA_0D_0S move tablespace TS_HQDB_16K;
alter table HQADMIN.HQ_METRIC_DATA_0D_1S move tablespace TS_HQDB_16K;
alter table HQADMIN.HQ_METRIC_DATA_1D_0S move tablespace TS_HQDB_16K;
alter table HQADMIN.HQ_METRIC_DATA_1D_1S move tablespace TS_HQDB_16K;
alter table HQADMIN.HQ_METRIC_DATA_2D_0S move tablespace TS_HQDB_16K;
alter table HQADMIN.HQ_METRIC_DATA_2D_1S move tablespace TS_HQDB_16K;
alter table HQADMIN.HQ_METRIC_DATA_3D_0S move tablespace TS_HQDB_16K;
alter table HQADMIN.HQ_METRIC_DATA_3D_1S move tablespace TS_HQDB_16K;
alter table HQADMIN.HQ_METRIC_DATA_4D_0S move tablespace TS_HQDB_16K;
alter table HQADMIN.HQ_METRIC_DATA_4D_1S move tablespace TS_HQDB_16K;
alter table HQADMIN.HQ_METRIC_DATA_5D_0S move tablespace TS_HQDB_16K;
```

```

alter table HQADMIN.HQ_METRIC_DATA_5D_1S move tablespace TS_HQDB_16K;
alter table HQADMIN.HQ_METRIC_DATA_6D_0S move tablespace TS_HQDB_16K;
alter table HQADMIN.HQ_METRIC_DATA_6D_1S move tablespace TS_HQDB_16K;
alter table HQADMIN.HQ_METRIC_DATA_7D_0S move tablespace TS_HQDB_16K;
alter table HQADMIN.HQ_METRIC_DATA_7D_1S move tablespace TS_HQDB_16K;
alter table HQADMIN.HQ_METRIC_DATA_8D_0S move tablespace TS_HQDB_16K;
alter table HQADMIN.HQ_METRIC_DATA_8D_1S move tablespace TS_HQDB_16K;
alter table HQADMIN.EAM_MEASUREMENT_DATA_1D move tablespace TS_HQDB_16K;
alter table HQADMIN.EAM_MEASUREMENT_DATA_1H move tablespace TS_HQDB_16K;
alter table HQADMIN.EAM_MEASUREMENT_DATA_6H move tablespace TS_HQDB_16K;

```

## Rebuild Indexes for Moved Tables

Run these commands to rebuild the indexes for the moved tables:

```

alter index HQADMIN.METRIC_DATA_0D_0S_MID_IDX rebuild;
alter index HQADMIN.METRIC_DATA_0D_1S_MID_IDX rebuild;
alter index HQADMIN.METRIC_DATA_1D_0S_MID_IDX rebuild;
alter index HQADMIN.METRIC_DATA_1D_1S_MID_IDX rebuild;
alter index HQADMIN.METRIC_DATA_2D_0S_MID_IDX rebuild;
alter index HQADMIN.METRIC_DATA_2D_1S_MID_IDX rebuild;
alter index HQADMIN.METRIC_DATA_3D_0S_MID_IDX rebuild;
alter index HQADMIN.METRIC_DATA_3D_1S_MID_IDX rebuild;
alter index HQADMIN.METRIC_DATA_4D_0S_MID_IDX rebuild;
alter index HQADMIN.METRIC_DATA_4D_1S_MID_IDX rebuild;
alter index HQADMIN.METRIC_DATA_5D_0S_MID_IDX rebuild;
alter index HQADMIN.METRIC_DATA_5D_1S_MID_IDX rebuild;
alter index HQADMIN.METRIC_DATA_6D_0S_MID_IDX rebuild;
alter index HQADMIN.METRIC_DATA_6D_1S_MID_IDX rebuild;
alter index HQADMIN.METRIC_DATA_7D_0S_MID_IDX rebuild;
alter index HQADMIN.METRIC_DATA_7D_1S_MID_IDX rebuild;
alter index HQADMIN.METRIC_DATA_8D_0S_MID_IDX rebuild;
alter index HQADMIN.METRIC_DATA_8D_1S_MID_IDX rebuild;
alter index HQADMIN.MEASUREMENT_DATA_1H_MID_IDX rebuild;
alter index HQADMIN.MEASUREMENT_DATA_6H_MID_IDX rebuild;
alter index HQADMIN.MEASUREMENT_DATA_1D_MID_IDX rebuild;

```

## Configure Tables for High Concurrency

Run these commands for any HQ 4.x version:

```

alter table <schema>.HQ_METRIC_DATA_0D_0S initrans 15;
alter index <schema>.METRIC_DATA_0D_0S_MID_IDX initrans 15;
alter table <schema>.HQ_METRIC_DATA_0D_1S initrans 15;
alter index <schema>.METRIC_DATA_0D_1S_MID_IDX initrans 15;
alter table <schema>.HQ_METRIC_DATA_1D_0S initrans 15;
alter index <schema>.METRIC_DATA_1D_0S_MID_IDX initrans 15;
alter table <schema>.HQ_METRIC_DATA_1D_1S initrans 15;
alter index <schema>.METRIC_DATA_1D_1S_MID_IDX initrans 15;
alter table <schema>.HQ_METRIC_DATA_2D_0S initrans 15;
alter index <schema>.METRIC_DATA_2D_0S_MID_IDX initrans 15;
alter table <schema>.HQ_METRIC_DATA_2D_1S initrans 15;
alter index <schema>.METRIC_DATA_2D_1S_MID_IDX initrans 15;
alter table <schema>.HQ_METRIC_DATA_3D_0S initrans 15;
alter index <schema>.METRIC_DATA_3D_0S_MID_IDX initrans 15;
alter table <schema>.HQ_METRIC_DATA_3D_1S initrans 15;
alter index <schema>.METRIC_DATA_3D_1S_MID_IDX initrans 15;
alter table <schema>.HQ_METRIC_DATA_4D_0S initrans 15;
alter index <schema>.METRIC_DATA_4D_0S_MID_IDX initrans 15;
alter table <schema>.HQ_METRIC_DATA_4D_1S initrans 15;
alter index <schema>.METRIC_DATA_4D_1S_MID_IDX initrans 15;
alter table <schema>.HQ_METRIC_DATA_5D_0S initrans 15;
alter index <schema>.METRIC_DATA_5D_0S_MID_IDX initrans 15;

```

```
alter table <schema>.HQ_METRIC_DATA_5D_1S initrans 15;  
alter index <schema>.METRIC_DATA_5D_1S_MID_IDX initrans 15;  
alter table <schema>.HQ_METRIC_DATA_6D_0S initrans 15;  
alter index <schema>.METRIC_DATA_6D_0S_MID_IDX initrans 15;  
alter table <schema>.HQ_METRIC_DATA_6D_1S initrans 15;  
alter index <schema>.METRIC_DATA_6D_1S_MID_IDX initrans 15;  
alter table <schema>.HQ_METRIC_DATA_7D_0S initrans 15;  
alter index <schema>.METRIC_DATA_7D_0S_MID_IDX initrans 15;  
alter table <schema>.HQ_METRIC_DATA_7D_1S initrans 15;  
alter index <schema>.METRIC_DATA_7D_1S_MID_IDX initrans 15;  
alter table <schema>.HQ_METRIC_DATA_8D_0S initrans 15;  
alter index <schema>.METRIC_DATA_8D_0S_MID_IDX initrans 15;  
alter table <schema>.HQ_METRIC_DATA_8D_1S initrans 15;  
alter index <schema>.METRIC_DATA_8D_1S_MID_IDX initrans 15;
```

## Restart HQ Server

Restart the HQ Server.

### 1.2.3. Set Up PostgreSQL

- [Set Up External Hyperic Database on PostgreSQL](#)
- [Example Configuration](#)
- [Install and Initialize PostgreSQL](#)
- [Create PostgreSQL User](#)
- [Configure PostgreSQL Server Options](#)
- [PostgreSQL Configuration Tips for Large HQ Deployments](#)
- [Configure Client Authentication](#)
- [Install PostgreSQL Client on the HQ Server](#)
- [Install the HQ Server](#)
- [Create PLPG Language for PostgreSQL 8 Installations](#)
- [Start HQ Server](#)
- [Troubleshooting Database Connection Problems](#)
- [Useful PostgreSQL Commands](#)

## Set Up External Hyperic Database on PostgreSQL

This section provides instructions for setting up PostgreSQL as your external Hyperic database.

**Note:** If you are installing Hyperic for evaluation, you can use Hyperic's built-in PostgreSQL database, rather than set up an external database.

Hyperic recommends that this setup be performed by or with the support of your database administrator.

These instructions assume that you are performing a new installation of an RPM package of PostgreSQL, using Yum, an automatic RPM package installer.

If you do not have an RPM package, you can obtain the source from the [PostgreSQL website](https://www.postgresql.org/).

### Example Configuration

The instructions that follow show commands for setting up PostgreSQL in an environment with these characteristics:

<b>Operating system</b>	Red Hat Enterprise Linux 4
<b>Database</b>	PostgreSQL 8.3
<b>Database IP address</b>	192.168.1.4
<b>HQ Server IP address</b>	192.168.1.6
<b>Database user name</b>	admin
<b>Database password</b>	hqadmin
<b>Database location</b>	/var/lib/pgsql/data/
<b>PostgreSQL configuration file</b>	/var/lib/pgsql/data/postgresql.conf
<b>PostgreSQL authorization file</b>	/var/lib/pgsql/data/pg_hba.conf

### Install and Initialize PostgreSQL

Log in as root to the target Linux machine, and enter this command:

```
yum install postgresql postgresql-server
```

PostgreSQL will be installed in:

```
/etc/init.d/  
/usr/bin/  
/usr/share/doc/  
/var/lib/pgsql/
```

### Create PostgreSQL User

1. Change user to PostgreSQL and connect to the database locally.

```
# su postgres
```

- The psql prompt is displayed.

2. Create a user named admin with login and createdb privileges.

```
create role admin with login createdb password 'hqadmin';
```

3. Create a default database for Hyperic. Place quotes around the string HQ so that the database name will be uppercase.

```
CREATE DATABASE "HQ" OWNER admin;
```

## Configure PostgreSQL Server Options

In this step, you configure PostgreSQL Server options in the PostgreSQL server configuration file, postgresql.conf.

1. Open the postgresql.conf file.
2. The default database permissions allow local connections only. To configure PostgreSQL to listen on all network interfaces, uncomment the listen address entry and change its value as shown below.

```
listen_addresses = '*'
```

3. Add the following settings to optimize HQ performance:

```
##performance changes for HQ
shared_buffers=10000
work_mem=2048
statement_timeout=30000
```

**Note:** In PostgreSQL 8.3, the PostgreSQL parameter that enables HQ to monitor the database - track\_counts - is enabled, because the PostgreSQL autovacuum daemon needs the collected information. Only superusers can change this setting.

## PostgreSQL Configuration Tips for Large HQ Deployments

These changes and additions to postgresql.conf can improve HQ performance in large environments, assuming that you have at least 2GB RAM available for the database.

```
shared_buffers = 20000
commit_delay = 10000
checkpoint_segments = 15
work_mem = 8192
maintenance_work_mem = 32768
max_fsm_pages = 40000
effective_cache_size = 5000
```

In particular, increasing effective\_cache\_size is beneficial, given sufficient RAM.

You may need to refine your database configuration based on the performance you experience. Review of the database log by a database administrator should indicate whether further adjustments to checkpoint\_segments or max\_fsm\_pages are appropriate.

## Configure Client Authentication

In this step, you configure PostgreSQL to allow connections from other users and from the HQ Server.

PostgreSQL client authentication is defined in the pg\_hba.conf file, which contains lines, referred to as records, that specify allowed connection types, users, client IP addresses, and authentication method. Locate this line in the file:

```
# TYPE DATABASE USER CIDR-ADDRESS AUTH-METHOD
```



and add these lines below it:

```
local    all        all        ident          sameuser
host     all        all        *192.168.1.6/32* password
```

For more information about pg\_hba.conf see [\[\[http://www.postgresql.org/docs/8.2/interactive/auth-pg-hba-conf.html](http://www.postgresql.org/docs/8.2/interactive/auth-pg-hba-conf.html)

## Install PostgreSQL Client on the HQ Server

The purpose of installing the PostgreSQL client is so that you can verify connectivity between the server and the database. To install the client, enter these commands:

```
yum -y install postgresql
psql -d postgres -h 192.168.1.4 -U admin -W
```

Once the connectivity is tested, you can remove the PostgreSQL client with this command:

```
yum -y remove postgresql
```

## Install the HQ Server

Install the HQ Server. For instructions, see [Section 1.4, “Installing Hyperic”](#).

Note: Do not start the HQ Server until after completing the steps in the following section.

## Create PLPG Language for PostgreSQL 8 Installations

**Note:** This step is necessary only for v8.0 of PostgreSQL.

HQ usually automatically creates a language in the PostgreSQL database. HQ is not able to create the language automatically in PostgreSQL 8.0, so for that version, you must run the following command on the HQ database before starting the HQ Server:

```
createlang plpgsql [ICG:DATABASE NAME]
```

For example:

```
createlang plpgsql HQ
```

The createlang executable is located in the bin directory of your PostgreSQL installation.

## Start HQ Server

Start the HQ Server. For instructions, see [Section 3, “Configure and Run the Hyperic Server”](#). If the server fails to start up, there may be problems with your PostgreSQL configuration. Check the PostgreSQL logs for connection failures or errors.

## Troubleshooting Database Connection Problems

If network connections to the database fail, you can troubleshoot the issue in PostgreSQL log files, using the UNIX® tail command with the -f parameter

`tail -f` displays the lines at the end of a file, and displays additional log messages that follow to the terminal. This is useful for watching log files, or any other file which may be appended over time. The following log files will have information about failed connections.

- `/var/lib/pgsql/data/pg_log/postgresql-day.log`
- `/var/lib/pgsql/pgstartup.log`

## Useful PostgreSQL Commands

```
\h      help   with SQL commands
?       help   with psql commands
\du     list   roles/users
\l      list   databases
\c      to choose a database
\d      to list tables once in a database
\q      quit
```

## 1.3. Choose and Download an Installation Package

Hyperic HQ and Hyperic Enterprise are available for download at <http://www.hyperic.com/downloads.html>.

This section describes the different types of installation packages.

- [Section 1.3.1, “Full Installation vs. Agent-Only Packages”](#)
- [Section 1.3.2, “Platform-Specific Versus Platform-Independent”](#)
- [Section 1.3.3, “Package Formats”](#)

### 1.3.1. Full Installation vs. Agent-Only Packages

The "HQ Server Package" contains the HQ Server, including a built-in database you can use for evaluation purposes, and the HQ Agent. You can use this package to install the server, the agent, or both.

The "HQ Agent Package" contains only the agent.

### 1.3.2. Platform-Specific Versus Platform-Independent

Operating system-specific packages and platform-independent packages are available.

Platform-specific installers include a JRE, and the server package includes a built-in PostgreSQL database, suitable for use in evaluations.

Platform-specific HQ Agent installers include a 1.5 JRE, and platform-specific HQ Server installers include a 1.6 JRE.

Platform-independent installers do not include a JRE, or, in the case of the HQ Server installer, the built-in PostgreSQL database.

#### Note for Solaris Sparc Environment

There is *no* platform-specific HQ Server package for Solaris SPARC. Install a no-JRE HQ Server package in Solaris SPARC environments. For information about how to ensure that HQ Server can determine what JRE to use, see [HQ Components and JRE Location](#).

A platform-specific HQ Agent package with a bundled JRE is available for SPARC Solaris in a tarball.

**Note:** You **must** use GNU Tar to unpack HQ tarballs.

Select a platform-independent installer if you are installing components on a platform for which there is no platform-specific installer, or you want to use a JRE that is already installed on the host.

### 1.3.3. Package Formats

HQ installation packages are provided in these formats:

- Tarball archive - Tarball packages are available for Linux-like systems. HQ tarballs are archived with GNU Tar and must be unpacked to GNU Tar.

```
tar zxvf hyperic-hq-installer-4.x.y-xxx.tgz
```

**Unpack Tarballs with GNU Tar Only**

Use GNU Tar to unpacking HQ tarballs. Use of proprietary Unix Tar utilities will result in warnings. GNU Tar is available at <http://www.gnu.org>

- Windows MSI package - The full installer is available as an MSI package.
- Zip archive - The full installer and an agent-only distribution are available in .zip form for use on Windows platforms that do not have a tar-compatible utility. Platform-independent packages are also available in .zip format. These packages are archived with GNU Zip. Hyperic recommends the use of GNU Zip to unpack zip archives.
- RPM package - Agent-only and server-only RPM packages are available for RedHat Linux. RPM is a method of distributing software that makes it easy to install, upgrade, query and delete. RPM files contain information on the package's name, version, other file dependency information (if applicable), platform information (such as Intel or Alpha, etc.), as well as default file install locations.

Note: The RPM package for the HQ Agent does not include a JRE.

## 1.4. Installing Hyperic

This section has instructions for installing the Hyperic Server and the Hyperic Agent. See the tips in [Section 1.4.1, “Before You Start”](#), then refer to the instructions for the package you are using.

**Note:** Unless you are using the internal Hyperic database, you must create a database for Hyperic before installing the Hyperic Server.

- [Section 1.4.1, “Before You Start”](#)
- [Section 1.4.2, “Installing the Agent and Server from a Tarball or Zip Archive”](#)
- [Section 1.4.3, “Installing an Agent-Only Package”](#)
- [Section 1.4.4, “Installing HQ Using the Windows MSI Installer”](#)
- [Section 1.4.5, “Installing an RPM Package”](#)
- [Section 1.4.6, “Installing an vFabric Hyperic License”](#)
- [Section 1.4.7, “What to Do After Installing the HQ Server and HQ Agent”](#)
- [Section 1.4.8, “Uninstalling an Agent”](#)

## 1.4.1. Before You Start

- [If You Are Using an External Database](#)
- [HQ Components and JRE Location](#)
  - [HQ Server JRE](#)
  - [HQ Agent JRE](#)
- [To Understand Agent - Server Communications](#)

Review these topics before starting the installation process.

### If You Are Using an External Database

If you are using an external database, setup the HQ database before installing HQ components. Refer to the instruction for the database server you use:

- [Section 1.2.1, “Set Up MySQL”](#)
- [Section 1.2.2, “Set Up Oracle”](#)
- [Section 1.2.3, “Set Up PostgreSQL”](#)

Make a note of the JDBC URL, database username, and database password---these values are required when you set up the HQ Server.

### HQ Components and JRE Location

If you are installing an HQ component from a package with with a bundled JRE on a system that doesn't not have a pre-existing JRE, the component should have no problem resolving its JRE. In other circumstances, you may need to set the HQ\_JAVA\_HOME environment variable.

#### HQ Server JRE

The order of preference when resolving the HQ Server's JRE is:

1. HQ\_JAVA\_HOME environment variable
2. embedded JRE
3. JAVA\_HOME environment variable

When installing the HQ Server with a platform-independent installer, make sure that your JAVA\_HOME environment variable points to your pre-existing JRE.

If you install the HQ Server from a package with an embedded JRE, and you want the server to use a different JRE on the host, set HQ\_JAVA\_HOME to point to the desired JRE.

#### HQ Agent JRE

On Unix-based platforms, the order of preference when resolving the agent's JRE is:

1. HQ\_JAVA\_HOME environment variable

2. Embedded JRE

3. JAVA\_HOME

On Windows, the order of preference when resolving the agent's JRE is:

1. HQ\_JAVA\_HOME **system** variable

2. Embedded JRE

**Note:** Under Windows, neither the JAVA\_HOME nor the HQ\_JAVA\_HOME environment variables are honored when resolving the JRE for HQ 4.x Agents. To run a 4.x agent under windows you must set HQ\_JAVA\_HOME as a system variable:

My Computer > Properties > Advanced > Environment Variables > System variables > New

## To Understand Agent - Server Communications

For those new to configuring HQ components, the [Section 2.1, “Understand Agent Environment and Operation”](#) summarizes key facts about how the agent and server communicate with each other.

## 1.4.2. Installing the Agent and Server from a Tarball or Zip Archive

This section has instructions for performing a new installation of Hyperic components using the full installer. If you wish to upgrade an existing Hyperic deployment, please see [Section 1.5, “Upgrade Hyperic Components”](#).

### About the Setup Script

The setup script, `setup.bat` for Windows or `setup.sh` for non-Windows, is in the Hyperic installation package. You can use to install the Hyperic Server, the Hyperic Agent, or both.

When you run the setup script, you can supply an argument that sets the installation mode.

Mode Argument	Associated Installation "Mode"
none	Quick install; the Hyperic components you choose to install will be installed with default settings for most configuration options---you supply installation directories only. If you install the server, it will be configured to use its built-in PostgreSQL database. This is the quickest way to install Hyperic for evaluation purposes.
-full	Full install; installer will prompt you to supply values for all installation options.
-upgrade	Server upgrade only; installer will prompt you for the path of the Hyperic server to upgrade. See <a href="#">Section 1.5, “Upgrade Hyperic Components”</a> for instructions.
-postgresql	Quick install when using a standalone (not the Hyperic built-in) PostgreSQL database; installer will prompt you for database connection information and use defaults for other configuration settings.
-oracle	Quick install mode for Oracle; installer will prompt you for database connection information and use defaults for other configuration settings.
-mysql	Quick install mode for MySQL; installer will prompt you for database connection information and take defaults for everything else.

### Run the Setup Procedure

This section describes the dialog that the installation script presents if you run it in `-full` mode, and select optional configuration options, such as an database and LDAP servers. Depending on the installation mode you select, some of the prompts described below will not appear.

1. Create a directory for the Hyperic installation.

- The installation dialog assumes your Hyperic installation directory is:

```
/home/hyperic
```

2. Unpack the tarball or zip archive.



**Unpack Tarballs with GNU Tar Only**

Use GNU Tar to unpacking Hyperic tarballs. Use of proprietary Unix Tar utilities will result in warnings. GNU Tar is available at <http://www.gnu.org>

3. Open a command shell.

**Opening Windows Command Shell**

On Windows, if you open your command shell from the **Start** menu, you must run as "Administrator", like this:

- a. Start button - All Programs - Accessories.
- b. Right click "Command Prompt".
- c. Select "Run as".
- d. Choose "Administrator" option.

- On Unix-based platforms, enter:

```
PathToInstaller/setup.sh -mode
```

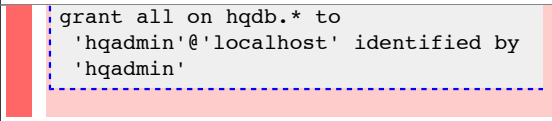
- On Windows platforms, enter:

```
PathToInstaller\setup.bat -mode
```

where *mode* is one of the values in the table above with the exception of *upgrade* — if you are upgrading an existing Hyperic deployment, refer to the instruction in [Section 1.5, “Upgrade Hyperic Components”](#).

Installation Prompt	Notes
Choose which software to install 1: Hyperic HQ Server 2: Hyperic HQ Agent	To install both the server and the agent, enter: 1,2
HQ server installation path [default '/home/hyperic']:	Accept the default, or enter a directory location. You must have write access the the location.
What port should the HQ server's web-based GUI listen on for http communication? [default '7080']:	
What port should the HQ server's web-based GUI listen on for secure https communication? [default '7443']	
Enter the base URL for the Hyperic server's web-based GUI [default...]	The URL used to access the Hyperic Server. This value is used in alert notification emails. This value can be changed on the Hyperic Server Administration page in the Hyperic Portal.
Enter the fully qualified domain name of the SMTP server that Hyperic will use to send email messages [default FQDN.local]	If the installer does not find a local SNMP server, if you do not specify one, Hyperic cannot send alert notifications. Alert functionality is still enabled.

Installation Prompt	Notes
	<b>Note:</b> You can configure Hyperic for an external SMTP server after installing Hyperic Server. See <a href="#">Section 3.6.3, “Configuring Hyperic Server for SMTP Server”</a> .
<i>Enter an encryption key to use to encrypt the database password.</i>	Supply a value of your choice, at least 8 characters.
<i>Enter the email address that HQ will use as the sender for email messages [default...]</i>	The email address of the Hyperic Administrator. Note that most mail servers will not deliver mail without a valid domain name in the From field.
<i>What backend database should the HQ server use? [default '1']: Choices: 1: HQ Built-in Database 2: Oracle 10g/11g 4: PostgreSQL 5: MySQL Enterprise / Community Server 5.x</i>	
<i>If in the previous step, you chose the HQ Built-in Database, this prompt appears: What port should HQ's built-in database use? [default '9432']:</i>	
<i>If instead you selected an external database, these prompts appear: Enter the JDBC connection URL. The prompt supplies a default URL, which assumes the external database is on local host. Enter the username to use to connect to the database: _Enter an encryption key to use to encrypt the database password: Enter the password to use to connect to the database:</i>	<p>Supply:</p> <ul style="list-style-type: none"> <li>a URL in the form shown, editing as appropriate to identify the connection details, such as host and name.</li> <li>the database username that was set up when the Hyperic database was created.</li> <li>the database password that was set up when the Hyperic database was created.</li> </ul> <div style="background-color: #ffe6e6; padding: 10px; margin-top: 10px;"> <p><b>Do you have an existing database?</b></p> <p>If the installer detects a database from a previous Hyperic installation, it will prompt you to upgrade, overwrite the existing database, or exit the installer. To preserve your existing Hyperic data, select <b>Upgrade the HQ server database</b>, and see <a href="#">Section 1.5, “Upgrade Hyperic Components”</a>. If you choose to overwrite, you will lose all the data in the Hyperic database.</p> <p>If in fact you <i>do</i> want to erase the database contents, note that it can take a long time to overwrite a large database. To erase the data, you can simply drop and recreate the Hyperic database prior to installing Hyperic Server.) The commands are:</p> <pre style="border: 1px dashed blue; padding: 5px;">drop database hqdb; create database hqdb; grant all on hqdb.* to 'hqadmin'@'%' identified by 'hqadmin'</pre> </div>

Installation Prompt	Notes
	
<i>What should the username be for the initial admin user? [ default 'hqadmin']:</i>	
<i>What should the password be for the initial admin user?:</i>	The installer will not echo the password but will prompt for it twice so it can be verified.
<i>*What should the email address be for the initial admin user? [default...]</i>	
If the installation procedure does not detect a local LDAP server, it offers the option of configuring the use of an external LDAP authorization data source. If you accept that option, you will be prompted for LDAP connection information.	LDAP authentication can also be configured after Hyperic is installed. For details see on how to configure Hyperic for LDAP authentication, see <a href="#">Section 3.6.1, “Configure LDAP Properties”</a> .
<i>HQ agent installation path [:default '/applications/hyperic']:</i>	

The installer indicates the installation was successful, provides the URL for the Hyperic Portal along with the default username and password, and returns you to the command prompt.

### 1.4.3. Installing an Agent-Only Package

This section has instructions for installing a single HQ Agent. If you have multiple agents to install, see [Section 2.7, “Deploying Multiple Hyperic Agents”](#).

- [Installing an Agent from an Agent-Only Tarball](#)
- [Installing an Agent from an Agent-Only Zip Archive](#)

#### Installing an Agent from an Agent-Only Tarball

On non-Windows systems, the Hyperic Agent is automatically installed as a daemon.

1. Create a directory for the Hyperic Agent.
2. Unpack the tarball into the agent directory.
  - Starting the agent will run it as a daemon process.

##### Unpack Tarballs with GNU Tar Only

Use GNU Tar to unpacking Hyperic tarballs. Use of proprietary Unix Tar utilities will result in warnings. GNU Tar is available at <http://www.gnu.org>

For instructions on how to configure the agent, see [Section 2, “Configure and Run the Hyperic Agent”](#).

#### Installing an Agent from an Agent-Only Zip Archive

To install the Hyperic Agent as a Windows Service on a Windows system:

1. Create a directory for the Hyperic Agent.
2. Unpack the archive into the agent directory.
3. Open a command shell and use this command to install the Hyperic Agent as a Windows Service:

```
AGENT_HOME\bin\hq-agent.bat install
```

For instructions on how to configure the agent, see [Section 2, “Configure and Run the Hyperic Agent”](#).

## 1.4.4. Installing HQ Using the Windows MSI Installer

This section has instructions for installing components using the HQ Windows MSI installation package. You can install the HQ Server, the HQ Agent, or both. The MSI installer can be run interactively or in silent mode.

- [Known Issues on Windows](#)
- [Installing HQ Interactively Using the HQ MSI Installer](#)
- [Solving Service Startup Problems After MSI Install](#)
- [Syntax for Running MSI Installer in Silent Mode](#)
- [Example Silent Mode MSI Invocations](#)
- [MSI Silent Mode Properties](#)
- [Silent MSI installation to Multiple Hosts Using Push Techniques](#)

### Known Issues on Windows

These requirements apply when you use the MSI installer on Windows:

- Installation requires administrator privileges and permissions to install Windows services.
- Uninstallation must be performed by the same user who performed the installation.

### Installing HQ Interactively Using the HQ MSI Installer

Follow these steps to run the MSI installer interactively:

1. Double-click the MSI package.
  - A Welcome panel is displayed.
2. Click **Next**.
  - The SpringSource license agreement is displayed.
3. Accept the license agreement and click **Next**.
  - The **Destination Folder** window is displayed, and supplies a default installation folder in the root of the Program Files folder.
4. If desired, select a different installation folder, and click **Next**.
  - The **Setup Type** window is displayed.
5. Choose **Complete** to install both the HQ Server and the HQ Agent, or **Custom** to install one or the other.
  - The **Ready to Install...** window displays the selected installation location, components to install, and a check box for choosing whether or not to install the agent and server as Windows services - uncheck the box if you do not want the services to be started after the installation is complete.
6. Click **Install**.

## Solving Service Startup Problems After MSI Install

If you install the HQ Server and the HQ Agent on the same machine, and accept the default "Start Hyperic HQ Services when install completes" option, agent startup problems can result.

In this scenario, as a last step, the installer will issue a server start command, followed by an agent start command.

The agent must contact the server to start up successfully. If the machine the HQ components run on is slow or busy, the HQ Server can take a long time to start. The HQ Agent makes a finite number of attempts to connect to the server, and if it continues to fail, the agent gets stuck. No software will be auto-discovered on the platform and the agent will not appear in the HQ user interface.

To solve this problem, force the agent to repeat the setup process by entering this command in a shell:

```
AGENT_HOME/bin/hq-agent.bat setup
```

## Syntax for Running MSI Installer in Silent Mode

You can run the HQ MSI installer in silent mode from the DOS prompt. For example:

- To install Hyperic HQ, type the following command in a terminal window, supplying the desired installation location, correct installer file name, and the desired installation properties - defined below in [MSI Silent Mode Properties](#). The /qn switch turns off the user interface.

```
%Comspec% /c msixec /i "installer_path\hyperic-hq-installer-4.5.n.build.msi" /qn  
PROPERTY1=VALUE1 PROPERTY2=VALUE2 ...
```

- To uninstall HQ Enterprise in silent mode, type the following command in a terminal window. You must uninstall from the same user account that used to perform the installation.

```
%Comspec% /c msixec /x "installer_path\hyperic-hqee-installer-4.5.n.build.msi" /qn
```

### Opening Windows Command Shell

On Windows, if you open your command shell from the **Start** menu, you must run as "Administrator", like this:

1. Start button - All Programs - Accessories.
2. Right click "Command Prompt".
3. Select "Run as".
4. Choose "Administrator" option.

## Example Silent Mode MSI Invocations

- To silently install HQ Server and the HQ Agent on a local machine under "C:\hyperic":

```
%Comspec% /c msixec /i "installer_path\hyperic-hq-installer-4.5.n.build.msi" /qn  
INSTALLDIR="C:\hyperic"
```

- To install (locally) an HQ Agent that will communicate securely with the HQ Server at 69.59.181.106:

```
%Comspec% /c msixec /i "installer_path\hyperic-hq-installer-4.5.n.build.msi" /  
qn ADDLOCAL=Agent AGENT_IS_SECURE=1 AGENT_SERVER_ADDRESS=69.59.181.106  
AGENT_SERVER_USER=hqadmin AGENT_SERVER_PASSWORD=password
```

- To install the HQ Server and the HQ Agent on a local machine using an MSI installer on a remote machine that is accessible on the network:

```
%Comspec% /c msexec /i "\\network_path\hyperic-hq-installer-4.5.n.build.msi"
PROPERTY1=VALUE1 PROPERTY2=VALUE2 ...
```

- To silently install the HQ Server and the HQ Agent to a local machine using an MSI installer on a remote machine that is accessible on the network, include the /qn switch to turn off the user interface:

```
%Comspec% /c msexec /i "\\network_path\hyperic-hq-installer-4.5.n.build.msi" /qn
PROPERTY1=VALUE1PROPERTY2=VALUE2...
```

## MSI Silent Mode Properties

Installation properties for installing HQ in MSI silent mode are described below.

Note that all properties and their values are case-sensitive.

Properties that begin with the strings SERVER and AGENT are server and agent properties, respectively.

Silent Mode Property	Description	Default
INSTALLDIR	Directory where the HQ components will be installed.	C:\Program Files\Hyperic HQ 4.5
ADDLOCAL	Comma-separated list of components to be installed. Allowable case-sensitive values are Agent and Server. Use this property if you want to install only the HQ Server or the HQ Agent.	If you do not specify ADDLOCAL, both agent and server will be installed.
HQ_ENGINE_JNP_PORT	The JNDI listen port. The value assigned will be saved in the <code>hq-engine.jnp.port</code> property in <code>server.conf</code> .	2099
HQ_ENGINE_PORT	The JRMP listen port. The value assigned will be saved in the <code>hq-engine.server.port</code> property in <code>server.conf</code> .	9093
SERVER_ADMIN_EMAIL	HQ Server Administrator's email address. Default is based on values of <code>SERVER_ADMIN_USER</code> and <code>SERVER_MAIL_HOST</code> properties.	<code>SERVER_ADMIN_USER@SERVER_MAIL_HOST</code>
SERVER_ADMIN_USER	Specifies the user name of the original admin user in HQ. Will be used in configuring that account.	hqadmin
SERVER_ADMIN_PASSWORD	Specifies the password of the original admin user in HQ server. Will be used in configuring that account.	hqadmin



Silent Mode Property	Description	Default
SERVER_DATABASE_USER	Defines the username the HQ server will use when connecting to the HQ database. The value assigned will be saved in the <code>server.database-user</code> property in <code>server.conf</code> .	hqadmin
SERVER_DATABASE_PASSWORD	Defines the password the HQ server will use when connecting to the HQ database. The value assigned will be saved in the <code>server.database-password</code> property in the <code>server.conf</code> file.	hqadmin
DB_ENC_KEY_PW	The Database Encryption Key Password. This value will be used to encrypt the database password so it can not be easily read. Must be at least 8 characters long, there are no restrictions on the type of characters that may be entered.	
SERVER_MAIL_HOST	The IP address or hostname of the SMTP server that the HQ server will use for sending alerts and other HQ-related emails. Most UNIX platforms have a local SMTP server. If you wish to use a non-local SMTP server, specify the address with this property. The value assigned will be saved in the <code>server.mail.host</code> property in <code>server.conf</code> file.	127.0.0.1
SERVER_MAIL_SENDER	The 'From' address in email notifications from the HQ Server.	SERVER_ADMIN_EMAIL
SERVER_POSTGRESQL_PORT	HQ Server's embedded database listen port.	9432
SERVER_WEBAPP_HOST	Specifies the HQ Server's listen address for HQ Portal communications. By default, this property and <code>AGENT_SERVER_ADDRESS</code> have the same value. If you wish, you can use these properties to designate different hosts for agent-server and agent-portal communications.	host IP address

Silent Mode Property	Description	Default
	The value assigned will be saved in <code>server.webapp.host</code> in <code>server.conf</code> .	
SERVER_WEBAPP_PORT	<p>Specifies the HQ Server listen port on which the server listens for HQ Portal communications in non-secure mode.</p> <p>By default, this property and <code>AGENT_SERVER_PORT</code> have the same value. If you wish, you can use these properties to designate different ports for agent-server and agent-portal communications.</p> <p>The value assigned will be saved in the <code>server.webapp.port</code> property in <code>server.conf</code>.</p> <p>The HQ Portal Dashboard will be located at the URL of the form:</p> <p><code>http:// SERVER_WEBAPP_HOST:SERVER_WEBAPP_PORT</code></p>	7080
SERVER_WEBAPP_SECURE_PORT	<p>Specifies the HQ Server port on which the server listens for HQ Portal communications in secure mode.</p> <p>The value assigned will be saved in the <code>server.webapp.secure.port</code> property in <code>server.conf</code>.</p>	7443
HQ_START_SERVICES	Property indicating whether the HQ Agent and Server processes should be started as a Windows Service at the end of the installation. 1 indicates true, 0 indicates false.	1
AGENT_ADDRESS	<p>The IP address to which the agent binds at startup. The default value allows the agent to listen on all IP addresses on the the agent host.</p> <p>The value assigned is saved in both the <code>agent.listenIp</code> and the <code>agent.setup.agentIP</code> properties in <code>agent.properties</code>.</p> <p>If there is a firewall between the agent and the server, set <code>AGENT_ADDRESS</code> to the firewall address. After installation is complete, set <code>agent.listenIP</code> in <code>agent.properties</code> to the agent's local IP address, and configure the</p>	host IP address

Silent Mode Property	Description	Default
	wall to forward agent-bound traffic to that address.	
AGENT_IS_SECURE	Indicates whether communications between the HQ Agent and the HQ Server should take place over a secure encrypted channel. 1 indicates secure communications, 0 indicates that communications will not be secured. The setting will be stored appropriately in the agent.setup.camSecure property in agent.properties.	0
AGENT_PORT	The port on the agent's listen address to which the agent binds at startup. This value is saved to both agent.setup.agentPort and agent.listenPort in agent.properties.	2144
AGENT_SERVER_ADDRESS	Specifies the IP address the agent connects to to reach the HQ server. The value is saved to agent.setup.camIP in agent.properties.	host IP address
AGENT_SERVER_USER	The HQ username to use when registering itself with the server. The value is saved to agent.setup.camLogin in agent.properties. Typically this property and AGENT_SERVER_PASSWORD have the same values as SERVER_ADMIN_USER and SERVER_ADMIN_PASSWORD respectively. However, if you are installing a server and an agent on the same host, and the agent will report to a server on a different host, you might specify different credentials for AGENT_SERVER_USER/ AGENT_SERVER_PASSWORD and SERVER_ADMIN_USER/ SERVER_ADMIN_PASSWORD.	hqadmin
AGENT_SERVER_PASSWORD	The password for the user specified by AGENT_SERVER_USER. The value is saved to agent.setup.camPword in agent.properties.	hqadmin

Silent Mode Property	Description	Default
AGENT_SERVER_PORT	Port on server port to use for non-secure communications with the server. The value is saved to agent.setup.camPort in agent.properties.	7080
AGENT_SERVER_SSL_PORT	Port on server to use for SSL communications with the server. The value is saved to agent.setup.camSSLPort in agent.properties.	7443

## Silent MSI installation to Multiple Hosts Using Push Techniques

This section describes alternatives for doing silent MSI installs to multiple machines.

### Using AT or SOON to start a process on a remote workstation

The AT and SOON commands can be used to schedule commands at a future time. AT, which is built into the command processor, schedules commands and programs to run on a local or remote computer at a specified time. Instead of running processes at a specific time, the SOON command runs them after a specified delay. SOON.EXE is available as a free Microsoft download.

Here are examples of how to run these commands:

```
AT targetPC 10:30 /INTERACTIVE \\myPC\myShare\quietInstall.bat
SOON targetPC 30 /INTERACTIVE \\myPC\myShare\quietInstall.bat
```

Executing processes on a remote system has security implications.

- The local machine must have sufficient privileges to start a batch routine on a remote system.
- You must establish privileges for the remote system to access network resources when running the install batch routine. When the command processor runs your batch routine on the target system, it executes with Local System privileges. It is therefore necessary for the batch routine to open a privilege pipe to the network resource containing the MSI package. A workaround is to add a NET command to your batch routine, as demonstrated in this sample quietInstall.bat:

```
net use * \\myPC\myShare /user:domain\username password /persistent:no
%Comspec% /c msixec /i "\\myPC\myShare\hyperic-hq-installer-4.5.n.build.msi" /qn
```

### Using PsExec to Start a Process on a Remote System

The Windows PsExec utility is a freely distributed light-weight telnet replacement that lets you execute processes on other systems, complete with full interactivity for console applications, without having to manually install client software. PsExec's most powerful uses include launching interactive command prompts on remote systems and remote-enabling tools to show information about remote systems. It can be downloaded as part of the Sysinternal PsToolspackage.

PsExec can be used to run the batch routine at a remote workstation by invoking the following command:

```
psexec targetPC -u domain\username -p password -i -c -f
```

```
\\myPC\myShare\quietInstall.bat
```

In the example above, domain\username has local administrative privileges for the targetPC machine. In addition, it should have the necessary privileges to access the myShare folder on the myPC machine.

The batch file quietInstall.bat is used to invoke the MSI installer over the network on the myPC machine. A sample quietInstall.bat might contain the following command:

```
%Comspec% /c msixec /i "\\myPC\myShare\Hyperic HQ 3.2.msi" /qn
```

## Remote Installs with Microsoft Management Console

This section has information on how to automatically install components to a group of machines, using Microsoft Management Control and Active Directory.

With Windows Group Policy, HQ components can be automatically installed on a group of machines by performing the following steps:

1. Log on to the domain controller.
2. Copy the MSI file into a folder that is shared with access granted to all target machines.
3. Open the Microsoft Management Control (MMC) Active Directory Users and Computers snap-in.
4. Navigate to the group of computers onto which an HQ component is to be deployed.
5. Open Properties.
6. Open Group Policies.
7. Add a new policies, and edit it.
8. In Computer Configuration/Software Installation, chose New/Package.
9. Select the MSI file through the network path.
10. Optionally, select that you want HQ to be uninstalled if the computer leaves the scope of the policy.

Propagation of group policy propagation typically takes some time. In order to reliably deploy the HQ MSI package, all machines should be rebooted.

## 1.4.5. Installing an RPM Package

This section has key facts for Linux administrators who will install Hyperic components from RPM packages. It is assumed that the administrator performing RPM installations is familiar with RPM packages and installation processes.

- [What You Should Know About Hyperic RPM Packages](#)
- [What You Need to Do Before Installing an Hyperic RPM](#)
- [What the RPM Package Does](#)
- [RPM Support Files](#)
- [What to Do After Installing the Hyperic Server and Hyperic Agent](#)

### What You Should Know About Hyperic RPM Packages

Hyperic RPM Packages include:

- `hyperic-agent-n.n.n-n.n.noarch.rpm` - The Hyperic Agent RPM to install the Hyperic Agent. The Hyperic Agent RPM does not include a JRE. Agent hosts must have the J2RE virtual package installed. A Sun 1.5 JRE is recommended.
- `hyperic-hq-installer-n.n.n-n.n.x86_64.rpm` - The Hyperic Server RPM to install only the Hyperic Server; installation of the agent with this RPM is not available. The Hyperic Server RPM includes a 1.6 JRE and the built-in PostgreSQL database. Note that this RPM is based on the standard Hyperic Server installation script, which is wrapped in an Expect script. The Hyperic Server RPM is primarily intended for evaluation installations in environments that dictate the use of RPM.

### What You Need to Do Before Installing an Hyperic RPM

- **Configure Hyperic Agent Properties** - A Hyperic Agent obtains the settings it needs to connect to and communicate with the Hyperic Server at first startup, either interactively, or from startup properties that can be specified in the `agent.properties` file. If you wish to automate agent installation and configuration, you must edit `agent.properties` to specify the startup properties. For more information, see [Section 2.2.2, “Configure Agent - Server Communication in Properties File”](#).
- **Check Path to JRE** - The agent init script is installed as `/etc/init.d/hyperic-hq-agent`, and is added to the appropriate run-levels via `chkconfig`. The script assumes the path to your JRE is `/usr/java/jdk1.5.0_12`. If this is not the case on the target host, you must modify the init script and specify the path to your JRE in the `HQ_JAVA_HOME` environment variable.
- **Open Firewall Port if Necessary** - If `iptables` (a host-based firewall tool typically enabled by default on Redhat and Fedora installations) is configured, you may need to open up the port for communication from the Hyperic Server, using a command similar to this:

```
^/sbin/iptables -A RH-Firewall-1-INPUT -p tcp --dport 2144 -j ACCEPT^
```

Additional configuration may be required if SELinux is enabled.

- **Start Local SMTP Server** - The RPM installer requires that your SMTP server is listening on port 25 on the host where you install Hyperic Server.

## What the RPM Package Does

If the "hyperic" user and "hyperic" group do not exist, they are created.

The /opt/hyperic directory is created if it does not already exist. This is set as the "hyperic" user's home directory if that user did not exist previously, and appropriate permissions are set.

The /opt/hyperic/hq-plugins directory is created to hold custom plugins.

The agent is installed in /opt/hyperic-hq-agent.

## RPM Support Files

If you wish to create your own RPMs for installing Hyperic components, you can download rpm\_support\_files\_EE.tgz, which contains the RPM spec files, init scripts, and other necessary files, from <http://www.hyperic.com/downloads/>

The spec file is noarch-EE.spec. The init script is hyperic-hq-agent.init.rh.

## What to Do After Installing the Hyperic Server and Hyperic Agent

If you have installed both the server and an agent, start the server first, and then start the agent.

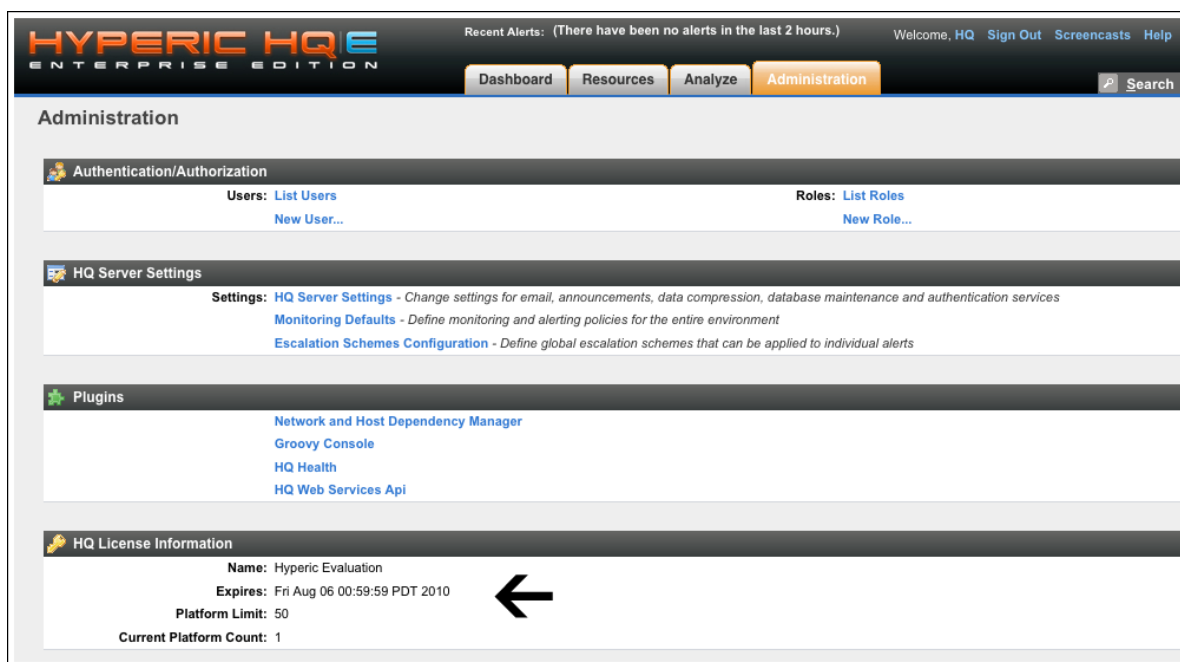
- For information about starting and configuring the server see [Section 3, “Configure and Run the Hyperic Server”](#).
- For information on starting the agent the first time, see [Section 2, “Configure and Run the Hyperic Agent”](#).

## 1.4.6. Installing an vFabric Hyperic License

In vFabric Hyperic, the `license.xml` file in the Hyperic Server's `/conf` directory specifies the number of platforms you are licensed to manage, and the expiration date of the license, as applicable. Perpetual licenses have no expiration date.

vFabric Hyperic evaluation distributions include a time-limited license for 50 platforms.

You can view your license terms on the **HQ License Details** section of the **Administration** tab, as shown in the screenshot below.



After you purchase vFabric Hyperic, you download `license.xml` from the SpringSource Portal and install it in `ServerHome/conf`. You need to obtain a new license to increase the number of managed platforms or upon expiration of your license.

vFabric Hyperic sends an email notification of upcoming expiration starting 45 days prior to the expiration date.



## 1.4.7. What to Do After Installing the HQ Server and HQ Agent

Topics marked with\* relate to features available only in vFabric Hyperic.

- [Obtain and Install Drivers for Monitoring Databases](#)
- [Starting HQ Components](#)
- [Installing Additional Agents](#)

### Obtain and Install Drivers for Monitoring Databases

In Hyperic HQ, the plugins packaged with the HQ Agent for MSSQL, Oracle, Informix, DB2, and Sybase do not include the database vendor's JDBC plugin. After installing Hyperic HQ you must download and install the vendor-provided JDBC drivers for these plugins to work. Install drivers in:

```
AgentHome/bundles/AgentBundle/pdk/lib/jdbc
```

**Note:** The database plugins in vFabric Hyperic include the JDBC drivers.

### Starting HQ Components

If you have installed both the server and an agent, start the server first, and then start the agent.

For information about starting and configuring your HQ server see [Section 3, “Configure and Run the Hyperic Server”](#).

For information on starting the agent the first time, see [Section 2, “Configure and Run the Hyperic Agent”](#).

### Installing Additional Agents

If you have installed the server only, and are going to install one or more agents on other platforms, see [Section 1.4.3, “Installing an Agent-Only Package”](#) or [Section 2.7, “Deploying Multiple Hyperic Agents”](#).

## 1.4.8. Uninstalling an Agent

If the agent is managed by Hyperic, remove the platform for the agent before uninstalling it. Then, simply delete the agent's installation folder.

If the agent is installed the agent as a Windows service, run `hq-agent.bat remove` to remove the Windows service.

## 1.5. Upgrade Hyperic Components

Topics marked with\* relate to features available only in vFabric Hyperic.

This section has instruction for upgrading your Hyperic deployment to a new version. You should upgrade both the Hyperic Server and the Hyperic Agent to the same version.

- [Section 1.5.1, “Upgrade Hyperic Server”](#)
- [Section 1.5.2, “Upgrade Hyperic 4.x Agents”](#)
- [Section 1.5.3, “Upgrade Hyperic 3.1.x and 3.2.x Agents”](#)

## 1.5.1. Upgrade Hyperic Server

You upgrade the Hyperic Server using the full installer, using the upgrade option. (The installer does not upgrade the Hyperic Agent.)

### What Happens During Server Upgrade

The installer installs a new version of Hyperic Server; it obtains the configuration information from your previous server installation and configures the new server instance accordingly.

If you use Hyperic's internal database, the installer creates a new database instance that contains the data from the existing instance. The new instance has an updated schema, but the PostgreSQL server itself is not upgraded to a new version.

If you use an external database, the installer updates the existing instance.

### Upgrade Hyperic Server on Unix-Based Platforms

1. Stop the current server instance. For example:

```
/opt/hyperic/server-4.5.0/bin/hq-server.sh stop
```

2. **If you use an external Hyperic database, back it up before proceeding.**

3. Run the Hyperic installer in upgrade mode. For example:

```
/opt/hyperic/hyperic-hq-installer/setup.sh -upgrade
```

4. You are prompted to acknowledge the SpringSource license agreement.

5. The installer prompts for the path to the previous Hyperic Server instance. Enter the path, for example:

```
/opt/hyperic/server-4.5.0
```

6. The installer prompts for the path to the new server instance. Enter the path to the directory under which the new server instance will be installed. For example, to install the new instance under your existing Hyperic home directory:

```
/opt/hyperic
```

- The installer will finish the upgrade.

#### Update Java options

In this version of Hyperic, to avoid a known problem, after upgrading the Hyperic Server, you must edit the `server.java.opts` property in `server.conf` to add the `-Djava.awt.headless=true` option.

7. Archive your old Hyperic Server directory, so that if you want, you can revert to the previous version. For example:

```
tar -zcvf hq-server-4.5.0-archive.tgz hq-server-4.4.0-EE
```

8. Start the new server instance. For example:

```
/opt/hyperic/server-4.5.0/bin/hq-server.sh start
```

## Upgrade Hyperic Server on Windows Platforms

1. Stop the existing server instance using the Windows Services Control Panel.
2. Follow the instructions that apply, depending on whether you use the Hyperic built-in database or an external database:
  - If you use the built-in Hyperic database, the upgrade process will migrate your database schema to the latest edition. Note that PostgreSQL itself is not upgraded to the latest version that ships with Hyperic. The database server remains the one installed when you first installed Hyperic Server.
  - **If using an external database, back it up.**
3. Run the Hyperic installer in upgrade mode:

```
c:\hyperic\hyperic-hq-installer\setup.bat -upgrade
```

4. You are prompted to acknowledge the SpringSource license agreement.
5. The installer prompts for the path to the previous Hyperic Server instance. Enter the full path to your existing server installation, for instance:

```
c:\hyperic\server-4.5.0
```

6. The installer prompts for the path where the upgrade version should be installed. Enter the path to the directory that will contain the new server installation. For instance, to install the new instance under your existing Hyperic home directory:

```
c:\hyperic\
```

- The installer will finish the upgrade.

### Update Java options

In this version of Hyperic, to avoid a known problem, after upgrading the Hyperic Server, you must edit the `server.java.opts` property in `server.conf` to add the `-Djava.awt.headless=true` option.

7. Archive your previous Hyperic Server directory so that if you wish you can revert to the previous version.
8. Update the Windows Service with the new version information:

```
c:\hyperic\server-4.5.0\bin\hq-server.bat install
```

9. Start the upgraded Hyperic Server using the Windows Services Control Panel.

## Solving Problems with Upgraded Servers with an Oracle Database

If you are upgrading an Hyperic installation with an Oracle backend and you experience any of the following errors during upgrade, follow the steps below to resolve the problem.

```
Error updating EAM_SERVICE.SERVICE_TYPE_ID: java.sql.SQLException:
ORA-02296: cannot enable (HQBUSER.) - null values found
Error executing statement desc=null SQL=[
ALTER TABLE eam_stat_errors DROP CONSTRAINT rt_errs_fk_rstat CASCADE
] java.sql.SQLException: ORA-02443: Cannot drop constraint - nonexistent
constraint
```

Fix this with these steps:

1. Restore your database from backup.
2. Execute this SQL:

```
DELETE FROM EAM_SERVICE WHERE SERVICE_TYPE_ID IS NULL;
```

3. Re-run the upgrade.

## Clear Browser Cache Before Using the Portal

If you are upgrading to 4.x from a 3.2.x or 3.1.x version, before using the Hyperic user interface, users must clear the browser cache, or reload the Dashboard using the Shift Refresh key sequence.

If you do not clear the browser cache, portions of the Hyperic user interface may display improperly.

## 1.5.2. Upgrade Hyperic 4.x Agents

### Agent Name Unchanged by Upgrade Process

Note that if you upgrade a 4.x agent by pushing a new bundle from the Hyperic Server, the name of the agent is not changed. So, an agent name that contains the Hyperic version number — as is the default naming convention — reflects the originally installed version, rather than the version to which it has been upgraded. For example, if you push a 4.4 bundle to an agent whose name is "HQ Agent 4.3", the agent name will remain "HQ Agent 4.3".

### No Agent Build Number in Hyperic 4.5

Prior to Hyperic 4.5, the build number of a Hyperic Agent was a resource property that appeared the Hyperic user interface for an agent selected in the Resource Hub. Starting in Hyperic 4.5, there is no agent build number property. Note that if you upgrade a 4.x agent to 4.5 by pushing a new bundle from the Hyperic Server, the build number of the previous version of the agent still appears in the user interface, but does not apply to the 4.5 agent.

## Push Agent Bundle from the Hyperic Server

Available only in **vFabric Hyperic**

You can update one or more Hyperic Agents by pushing the new bundle to it from the Hyperic Server, using the Hyperic user interface. The bundle must reside in the /hq-agent-bundles directory of the HQ Server installation:

```
ServerHome/hq-engine/hq-server/webapps/ROOT/WEB-INF/hq-agent-bundles
```

The agent upgrade command is available on the **Views** tab for an Hyperic Agent (or a group of agents). For more information, see [Agent Control Commands](#).

**Note:** When you update an agent bundle, its previous agent configuration is preserved.

## Upgrade a 4.x Agent Bundle Manually

Available only in **vFabric Hyperic**

Follow these steps if you wish to manually upgrade the agent bundle in your agent installation, instead of pushing the bundle from the Hyperic Server.

**Note:** When you update an agent bundle, your previous agent configuration is preserved.

1. Copy the agent bundle (agent-4.x.y-nnn.tgz or agent-4.x.y-nnn.zip) from

```
ServerHome/hq-engine/hq-server/webapps/ROOT/WEB-INF/hq-agent-bundles
```

to

```
AgentHome/bundles
```

2. Unpack the agent bundle.
3. Edit the rollback.properties file in AgentHome/conf to specify the location of the new agent bundle and the bundle it will supersede.

Property	Description	Example
HQ_AGENT_BUNDLE	Name of directory with the new bundle, without full path specification.	agent-4.5.0-EE-nnn
HQ_AGENT_ROLLBACK_BUNDLE	Name of directory with the old bundle (the one you are upgrading from), without full path specification.	agent-4.5.0-EE-nnn

- Restart the agent. For instructions, see [Restart the Hyperic Agent](#).

### First Agent Startup

The first time you start the agent, it will prompt for start settings. If you prefer, you can supply the settings in the agent properties file before starting the agent the first time. For instructions, see [Section 2.2.2, “Configure Agent - Server Communication in Properties File”](#).

If the upgrade to the new agent bundle fails, an attempt will be made to start the agent using the old agent bundle.

You can determine whether the upgrade was successful and what version you are running by looking at the log files in AgentHome/logs.

## Create a Custom 4.x Agent Upgrade Bundle

Available only in **vFabric Hyperic**

This section describes how to create a custom agent bundle. Pre-configuring the agent eases the process of upgrading multiple agents. For additional information, see [Section 2.7, “Deploying Multiple Hyperic Agents”](#).

- Back up an existing agent located in:

```
ServerHome/hq-engine/hq-server/webapps/ROOT/WEB-INF/hq-agent-bundles
```

- For example:

```
cp ServerHome/hq-engine/hq-server/webapps/ROOT/WEB-INF/hq-agent-bundles/agent-4.5.0-EE-nnn.tgz
ServerHome/hq-engine/hq-server/webapps/ROOT/WEB-INF/hq-agent-bundles/agent-4.5.0-EE-nnn.tgz.bak
```

- Extract the bundle. For example:

```
tar xzf ServerHome/hq-engine/hq-server/webapps/ROOT/WEB-INF/hq-agent-bundles/agent-4.5.0-EE-nnn.tgz
```

- This results in a new directory corresponding to the agent bundle, like this:

```
ServerHome/hq-engine/hq-server/webapps/ROOT/WEB-INF/hq-agent-bundles/agent-4.5.0-EE-nnn
```

- Update the contents of expanded directory. For instance, you could add custom plugins to the plugins directory:

```
ServerHome/hq-engine/hq-server/webapps/ROOT/WEB-INF/hq-agent-bundles/agent-4.5.0-EE-nnn/pdk/plugins
```



4. Rename expanded directory to the name of custom agent bundle. For example:

```
mv ServerHome/hq-engine/hq-server/webapps/ROOT/WEB-INF/hq-agent-bundles/agent-4.5.0-EE-  
nnn  
ServerHome/hq-engine/hq-server/webapps/ROOT/WEB-INF/hq-agent-bundles/my-bundle
```

5. Pack up agent bundle, using the directory name from the previous step as the tarball file name. For example:

```
tar cvf ServerHome/hq-engine/hq-server/webapps/ROOT/WEB-INF/hq-agent-bundles/my-  
bundle.tar  
ServerHome/hq-engine/hq-server/webapps/ROOT/WEB-INF/hq-agent-bundles/my-bundle;  
gzip ServerHome/hq-engine/hq-server/webapps/ROOT/WEB-INF/hq-agent-bundles/my-bundle
```

## Upgrade a 4.x Agent Using a Full Agent Package

These instructions apply to both Hyperic HQ and vFabric Hyperic.

To upgrade an Hyperic 4.x Agent using full the agent package:

1. Stop the 4.x agent.
2. Preserve the existing agent configuration.
  - Back up the `agent.properties` file from your previous installation. The default location for `agent.properties` in 4.x installations is the `AgentHome/conf` directory.
  - Note: In some Hyperic environments, `agent.properties` is stored in an alternative location that eases the process of automating the deployment of multiple agents. On Unix-based platforms, that location is the `.hq` subdirectory of the home directory of the user that runs the Agent. If your agent configuration is stored in that location, it will not be over-written by the new installation.
3. If the agent runs on Windows, uninstall the agent service from a command shell in `AgentHome/bin`:

```
hq-agent.bat remove
```
4. Unpack the 4.y agent into the agent installation directory.
5. On Windows, install the new agent service. In a command shell in `AgentHome/bin` enter:

```
hq-agent.bat install
```
6. Start the agent. For instructions, see [Start the Hyperic Agent](#).

### First Agent Startup

The first time you start the agent, it will prompt for start settings. If you prefer, you can supply the settings in the agent properties file before starting the agent the first time. For instructions, see [Section 2.2.2, "Configure Agent - Server Communication in Properties File"](#).

### 1.5.3. Upgrade Hyperic 3.1.x and 3.2.x Agents

These instructions apply to both Hyperic HQ and HQ Enterprise.

You must use an agent-only package to upgrade 3.1.x and 3.2.x agents to 4.x.

1. Stop the 3.1.x or 3.2.x agent.
2. Unpack the 4.x agent into the agent installation directory.
3. To preserve your previous configuration settings, copy property settings that you have customized from the 3.2.x or 3.1.x agent.properties file into the 4.x properties file, in *AgentHome/conf*.
  - There are new properties as of Hyperic 4.0, so you cannot use a 3.x properties file.
4. Install the Hyperic Agent service (Windows only):

```
AgentHome\bin\hq-agent.bat install
```

5. Start the agent. For instructions, see [Start the Hyperic Agent](#).

#### First Agent Startup

The first time you start the agent, it will prompt for start settings. If you prefer, you can supply the settings in the agent properties file before starting the agent the first time. For instructions, see [Section 2.2.2, “Configure Agent - Server Communication in Properties File”](#).

#### Upgrading from 3.x

When you first upgrade an HQ Enterprise 3.1.x or 3.2.x agent to 4.x, you cannot configure unidirectional communications at first startup. You must configure bidirectional communications at first startup of an agent upgraded from 3.x to 4.x, and then, follow the instructions in [Section 2.2.4, “Configure Unidirectional Agent - Server Communication”](#).

## 2. Configure and Run the Hyperic Agent

This page is a linked table of contents for *Configure and Run the Hyperic Agent*.

- [Section 2.1, “Understand Agent Environment and Operation”](#)
- [Section 2.2, “Configure Agent - Server Communication”](#)
- [Section 2.3, “Start, Stop, and Other Agent Operations”](#)
- [Section 2.4, “Configure Auto-Discovery Scanning and Reporting”](#)
- [Section 2.5, “Configure Agent Logging”](#)
- [Section 2.6, “Configure Plugin Loading”](#)
- [Section 2.7, “Deploying Multiple Hyperic Agents”](#)
- [Section 2.8, “Tweak the Agent to Enable a Resource Plugin”](#)
- [Section 2.9, “Manage the Hyperic Agent”](#)
- [Section 2.10, “Agent Properties”](#)

## 2.1. Understand Agent Environment and Operation

These topics that explain key facts about agent configuration, startup, and communication:

- [Section 2.1.1, “Agent - Server Communication”](#)
- [Section 2.1.2, “Agent Server Communications Diagram”](#)
- [Section 2.1.3, “Agent Launcher and Agent Startup”](#)
- [Section 2.1.4, “Agent Configuration”](#)
- [Section 2.1.5, “Agent Directory Structure”](#)

### 2.1.1. Agent - Server Communication

This section is an overview of the communications between an HQ Agent and the HQ Server. The default communication between an agent and the server is bi-directional — some communication is initiated by the agent, some by the server:

#### Agent-to-Server Communication

Upon startup, an HQ Agent initiates a communications channel with the HQ Server. The agent continuously batches and sends monitoring results for the platform to the server. Agent-to-server data flows over HTTPS via a byte-encrypted XML protocol called Lather. The agent sends this data to the server:

- metric — the batch size is configurable.
- event — the batch size is configurable
- auto-discovery results — Auto-discovery results are reported after a each auto-discovery scan. Default scans run every 15 minutes, unless otherwise configured. Run-time scans run once a day, unless otherwise configured. Either type of scan can be run on-demand.

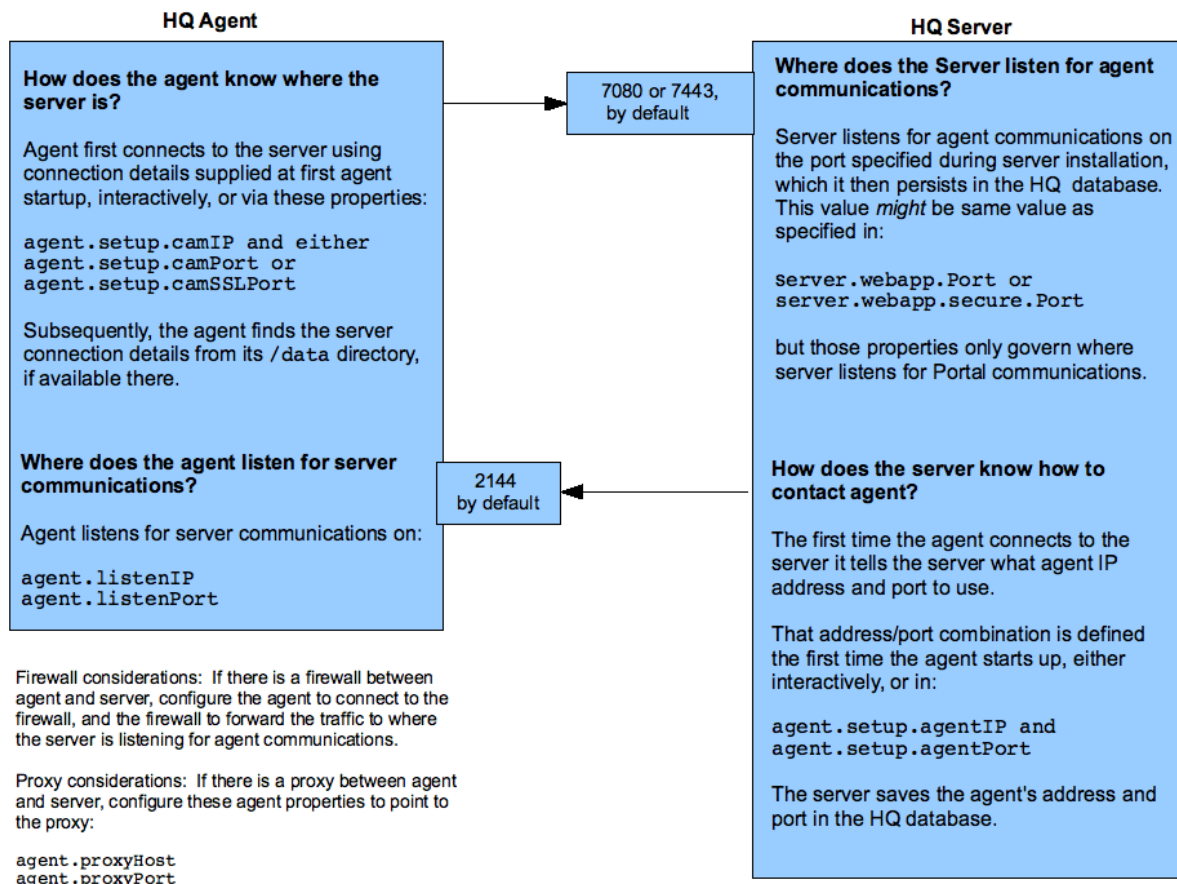
#### Server-to-Agent Communication

The HQ Server initiates communication with an HQ Agent to relay the commands and data issued or configured by authorized users. The server-to-agent traffic is sent using a simple protocol directly on top of TCP, and includes:

- metric collection schedules
- resource control actions
- requests to initiate auto-inventory scans

If your security policies dictate, you can configure the agent to initiate all communications with the HQ Server. You can configure unidirectional communications at first startup of a new 4.x agent, or an upgraded 4.x agent. If you want to change from bidirectional to unidirectional communications at a later time, see [Section 2.2.4, “Configure Unidirectional Agent - Server Communication”](#).

## 2.1.2. Agent Server Communications Diagram



### 2.1.3. Agent Launcher and Agent Startup

- [Agent Launcher](#)
- [What Happens When an Agent Starts Up](#)
- [Automatic Restart Behavior](#)

This section describes the HQ Agent launcher and the agent startup process.

#### Don't Need the Background Explanation?

See:

- [Section 2.3, “Start, Stop, and Other Agent Operations”](#)
- [Section 2.2, “Configure Agent - Server Communication”](#)

### Agent Launcher

The HQ Agent launcher is based on the Java Service Wrapper (JSW), a configurable tool that allows Java applications to be run as an NT service or Unix daemon process. It includes fault correction software to automatically restart crashed or frozen JVMs.

JSW runs as a native executable; it invokes and monitors the HQ Agent's JVM, based on configuration information provided to the wrapper at startup. In this way, the wrapper supports restarts of the JVM process without stopping the wrapper process itself. The JSW process acts as a watchdog for the JVM process, periodically pinging it for availability.

For more information about the features of the Java Services Wrapper, and how to configure its behavior, see <http://wrapper.tanukisoftware.org/doc/english/download.jsp>.

### What Happens When an Agent Starts Up

An HQ Agent needs to know how to connect to the HQ Server, and the HQ Server needs to know how to connect to the HQ Agent — each component needs the IP address and listen port (and other connection properties) to use to establish a connection with the other. You must provide this information the first time an agent starts up after installation.

There are two ways to provide the agent-server connection properties:

- In `agent.properties` — Before starting the agent for the first time, configure the HQ Server connection properties in the `agent.properties` for the agent, as described in [Section 2.2.2, “Configure Agent - Server Communication in Properties File”](#). When you start the agent, it will read the connection data from `agent.properties`. This configuration method is preferable if you have many agents to install, as it speeds the process and reduces the chance of error.
- Interactively — If you do not configure the HQ Server connection properties in the `agent.properties` for the agent, upon startup, the agent prompts for the properties in the command shell, as described in [Section 2.2.1, “Configure Agent - Server Communication Interactively”](#).

When you start an HQ Agent:

1. The agent checks to see if there is a `/data` directory that contains the HQ Server's connection properties.

- At first startup, the `/data` directory will not exist — it is created after the first successful connection between agent and server is established.
  - If the agent finds the connection properties, it tries to connect to the HQ Server - see step 5 below.
2. If the agent did not find the `/data` directory, it looks for an `agent.properties` file in a hidden directory named `.hq` in the home directory of the user that runs the agent.
    - This directory will not exist unless you have previously created it. This location for the properties file is supported to ensure configuration data is not lost in the event that the complete agent installation is overwritten in an upgrade.
    - If the agent finds the connection properties, it tries to connect to the HQ Server - see step 5 below.
  3. If the agent did not find the agent-server connection properties in an `agent.properties` file in the hidden `/.hq` directory, it looks at the `agent.properties` file in its `/conf` directory.
    - If the agent finds the connection properties, it tries to connect to the HQ Server - see step 5 below.
  4. If the agent did not find connection properties in the `{{agent.properties}}` file in its `/conf` directory, it prompts for the properties to be supplied interactively in the command shell.
  5. Upon obtaining the agent-server connection properties, the agent attempts to connect to the HQ Server.
  6. Once communications between agent and server have been successfully established:
    - The agent saves the HQ Server connection settings in `AgentHome/data`.
    - The HQ Server saves the HQ Agent's connection settings in the HQ database.

## Automatic Restart Behavior

As described above, JSW process periodically pings the agent JVM process.

If the JVM process exits with a non-zero exit code, indicating the JVM is hung or crashed, the JSW will restart the JVM process. Messages in the Agent and JSW logs indicate that the JSW has restarted the Agent. If the JSW cannot restart the JVM process, it will try again, up to a total of five times. You can configure different actions for the JSW, including the action it takes upon a particular JVM process exit code, and the number of times that it will attempt to restart the process.

## 2.1.4. Agent Configuration

The paragraphs that follow provide useful information about agent configuration data - how it is initially supplied, where it is saved, and how to change it.

- [Agent Startup Configuration Data](#)
- [Supported Locations for agent.properties](#)
- [How to Change Agent Setup Configuration Properties](#)

### Agent Startup Configuration Data

The configuration choices you must supply for the agent to start up specify where and how it communicates with the Hyperic Server, and vice versa. As described in [What Happens When an Agent Starts Up. you can supply these setup values either interactively or in the agent's properties file.

Note that upon successful startup, for future reference, the agent saves the server connection data in `AgentHome/data` and the Hyperic Server stores the agent's connection data in the Hyperic database. The agent creates the `/data` directory upon first successful startup. On subsequent startups, the agent will look at the connection data stored in its `/data` directory to determine where and how to connect to the Hyperic Server.

The agent properties that govern communications are the ones that are absolutely required for an agent to start up. In addition, there are a number of other agent properties that you can use to configure optional agent features and behaviors. Unlike the setup properties, which the agent obtains from its data directory and the Hyperic database, the properties that control optional agent behaviors are persisted only in `agent.properties`.

For a complete list of agent properties, see [Section 2.10, "Agent Properties"](#). The setup properties related to communications with the Hyperic server are those whose name starts with "agent.setup"; see [Communication Properties Reference](#).

### Supported Locations for agent.properties

`agent.properties` is installed in `AgentHome/conf`.

Note however, that agent first honors an external (from the agent installation) location for the properties file: an `.hq` directory under the home directory of the user under which the agent runs. If that directory does not exist, you can create it. (Under Windows, you can only create directory names with a leading period (.) in the DOS window (`mkdir`}).

Storing `agent.properties` external to the agent installation directory is useful, because some upgrade scenarios will overwrite the `/conf` directory. Specifically, upgrading an agent by installing a full agent package will overwrite your previous agent installation. This is relevant when you first upgrade an agent from 3.2.x or 3.1.x to 4.x, or if you choose to upgrade a 4.x agent to a later version by installing a full agent package. In these cases, if you don't keep the properties file in the `.hq/` directory, back it up prior to upgrade, and restore it after upgrade.

Once you have an operational 4.x agent installation, you can upgrade it by installing an agent bundle only - you can perform this upgrade operation from the Hyperic user interface or using a manual procedure, if you prefer. When you upgrade the agent bundle only, the agent's `/conf` directory is not updated---this method preserves the `agent.properties` file within the agent installation through the upgrade.

### How to Change Agent Setup Configuration Properties

As described in [Section 2.1.3, "Agent Launcher and Agent Startup"](#), you configure an agent's setup properties - the properties that begin with "agent.setup" - the first time you start it up. The properties that relate to where



and how to contact the Hyperic server, are saved in the agent's /data directory; those related to how the server can reach the agent are stored in the Hyperic database.

If you need to make changes to those configuration values for an agent at a later time, you can delete the agent's /data directory, edit the agent.setup.\* properties in the agent.properties file, and restart the agent. You *must* use this method if you wish to change the agent's listen port--you cannot change port settings without restarting the agent.

If you prefer to change the startup configuration (other than agent listen port) without having to restart the agent, run the following command in a command shell, while the agent is running.

```
AgentHome/hq-agent.sh setup
```

This will allow you to interactively supply new values for the setup properties, as described in [Section 2.2.1, “Configure Agent - Server Communication Interactively”](#). This will cause all properties in the properties file to be re-read, and take effect.

## 2.1.5. Agent Directory Structure

The structure of the Hyperic Agent installation directory is shown below.

```
agent-4.x.y
  bin
  bundles
    agent-4.x.y-nnnn
  conf
  data
  log
  wrapper
  lib
  sbin
```

The root of the directory structure shown above as `agent-4.x.y` is typically referred to in Hyperic documentation as the *AgentHome* directory. Key agent directories include:

- `AgentHome/bin` - contains agent start scripts: `hq-agent.sh` and `hq-agent.bat`
- `AgentHome/conf` - contains `agent.properties`
- `AgentHome/data` - this is the directory where the agent saves the connection properties it uses to contact the Hyperic Server.
- `AgentHome/log` - contains:
  - `agent.log`
  - `agent.startup.log`
  - `wrapper.log`

## 2.2. Configure Agent - Server Communication

These topics have instructions for configuring agent-server communications in `agent.properties` or interactively at first agent startup, and instructions for two vFabric Hyperic options: unidirectional and proxied communications.

- [Section 2.2.1, “Configure Agent - Server Communication Interactively”](#)
- [Section 2.2.2, “Configure Agent - Server Communication in Properties File”](#)
- [Section 2.2.3, “Configure Proxy Server for Agent - Server Communication”](#)
- [Section 2.2.4, “Configure Unidirectional Agent - Server Communication”](#)

## 2.2.1. Configure Agent - Server Communication Interactively

These are instructions for supplying agent configuration settings interactively the first time you start the agent.

### Agent can be configured in agent.properties file

Instead of configuring agent - server configuration properties interactively, you can specify them in the `agent.properties` file, as described in [Section 2.2.2, “Configure Agent - Server Communication in Properties File”](#) — preferable when you deploy a large number of agents.

**Note:** You **must** configure an HQ Agent that will manage VMware vSphere components in `agent.properties`; do not configure such agents interactively.

1. Make sure that the HQ Server to which the Agent will connect is running.
2. Start the agent from the command line under your user account.

On Windows, install the HQ Agent service, and then start it with these commands:

```
hq-agent.bat install
hq-agent.bat start
```

On Unix-based platforms, assuming you have already installed, or unpacked the agent into its installation directory, enter:

```
AgentInstallationDirectory/bin/hq-agent.sh start
```

3. When the HQ Agent starts for the first time, it issues the prompts that follow so that it can establish communications with the HQ Server.

Prompt	Notes
<i>Should Agent communications to HQ be unidirectional [default=no]</i>	This prompt only appears if you are installing HQ Enterprise. To understand this option, see <a href="#">Section 2.2.4, “Configure Unidirectional Agent - Server Communication”</a>
<i>What is the HQ server IP address</i>	Enter the listen address of your HQ Server. The server must be running. If the server is on the same machine as the agent, you can enter localhost. If there is a firewall blocking traffic from the agent to the server, specify the address of the firewall.
<i>Should Agent communications to HQ always be secure [default=no].</i>	If you want agent and server to communicate via SSL, enter yes. Otherwise, enter Return to accept the default - which is plain HTTP communications.
<i>What is the HQ server port [default=7080]</i>	This prompt appears if you accepted the default for the previous prompt. If your HQ Server is configured to listen on the default port of 7080, press Return. If there is a firewall blocking traffic from the agent to the server, configure it to forward traffic on TCP port 7080 (or 7443) to the host running the HQ Server.
<i>What is the secure HQ server port [default=7443]</i>	This prompt only appears if you answered yes to "Should Agent communications to HQ always be secure" prompt. If your HQ Server is configured to lis-

Prompt	Notes
	ten for SSL communications on the default port of 7443, press Return.
<i>What is your HQ login [default=hqadmin]:</i>	By default, the HQ server is initially configured with an administrative account with username hqadmin. Unless you have configured a different HQ user account for agent-server communications, accept the default.
<i>What is your HQ password</i>	Enter the password for the username you supplied at the previous prompt. If you accepted the default admin user password when installing the HQ server, the password is hqadmin.
<i>What IP should HQ use to contact the agent [default=n.n.n.n]</i>	The prompt will show default that is the IP address the agent detected on the host. If there is another IP address on the host you prefer to use, enter it. If there is a firewall blocking traffic from the server to the agent, enter the IP address of the firewall, and configure the firewall to forward traffic intended for the HQ Agent to the listen address of the agent host.
<i>What port should HQ use to contact the agent [default=2144]</i>	Enter the agent port the HQ Server should use when it initiates contact with the agent. The value you supply to this prompt should be the port that the agent binds to at startup, which by default is 2144. <b>Note:</b> If you have previously edited agent.properties to explicitly define a different listen port, using the optional agent.listenPort property, that is the value you should supply to this prompt. If there is a firewall blocking traffic from the server to the agent, configure the device to forward traffic on TCP port 2144 to the HQ Agent.

Messages similar to the following are displayed upon successful startup of an agent.

```
Received temporary auth token from agent
Registering agent with HQ
HQ gave us the following agent token
1215038691323-8570363106994871928-8259195015465958356
Informing agent of new HQ server
Validating
Successfully setup agent
```

## 2.2.2. Configure Agent - Server Communication in Properties File

Topics marked with\* relate to features available only in vFabric Hyperic.

- [File-Based Configuration of Agent-Server Communications is Efficient](#)
- [Procedure: Configure Agent-Server Communication Properties](#)
  - [Step 1: Open or create agent.properties](#)
  - [Step 2: Uncomment Agent-Server Communication Properties](#)
  - [Step 3: Define Communication Properties in agent.properties File](#)
  - [Step 4 - Configure Unidirectional Communications \(Optional\)](#)
  - [Step 5 - Configure Additional Agent Behaviors \(Optional\)](#)
  - [Step 6 - Copy agent.properties to Agent Installation](#)
- [Communication Properties Reference](#)

### File-Based Configuration of Agent-Server Communications is Efficient

It is simple to configure a single Hyperic Agent interactively: start it up, and respond to the prompts for locations, ports, and the other properties required for the Hyperic Agent and Hyperic Server to communicate.

However, if you have multiple Hyperic Agents to deploy, it is more efficient to set the values for the communication properties in properties files. For example, you can create a standard agent profile that you can copy to the agent installation, or to a location available to the agent installation.

Note also that Hyperic Agents that manage VMware vSphere components **must** be configured using in `agent.properties` - interactive configuration is not supported for such agents.

In a standard `agent.profile` - one that you can deploy to multiple agents that report to the same Hyperic Server, you do not edit the properties that specify an agent's listen address and port. At first startup, if explicit values for IP address and port are not set, the Hyperic Agent - which detects the network interfaces on the platform - uses the first detected interface as its listen address, and port 2144 or 2443 as its listen port, depending on whether you configure the agent for plain text or SSL communications.

Whether you use a standard agent profile, or create a unique `agent.properties` file for each agent, be sure to copy the file into the `AgentHome/conf` directory, or to the hidden `HqUserHome/.hq` directory, if it exists. (The agent will honor the settings in an `agent.properties` in the Hyperic users home directory over those found in a properties file in `AgentHome/conf`.)

### Procedure: Configure Agent-Server Communication Properties

#### Step 1: Open or create agent.properties

Make a copy of the `agent.properties` file from the agent installation.

#### Step 2: Uncomment Agent-Server Communication Properties

In the `agent.properties` file, find the section excerpted below, and remove the hash mark (#) in front of each the properties shown at the end of the excerpt.

```
## Use the following if you'd like to have the agent setup
```

```
## automatically from these properties. The values for these
## properties are used to answer the setup questions
##
## If any of these properties are left undefined, the setup
## process will prompt for their values
##
## If the value that should be used is the default when interactive
## setup is done, use the string *default* as the value for the option

#agent.setup.camIP=localhost
#agent.setup.camPort=7080
#agent.setup.camSSLPort=7443
#agent.setup.camSecure=yes
#agent.setup.camLogin=hqadmin
#agent.setup.camPword=hqadmin
#agent.setup.agentIP=*default*
#agent.setup.agentPort=*default*
#agent.setup.resetupTokens=no
```

**Note:** For more information about the properties above, see [Communication Properties Reference](#) below.

### Step 3: Define Communication Properties in agent.properties File

The `agent.properties` file contains properties you can configure to govern both agent-initiated and server-initiated communication.

- Specify the location and credentials the agent should use to contact the Hyperic Server with these properties:
  - `agent.setup.camIP` — Specify the address or hostname of the Hyperic Server.
  - `agent.setup.camPort` — The default value is the standard plaintext Hyperic listen port. You can specify a different port — as long as it is not already in use.
  - `agent.setup.camSSLPort` — The default value is the standard SSL Hyperic listen port. You can specify a different port — as long as it is not already in use.
  - `agent.setup.camSecure` — The default value is "yes" (use SSL). Change to "no" if you do not require the agent to use secure communications when contacting the Hyperic Server.
  - `agent.setup.camLogin` — Specify the username the agent should use when connecting to the server. If you change the value from the default value ("hqadmin"), make sure that that user account is properly configured on the Hyperic Server.
  - `agent.setup.camPword` — Specify the password the agent should use, along with the username above, when connecting to the server. Make sure that the password is the one configured in Hyperic for the user account.
- Specify the address or hostname and the listen port the Hyperic Server should use to contact the Hyperic Agent with these properties:
  - `agent.setup.agentIP` — If you leave the default setting — "**default**" — the Hyperic Agent will detect an IP address on the platform and choose it as its listen address.
  - `agent.setup.agentPort` — If you leave the default setting — "**default**" — the Hyperic Agent will use the default listen port (either 7080 or 7443) as its listen address. If that port is unavailable, the agent will detect a free port and choose it as its listen port.

These are the minimum properties required for agent-server communication.

Unless you want to configure other agent behaviors, save your changes and proceed to [Step 6 - Copy agent.properties to Agent Installation](#)

#### Step 4 - Configure Unidirectional Communications (Optional)

By default, agent-server communication is bi-directional. If your policies dictate that all communication between agent and server are agent-initiated, you can uncomment the `agent.setup.unidirectional` property and set it to "yes".

Unless you want to configure other agent behaviors, save your changes and proceed to [Step 6 - Copy agent.properties to Agent Installation](#).

#### Step 5 - Configure Additional Agent Behaviors (Optional)

As desired, you can configure additional agent behaviors in the `agent.properties` file. For information about configurable agent behaviors, see:

- [Section 2.4, "Configure Auto-Discovery Scanning and Reporting"](#)
- [Section 2.6, "Configure Plugin Loading"](#)
- [Section 2.5, "Configure Agent Logging"](#)
- [Section 2.8, "Tweak the Agent to Enable a Resource Plugin"](#)

After completing your edits, save your changes and copy the updated properties file to desired agent or agents, as described in the next section.

#### Step 6 - Copy agent.properties to Agent Installation

### Communication Properties Reference

#### **agent.setup.camIP**

##### **Description**

You can use this property to define for the agent the IP address of the HQ server. The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other `agent.setup.*` properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

The value can be provided as an IP address or a fully qualified domain name. To identify an server on the same host as the server, set the value to 127.0.0.1.

If there is a firewall between the agent and server, specify the address of the firewall, and configure the firewall to forward traffic on port 7080, or 7443 if you use the SSL port, to the HQ Server.

##### **Default**

Commented out, localhost.

#### **agent.setup.camPort**

##### **Description**

You can use this property to define for the agent, at first startup after installation, what server port to use for non-secure communications with the server. The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other `agent.setup.*` properties is a way to reduce the user interaction required to configure an agent to communicate with the server.



**Default**

Commented out, 7080.

**agent.setup.camSSLPort****Description**

You can use this property to define for the agent, at first startup after installation, what server port to use for SSL communications with the server. The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.\* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

**Default**

Commented out, 7443

**agent.setup.camSecure****Description**

You can use this property to define for the agent, at first startup after installation, whether to communicate with the server over SSL. If you set this property to yes, all agent-server communications will be use the SSL secure port.

If acceptable in your environment, non-SSL communication offers improved performance for agent-server communications.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.\* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

**Default**

Commented out, value of yes

**agent.setup.camLogin****Description**

You can use this property to define for the agent, at first startup after installation, the HQ username to use when registering itself with the server. The permission required on the server for this initialization is Create, for Platforms.

A login from the agent to the server is only required during the initial configuration of the agent.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.\* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

**Default**

Commented out, hqadmin

**agent.setup.camPword****Description**

You can use this property to define for the agent, at first startup after installation, the password for the user specified by `agent.setup.camLogin`.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other `agent.setup.*` properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

**Default**

Commented out, `hqadmin`.

**agent.setup.agentIP****Description**

This specifies the IP address that the HQ Server will use to contact the HQ Agent. If the agent is on the same host as the server, value of `127.0.0.1` is valid.

If there is a firewall between the server and agent, specify the IP address of the firewall, and configure the firewall to forward traffic intended for the agent to the agent's listen address, which can be configured with `agent.listenIP`.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other `agent.setup.*` properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

**Default**

As installed, `agent.properties` contains a commented out statement that sets the value to `*default*`. If you use the `agent.setup.*` properties to supply an agent's configuration at first startup, and uncomment this property and leave the value `*default*`, the HQ Server will contact the agent using the IP address that SIGAR detects on the agent host.

**agent.setup.agentPort****Description**

This specifies the port (on the IP address configured with `agent.setup.agentIP`) on the agent on which the HQ Server will communication with the agent.

If there is a firewall between the agent and the server, set `agent.setup.agentPort` to the appropriate port on the firewall, and configure the firewall to forward traffic intended for the agent to the agent listen port, which can be configured with.

The agent reads this value only in the event that it cannot find its connection configuration in its data directory. Specifying this and other `agent.setup.*` properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

**Default**

As installed, `agent.properties` contains a commented out statement that sets the value to `*default*`. If you use the `agent.setup.*` properties to supply an agent's configuration at first startup, and uncomment this property and leave the value `*default*`, the HQ Server will contact the agent on port 2144, unless SIGAR detects it is not available, in which case another default is selected.

**agent.setup.resetupToken****Description**

You can use this property to define for the agent, at first startup after installation, whether the agent will create a new token to use to authenticate with the server each time it starts up. Regenerating a token is useful if the Agent cannot connect to the server because the token has been deleted or corrupted.

Regardless of the value of this property, an agent will generate a token the first time it is started after installation.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.\* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

**Default**

As installed, agent.properties contains a commented out statement that sets the value to no.

**agent.setup.unidirectional**

Available only in **HQ Enterprise**

**Description**

Enables the unidirectional communications between HQ Agent and HQ Server, in HQ Enterprise. For more information, see [Section 2.2.4, “Configure Unidirectional Agent - Server Communication”](#).

**Default**

Commented out, defaults to no.

## 2.2.3. Configure Proxy Server for Agent - Server Communication

Available only in **vFabric Hyperic**

This page lists the agent properties for configuring the Hyperic Agent to connect to the Hyperic Server via a proxy server.

### agent.proxyHost

**Description**

The host name or IP address of the proxy server that the agent must connect to first when establishing a connection to the HQ server. Supported in HQ Enterprise only.

**Default**

none

### agent.proxyHost

**Description**

The port number of the proxy server that the agent must connect to first when establishing a connection to the HQ server. Supported in HQ Enterprise only.

**Default**

none

## 2.2.4. Configure Unidirectional Agent - Server Communication

Available only in **vFabric Hyperic**

If your security policies dictate, you can configure the agent to initiate all communications with the HQ Server. You can configure unidirectional communications at first startup. Unidirectional communications are always via SSL.

This section has instructions for changing agent communications from bidirectional to unidirectional and vice versa after the agent has already been configured.

- [Changing from Bidirectional to Unidirectional Communications](#)
- [Changing from Unidirectional to Bidirectional Communications](#)

### Changing from Bidirectional to Unidirectional Communications

1. Stop the agent.
2. Remove the agent's `\data` directory.
  - Removing the `\data` directory will cause the agent, at next startup, to look for the startup settings it needs to connect to the HQ Server in its `agent.properties` file; if the properties file doesn't contain them, it will prompt for settings in the shell.
3. Configure the agent for unidirectional communications using one of these methods:
  - If your practice is to provide all agent startup properties in the properties file, edit `agent.properties` to set `agent.setup.unidirectional=yes`, and start the agent.
  - If your practice is to configure the agent startup properties interactively, start the agent, and respond "yes" when asked if the agent should be configured for unidirectional communications.
4. In the HQ user interface, navigate to the platform's Inventory tab and click **Edit** in the "Type & Network Properties" section.
  - In the edit view for "Type & Network Properties", the "Agent Connection" drop-down list will show your currently selected port for bidirectional communications, something like `10.2.0.213:2144`, where `10.2.0.213` is the IP address of the platform, and `2144` is the bidirectional port number previously used.
5. Expand the drop-down list and select the entry that shows the same IP address, and "-1" as the port:
6. `10.2.0.213:-1`
  - Your agent will now use unidirectional communications.

### Changing from Unidirectional to Bidirectional Communications

1. Stop the agent.
2. Remove the agent's `data` directory.
  - Removing the `data` directory will cause the agent, at next startup, to look for the startup settings it needs to connect to the HQ Server in its `agent.properties` file; if the properties file doesn't contain them, it will prompt for settings in the shell.

3. Configure the agent for bidirectional communications using one of these methods:
  - If your practice is to provide all agent startup properties in the properties file, edit `agent.properties` to set `agent.setup.unidirectional=no`, and start the agent.
  - If your practice is to configure the agent startup properties interactively, start the agent, and when prompted for communications direction, respond "no" when asked if the agent should be configured to run in unidirectional mode.
4. In the HQ user interface, navigate to the platform's Inventory tab and click Edit in the "Type & Network Properties" selection.
5. Select the appropriate agent in the "Agent Connection" drop down.
  - In the edit view for "Type & Network Properties", the "Agent Connection" drop-down list will show your currently selected port for unidirectional communications, something like 10.2.0.213:-1, where 10.2.0.213 is the IP address of the platform, and -1 is the port number.
6. Expand the drop-down list and select the entry that shows the same IP address, and "2144" as the port (or the port you are configured to use, if not the default), for example, 10.2.0.213:2144
  - Your agent will now use bidirectional communications.

## 2.3. Start, Stop, and Other Agent Operations

Topics marked with\* relate to features available only in vFabric Hyperic.

- [Section 2.3.1, “Run the Agent Launcher from the Command Line”](#)
  - [Running hq-agent.sh](#)
  - [Running hq-agent.bat](#)
- [Section 2.3.2, “Run the Agent Launcher from the Hyperic User Interface”](#)
  - [Restart an Agent from the HQ User Interface](#)
  - [Ping an Agent from the HQ User Interface](#)
  - [Upgrade an Agent from the HQ User Interface](#)
  - [Push a Resource Plugin to the Agent from the HQ User Interface](#)
- [Section 2.3.3, “Run the Agent Without the Java Service Wrapper”](#)

### 2.3.1. Run the Agent Launcher from the Command Line

You initiate the agent launcher and agent lifecycle commands with the `hq-agent.sh`, or `hq-agent.bat` script, in the `AgentHome/bin` directory.

#### Running `hq-agent.sh`

1. Open a command shell or terminal window.
2. Enter a command of this form:

```
sh hq-agent.sh command
```

Where `command` is one of the following:

- `start` — Starts the agent as a daemon process.
- `stop` — Stops the agent's JVM process.
- `restart` — Stops and then starts the agent's JVM process
- `status` — Queries the status of the agent's JVM process.
- `dump` — Runs a thread dump for the agent process, and writes the results to the `agent.log` file in `AgentHome/log`.
- `ping` — Pings the agent process.
- `setup` — Causes the Hyperic Agent to prompt you for the agent-server connection properties, allowing you to change the values that were provided at first agent startup.

#### Running `hq-agent.bat`

1. Open a terminal window.
2. Enter a command of this form:

```
hq-agent.bat command
```

Where `command` is one of the following:

- `start` - starts the agent as an NT service
- `stop` - stops the agent as an NT service
- `restart` - stops and then starts the agent's JVM process
- `install` - installs the agent NT service
- `remove` - removes the agent's service from the NT service table
- `query` - queries the current status of the agent NT service (status)
- `ping` - pings the agent process for availability
- `setup` - prompts for setup configuration for the agent process



## 2.3.2. Run the Agent Launcher from the Hyperic User Interface

In vFabric Hyperic, you can issue selected commands to an running Hyperic Agent.

Agent control commands are available on the **Views** tab for an HQ Agent or a group of agents in inventory.

### Restart an Agent from the HQ User Interface

Available only in **vFabric Hyperic**

The **restart** command invokes the agent's Java Service Wrapper's restart command. The restart command shuts down the JVM process in which the agent runs, waits for the process to terminate cleanly, and spawns a new JVM process for the agent. During the restart process, the agent's metric collection and resource control functionality will be interrupted.

The restart command happens asynchronously. To verify that the restart succeeded you can go to the page for the agent in the HQ Portal and check its availability. Alternatively, you could configure an alert for the agent that fires when the agent's availability changes.

To restart an agent:

1. Navigate to the **Views** tab for an HQ Agent or a group of agents.
2. Select **restart** from the pull-down list
3. Click **Execute**.

### Ping an Agent from the HQ User Interface

Available only in **vFabric Hyperic**

1. Navigate to the **Views** tab for an HQ Agent or a group of agents.
2. Select **ping** from the pull-down.
3. Click **Execute**.

### Upgrade an Agent from the HQ User Interface

Available only in **vFabric Hyperic**

1. Navigate to the **Views** tab for an HQ Agent or a group of agents.
2. Select the **upgrade** command from the pull-down.
3. On the pull-down list of agent bundles that appears, select the desired bundle
4. Click **Execute**.
  - The selected agent bundle is transferred from the HQ Server to the target agent(s).
  - The agent expands the bundle locally.
  - The agent updates the local bundle property.
  - The server restarts the agent.

The configuration settings in the agent's `/conf/agent.properties` file are preserved.

## Push a Resource Plugin to the Agent from the HQ User Interface

Available only in **vFabric Hyperic**

This **push plugin** command sends new and changed resource plugins to the target agent(s). Pushing plugins to an agent results in an agent restart.

1. Navigate to the **Views** tab for an HQ Agent or a group of agents.
2. Select the **push plugin** command from the pull-down.
3. On the pull-down list of plugins that appears, select the desired plugin.
4. Click **Execute**.
  - The selected plugin is transferred from the HQ Server to the target agent(s)
  - The server restarts the agent(s).

### 2.3.3. Run the Agent Without the Java Service Wrapper

If you run an Hyperic Agent on a system that does not support the Java Service Wrapper, or for other reasons prefer not to use the wrapper, you can start the agent without the wrapper.

The `hq-agent-nowrapper.sh` agent start script in `AgentHome/bundles/agent-x.y.z/bin`

Because `hq-agent-nowrapper.sh` does not fork itself into the background, run it using `nohup`:

```
nohup AgentHome/bundles/agent-x.y.z/bin/hq-agent-nowrapper.sh &
```

## 2.4. Configure Auto-Discovery Scanning and Reporting

This page lists agent properties that control auto-inventory scanning and reporting. For more information, see the "Discover and Import Resources to Inventory" section in *Manage HQ Resource Inventory*.

### **autoinventory.defaultScan.interval.millis**

#### **Description**

Specifies how frequently the agent performs a default autoinventory scan.

The default scan detects servers and platform services, typically using the process table or the Windows registry. Default scans are less resource-intensive than runtime scans.

#### **Default**

Commented out, set to 86,400,000 milliseconds, or 1 day.

**Note however, that by default, the agent performs the default scan at startup and every 15 minutes thereafter.**

### **autoinventory.runtimeScan.interval.millis**

#### **Description**

Specifies how frequently the agent performs a runtime scan.

A runtime scan may use more resource-intensive methods to detect services than a default scan. For instance, a runtime scan may involve issuing an SQL query or looking up an MBean.

#### **Default**

86,400,000 milliseconds, or 1 day.

### **agent.eventReportBatchSize**

#### **Description**

The maximum number of events that the HQ Agent will send per contact with the server.

#### **Default**

As installed, agent.properties does not contain a line that defines the value of this property. The default behavior of the agent is to send a maximum of 100 events per contact with the server.

### **agent.maxBatchSize**

#### **Description**

The maximum number of metrics that the agent will send per contact with the server.

#### **Default**

As installed, agent.properties does not contain a line that defines the value of this property. The default behavior of the agent is to send a maximum of 500 per contact with the server.

## 2.5. Configure Agent Logging

Topics marked with\* relate to features available only in vFabric Hyperic.

- [Section 2.5.1, “Agent Log Files”](#)
- [Section 2.5.2, “Configure Agent Log Name or Location”](#)
- [Section 2.5.3, “Configure Agent Logging Level”](#)
- [Section 2.5.4, “Redirect System Messages to the Agent Log”](#)
- [Section 2.5.5, “log4j Properties”](#)

### 2.5.1. Agent Log Files

Agent log files are stored in the AgentHome/log directory. They include:

- agent.log
- agent.startup.log
- wrapper.log - The agent JSW-based launcher writes messages to this log.

### 2.5.2. Configure Agent Log Name or Location

Use these properties to change the name or location of the agent log file:

#### agent.logDir

##### Description

You can add this property to the agent.properties file to specify the directory where the HQ Agent will write its log file. Unless you specify a fully qualified path, agent.logDir is evaluated relative to the agent installation directory.

This property does not exist in agent.properties unless you explicitly add it. To change the location for the agent log file, add agent.logDir to agent.properties and enter a path relative to the agent installation directory, or if desired, a fully qualified path.

Note that the name of the agent log file is configured with the [Section 2.10.6, “agent.logFile”](#) property.

##### Default

agent.logDir does not exist in agent.properties unless you explicitly add it. The default behavior is equivalent to this setting:

```
agent.logDir=log
```

resulting in the agent log file being written to the AgentHome/log directory.

#### agent.logFile

##### Description

Specifies the path and name of the agent log file.

**Default**

Note that in `agent.properties`, the default setting for `agent.LogFile` is made up of a variable and a string.

```
agent.logFile=${agent.logDir}\agent.log
```

- `agent.logDir` is a variable - it supplies the value of an identically named agent property. By default, the value of `agent.logDir` is `log`, interpreted relative to the agent installation directory.
- `agent.log` is the name for the agent log file.

So, by default, the agent log file is named `agent.log`, and is written to the `AgentHome/log` directory.

If you want the agent to log to a different directory, you must explicitly add the [Section 2.10.5, “agent.logDir”](#) property to `agent.properties`.

## 2.5.3. Configure Agent Logging Level

Use this property to control what severity of messages that agent writes to the agent log file:

**agent.logLevel****Description**

Specifies the level of detail of the messages the Agent writes to the log file. Allowable values are: INFO and DEBUG.

**Default**

INFO

## 2.5.4. Redirect System Messages to the Agent Log

Use these properties to redirect system-generated messages to the agent log file:

**agent.logLevel.SystemErr****Description**

Redirects `System.err` to `agent.log`. Commenting out this setting will cause `System.err` to be directed to `agent.log.startup`.

**Default**

ERROR

**agent.logLevel.SystemOut****Description**

Redirects `System.out` to `agent.log`. Commenting out this setting will cause `System.out` to be directed to `agent.log.startup`.

**Default**

INFO

## 2.5.5. log4j Properties

The log4j properties in `agent.properties` are excerpted below.

### Agent log4j Properties

```
log4j.rootLogger=${agent.logLevel}, R
log4j.appender.R.File=${agent.logFile}
log4j.appender.R.MaxBackupIndex=1
log4j.appender.R.MaxFileSize=5000KB
log4j.appender.R.layout.ConversionPattern=%d %-5p [%t] [%c{1}] %m%n
log4j.appender.R.layout=org.apache.log4j.PatternLayout
log4j.appender.R=org.apache.log4j.RollingFileAppender
##
## Disable overly verbose logging
##
log4j.logger.httpclient.wire=ERROR
log4j.logger.org.apache.commons.httpclient=WARN
log4j.logger.org.hyperic.hq.measurement.agent.server.SenderThread=INFO
log4j.logger.org.hyperic.hq.agent.server.AgentDLListProvider=INFO
log4j.logger.org.hyperic.hq.agent.server.MeasurementSchedule=INFO
log4j.logger.org.hyperic.hq.measurement.agent.server.ScheduleThreadTrace=INFO
log4j.logger.org.hyperic.util.schedule.ScheduleTrace=INFO
##
## Only log errors from naming context
##
log4j.category.org.jnp.interfaces.NamingContext=ERROR
log4j.category.org.apache.axis=ERROR
```

## 2.6. Configure Plugin Loading

By default, at startup, an HQ Agent loads all of the plugins in its plugin directory - `AgentHome/bundles/agent-x.y.z-nnnn/pdk/plugins`.

You can reduce the agent's memory footprint by configuring it to load only the plugins you use. You can either specify a list of plugins to exclude, or configure a list of the plugins to load.

### **plugins.exclude**

#### **Description**

Use this property to specify plugins that you do not wish the HQ Agent to load at startup. This is useful for reducing the agents memory footprint.

#### **Usage**

Supply a comma-separated list of plugins to exclude, for example:

```
plugins.exclude=jboss,apache,mysql
```

### **plugins.include**

#### **Description**

Use this property to specify plugins that you do wish the HQ Agent to load at startup. This is useful for reducing the agents memory footprint.

#### **Usage**

Supply a comma-separated list of plugins to include, for example:

```
plugins.include=weblogic,apache
```

## 2.7. Deploying Multiple Hyperic Agents

- [Section 2.7.1, “Establish Installation Environment”](#)
  - [Set up an Install Server](#)
  - [Create Accounts and Installation Directories on Target Platforms](#)
- [Section 2.7.2, “Create Standard Agent Properties File”](#)
- [Section 2.7.3, “Perform Remote Agent Installations”](#)
  - [Install and Start Agents One-by-One](#)
  - [Deploy and Start Multiple Agents at Once](#)
- [Section 2.7.4, “Verify Successful Agent Startup”](#)

This section has recommendations for how to deploy agents in volume in a large Hyperic environment.

### 2.7.1. Establish Installation Environment

#### Set up an Install Server

Choose a machine that can access all target platforms from which to perform remote installation. We refer to this as the "install server". On the install server, create a user account, for instance "hyperic", with permissions required to SSH into each target platform without a password.

#### Create Accounts and Installation Directories on Target Platforms

On each platform to which you'll install an agent:

- Create an account that is identical to the one you created on the install server.
- Create identical installation directories, for example, `/home/hyperic`.
- On Windows systems, create an `.hq` directory in the user home directory that contains the installation directory you just created. You must open a command shell, and use the `mkdir` command to create the `.hq` directory. (This directory is created automatically when you install the Hyperic Agent on Unix-like systems.

### 2.7.2. Create Standard Agent Properties File

To enable mass agent deployment, you create an `agent.properties` that defines the agent properties required for the agent to start up and connect with the Hyperic Server. If you supply the necessary information in the properties file, each Hyperic Agent will find its setup configuration at startup, rather querying for it interactively.

At a minimum, you must define the Hyperic Server address and port. In addition, you can configure optional agent behaviors that are controlled by agent properties. For more information, see [Section 2.2.2, “Configure Agent - Server Communication in Properties File”](#).

The first time you start the agent, it will obtain the server connection information it needs to contact the server and register itself.



## 2.7.3. Perform Remote Agent Installations

### Install and Start Agents One-by-One

Follow these steps to install agents one-by-one. To install to all target platforms at once, see [Deploy and Start Multiple Agents at Once](#) below.

1. Log on to your installation account on the install server.
2. SSH to the remote platform.
3. Copy the agent archive to the agent host.
4. Unpack the agent archive.
5. Copy the `agent.properties` file to the `/.hq` directory under the home directory of the standard agent installation user account.
6. If you are upgrading an 3.x agent to 4.x, stop the 3.x agent, if necessary.
7. Start the new agent.

### Deploy and Start Multiple Agents at Once

Deploy an agent to each host listed in `hosts.txt` by

1. Create a `hosts.txt` file on your install server that maps hostname to IP address for each platform to which you wish to install the agent.
2. Open a command shell on the install server.
3. Entering the following command in the shell, supplying the correct name of the agent package in the export command:

```
$ export AGENT=hyperic-hq-agent-4.0.0-x86-linux.tgz
$ for host in `cat hosts.txt`; do scp $AGENT $host: && ssh $host "tar zxf $AGENT && ./hyperic-hq-agent-4.0.0/hq-agent.sh start"; done
```

If target hosts have sequential names (for example, `host001`, `host002`, `host003`, etc), you can skip the `hosts.txt` file and use the `seq` command like this:

```
$ export AGENT=hyperic-hq-agent-4.0.0-x86-linux.tgz
$ for i in `seq 1 9`; do scp $AGENT host$i: && ssh host$i "tar zxf $AGENT && ./hyperic-hq-agent-4.0.0/hq-agent.sh start"; done
```

## 2.7.4. Verify Successful Agent Startup

After successfully registering itself with the Hyperic Server, an Hyperic Agent runs an autoscan, and should discover its host platform, and supported servers managed products. Check the Auto-Discovery portlet in the Hyperic Dashboard to verify the the platforms were discovered.

If you have problems, see [Section 4, “Troubleshoot Agent and Server Problems”](#).

## 2.8. Tweak the Agent to Enable a Resource Plugin

These topics describe how to configure the agent to enable a particular plugin to perform one or more of its management functions:

- [Section 2.8.1, “Configure Agent Account Privileges under Solaris 10”](#)
- [Section 2.8.2, “Configure Agent HTTP Request Header”](#)
- [Section 2.8.3, “Configure Agent to Monitor JBoss”](#)
- [Section 2.8.4, “Configure Agent to Monitor WebSphere Application Server”](#)
- [Section 2.8.5, “Configure HQ Agent to Manage WebLogic Server”](#)
- [Section 2.8.6, “Configure the Data to Log for Windows Events”](#)

## 2.8.1. Configure Agent Account Privileges under Solaris 10

To auto-discover certain products under Solaris 10, the HQ Agent must run as root or you need explicitly grant additional permissions to the account where the agent runs. For background information, see the "Solving Auto-Discovery Problems" section in *Manage HQ Resource Inventory*.

Under Solaris 10's Least Privilege Model (LPM), default privileges are minimal. The HQ Agent must be able to read `/proc/$pid/` files on the platform. Problems with auto-discovery on Solaris 10 may be the result of insufficient privileges. Depending on your account privilege implementation you may need to grant the `proc_zone` privilege to the agent account.

For example, you could add this line to `/etc/user_attr`, to grant `proc_owner` privilege the `hq` user and deny the `proc_session` privilege:

```
hq::::type=normal;defaultpriv=basic,proc_owner,!proc_session
```

**Note:** After changing account privileges, the user needs to re-login.

Your approach for enabling agent access to `/proc/$pid/` files will depend on your company's LPM implementation and best practices.

For related information, see [The Least Privilege Model in the Solaris 10 OS](#) and [process privilege model](#).

## 2.8.2. Configure Agent HTTP Request Header

If you monitor a remote HTTP server, it is useful to configure the HTTP request header for agent HTTP requests.

### **http.useragent**

#### **Description**

The `http.useragent` property defines the value for the User-Agent request header in HTTP requests issued by the HQ Agent. By default, the User-Agent in agent requests includes the HQ Agent version - and so change upon agent upgrade. If a target HTTP server is configured to block requests with an unknown User-Agent, agent requests will fail after agent upgrade.

You can use `http.useragent` to define a User-Agent value that will be consistent across upgrades.

Note: `agent.properties` does not contain this property by default. You must explicitly add it to the file.

#### **Default**

`Hyperic-HQ-Agent/Version`

For example:

`Hyperic-HQ-Agent/4.1.2-EE`

### 2.8.3. Configure Agent to Monitor JBoss

**jboss.installpath****Description**

To enable the agent to monitor JBoss, specify the location of the JBoss root directory.

**Default**

/usr/local/jboss-4.0.0

## 2.8.4. Configure Agent to Monitor WebSphere Application Server

This page has agent properties related to monitoring WebSphere.

### **websphere.installpath**

#### **Description**

To enable the agent to monitor WebSphere, specify the location of the WebSphere jars.

#### **Default**

/opt/WebSphere/AppServer

### **websphere.useext**

#### **Description**

This property is required to enable management of WebSphere 6.0 and 6.1.

Do **not** define the `websphere.useext` property to monitor WebSphere 7.

#### **Usage**

Add the following property definition to the `agent.properties` file for an an HQ Agent that will manage WebSphere 6.0 or 6.1.

```
websphere.useext=true
```

## 2.8.5. Configure HQ Agent to Manage WebLogic Server

This page lists agent properties related to managing WebLogic Server.

### Specify WebLogic Server Installation Path

#### **weblogic.installpath**

##### **Description**

To enable the agent to monitor WebLogic 8.1, specify the location `server/lib/weblogic.jar`

##### **Default**

`/usr/local/bea/weblogic-8.1`

### Configure HQ Agent for Two-Way SSL with WebLogic Server

In Two-Way SSL, each side presents an SSL certificate to the other - the client must trust the server cert and the server must trust the client cert. The process is:

1. HQ Agent initiates a connection with the Administration Server.
2. The Administration server presents its certificate and asks the HQ Agent for its certificate.
3. The HQ Agent presents its certificate (*client2certs.pem*), using the private key (*clientkey.pem* and *clientkey*) to encrypt the communication.

To enable the HQ Agent to use Two-Way-SSL with a Weblogic Administration Server, you specify the client cert the HQ Agent presents to the Administration Server by adding these properties to `agent.properties`.

- `weblogic.auth.method=ssl2ways`
- `weblogic.ssl2ways.key=ClientKey.pem` (client private key)
- `weblogic.ssl2ways.key.pass=ClientKey` (passphrase for client private key)
- `weblogic.ssl2ways.cert=Client2Certs.pem` (client certificate)

Weblogic docs: [Using Two-Way SSL Authentication](#)

#### **weblogic.auth.method**

##### **Description**

`weblogic.auth.method` is one of four properties you define to enable an HQ Agent to communicate with a WebLogic Administration Server using Two-Way-SSL.

Add the following line to the `agent.properties` file to specify that the agent will use Two-Way-SSL for communications with the Administration Server.

```
weblogic.auth.method=ssl2ways
```

##### **Default**

None.

**weblogic.ssl2ways.cert****Description**

`weblogic.ssl2ways.cert` is one of four properties you define to enable an HQ Agent to communicate with a WebLogic Administration Server using Two-Way-SSL.

Add `weblogic.ssl2ways.cert` to the `agent.properties` file and set its value to the location of the client certificate that the HQ Agent will present to the Administration Server:

```
weblogic.ssl2ways.cert=Client2Cert.pem
```

where *Client2Cert.pem* is the **path to the client certificate** the HQ Agent presents to the Administration Server it manages.

**Default**

None.

**weblogic.ssl2ways.key****Description**

`weblogic.ssl2ways.cert` is one of four properties you define to enable an HQ Agent to communicate with a WebLogic Administration Server using Two-Way-SSL.

Add `weblogic.ssl2ways.cert` to the `agent.properties` file and set its value to the location of the client certificate that the HQ Agent will present to the Administration Server:

```
weblogic.ssl2ways.cert=Client2Cert.pem
```

where *Client2Cert.pem* is the **path to the client certificate** the HQ Agent presents to the Administration Server it manages.

**Default**

None.

**weblogic.ssl2ways.key.pass****Description**

`weblogic.ssl2ways.key.pass` is one of four properties you define to enable an HQ Agent to communicate with a WebLogic Administration Server using Two-Way-SSL.

Add `weblogic.ssl2ways.key.pass` to the `agent.properties` file and set its value to the passphrase for the client private key:

```
weblogic.ssl2ways.key.pass=ClientKey
```

where *ClientKey* is the passphrase for the client private key.

**Default**

None.



## 2.8.6. Configure the Data to Log for Windows Events

This page describes the use of the `platform.log_track.eventfmt` agent property to customize the content of events that the HQ Agent logs for Windows Events, when log tracking is enabled for a Windows resource.

### **platform.log\_track.eventfmt**

#### **Description**

Specifies the content and format of the Windows event attributes that an HQ Agent includes when logging a Windows event as an event in HQ. `agent.properties` does not contain the `platform.log_track.eventfmt` property, you must explicitly add it if you want to tailor the data logged for Windows events.

#### **Default Behavior**

When Windows log tracking is enabled, an entry of this form is logged for events that match the criteria you specified on the resource's **Configuration Properties** page:

[Timestamp] Log Message (EventLogName):EventLogName:EventAttributes

where:

- **Timestamp** - is when the event occurred
- **Log Message** - is an text string
- **EventLogName** - is the Windows event log type, "System", "Security", or "Application".
- **EventAttributes** - a colon delimited string made of the Windows event **Source** and **Message** attributes.

For example, this log entry:

04/19/2010 06:06 AM Log Message (SYSTEM): SYSTEM: Print: Printer HP LaserJet 6P was paused.

is for an Windows event written to the Windows System event log at 6:06 AM on 04/19/2010. The Windows event **Source** and **Message** attributes, are "Print" and "Printer HP LaserJet 6P was paused.", respectively.

#### **Configuration**

You can use the parameters below to configure the Windows event attributes that the agent writes for a Windows event. Each parameter maps to Windows event attribute of the same name.

- `%user%` - The name of the user on whose behalf the event occurred.
- `%computer%` - The name of the computer on which the event occurred.
- `%source%` - The software that logged the Windows event.
- `%event%` - A number identifying the particular event type.
- `%message%` - The event message.
- `%category%` - An application-specific value used for grouping events.

For example, with this property setting:

```
platform.log_track.eventfmt=%user%%computer% %source%:%event%:%message%
```

the HQ Agent will write the following data when logging Windows event:

```
04/19/2010 06:06 AM Log Message (SYSTEM): SYSTEM: HP_Administrator@Office
Print:7:Printer HP LaserJet 6P was paused.
```

This entry is for as for an Windows event written to the Windows System event log at 6:06 AM on 04/19/2010. The software associated with the event was running as "HP\_Administrator" on the host "Office". The Windows event's **Source**, **Event**, and **Message** attributes, are "Print", "7", and "Printer HP LaserJet 6P was paused.", respectively.

After you configure the content of the log entry written for a Windows event, when you configure an alert definition for a Windows resource, you can create an alert condition based on the message content, including the custom fields you have configured. For more information, see the "Define Alert Condition Set" section in *Configure Monitoring Options*.

## 2.9. Manage the Hyperic Agent

These topics have information about monitoring and tuning the Hyperic Agent.

- [Section 2.9.1, “View Status of all Hyperic Agents”](#)
- [Section 2.9.2, “View Hyperic Agent Metrics”](#)
- [Section 2.9.3, “Reduce Agent Memory Footprint”](#)

## 2.9.1. View Status of all Hyperic Agents

Topics marked with\*relate to features available only in vFabric Hyperic.

The page has information about the **Agents** tab of the **HQ Health** page — a screenshot, and definitions of the health data for an agent.

- [Agents Tab in HQ Health](#)
- [Health Data for an Agent](#)

For information about using **HQ Health** to troubleshoot problems, see [Section 4, “Troubleshoot Agent and Server Problems”](#).

### Agents Tab in HQ Health

The **Agents** tab of the **HQ Health** page — shown in the screenshot below — shows the status of all of the Hyperic Agents that are registered with the Hyperic Server.

To navigate to **HQ Health**, click the link for it in the **Plugins** section of the **Administration** page.

The screenshot displays the **HQ Health** interface. At the top, there are four summary boxes: **System Load Average** (1min: 0.16, 5min: 0.10, 15min: 0.08), **System Memory Stats** (Total: 5.8 GB, Used: 2.0 GB, Free: 3.9 GB), **HQ Process Information** (PID: 20375, Open FDs: 411, Process Start Time: Sep 23, 2010 3:21:24 PM, Size: 1.4 GB, Resident: 956.6 MB, Shared: 16.6 MB, CPU: 0%), and **JVM Memory** (% Used: 57, Free: 203.2 MB, Total Allocated: 481.2 MB, Max Allocation: 481.2 MB). Below these is the **System Processor Stats** box (User: 0%, System: 0%, Nice: 0%, Idle: 99%, Wait: 0%) and the **System Swap Stats** box (Total: 4.0 GB, Used: 92.0 KB, Free: 4.0 GB). A **Print** button is located under the HQ Process Information box.

The main section is the **Agents** tab, which contains a table of registered agents. The table has the following columns: FQDN, Address, Port, Version, Build #, Bundle Version, Creation Time, # Platforms, # Metrics, Time Offset (ms), and License Count. The table lists 15 agents, including vmc-ssrc-rh47.eng.vmware.com, win2003std.vmware.com, vmc-ssrc-2k810.s2qa.com, DSL-10.16.16.142, DSL-10.16.17.9, DSL-10.16.17.44, DSL-10.16.17.35, vmc-ssrc-2k332.s2qa.com, vmc-ssrc-2k816.s2qa.com, vmc-ssrc-rh17.eng.vmware.com, hubble.eng.vmware.com, vmlin-was-005.intranet.hyperic.net, and vmlin-was-02.intranet.hyperic.net.

FQDN	Address	Port	Version	Build #	Bundle Version	Creation Time	# Platforms	# Metrics	Time Offset (ms)	License Count
vmc-ssrc-rh47.eng.vmware.com	10.150.29.175	2144	4.4.0-EE	1509	agent-4.4.0-EE-1509	7/28/10 8:09 PM	1	328	?	1
win2003std.vmware.com	10.16.17.36	4177	4.4.0-EE	1509	agent-4.4.0-EE-1509	7/28/10 8:11 PM	12	139	874.0	2
vmc-ssrc-2k810.s2qa.com	10.150.29.72	2177	4.4.0-EE	1509	agent-4.4.0-EE-1509	7/28/10 8:17 PM	10	136	153.0	2
DSL-10.16.16.142	10.16.16.142	2144	4.3.0-EE	1381	agent-4.3.0-EE-1381	7/29/10 2:02 PM	1	45	?	1
DSL-10.16.17.9	10.16.17.9	2144	4.3.0-EE	1381	agent-4.3.0-EE-1381	7/29/10 2:26 PM	1	45	?	1
DSL-10.16.17.44	10.16.17.44	2144	4.3.0-EE	1381	agent-4.3.0-EE-1381	7/29/10 2:40 PM	1	45	?	1
DSL-10.16.17.35	10.16.17.35	2144	4.3.0-EE	1381	agent-4.3.0-EE-1381	7/29/10 2:45 PM	1	45	?	1
vmc-ssrc-2k332.s2qa.com	10.150.29.204	2175	4.4.0	1509	agent-4.4.0-1509	7/30/10 12:00 PM	1	87	646.0	1
vmc-ssrc-2k816.s2qa.com	10.150.29.103	2175	4.4.0-EE	1509	agent-4.4.0-EE-1509	7/30/10 1:51 PM	1	35	684.0	1
vmc-ssrc-rh17.eng.vmware.com	10.150.29.94	2175	4.4.0-EE	1464	agent-4.4.0-EE-1464	8/16/10 3:54 PM	1	82	2.0	1
hubble.eng.vmware.com	10.17.143.3	3309	4.3.0-EE	1433	agent-4.3.0-EE-1433	9/16/10 4:49 PM	1	95	?	1
vmlin-was-005.intranet.hyperic.net	10.0.0.124	2175	4.4.0-EE	1509	agent-4.4.0-EE-1509	9/22/10 12:46 PM	1	176	995.0	1
vmlin-was-02.intranet.hyperic.net	10.0.0.233	2233	4.4.0-EE	1509	agent-4.4.0-EE-1509	9/22/10 3:36 PM	1	58	33.0	1

## Health Data for an Agent

The table below defines the information on the **Agents** tab of **HQ Health**,

Field	Description	Notes
FQDN	Fully-qualified domain name of the platform where the agent runs.	
Address	IP address of the platform where the agent runs.	
Port	Port where the agent listens for communication with the Hyperic Server.	If you configure unidirectional agent - server communications, the agent initiates all communications with the Hyperic Server.
Version	Agent version number.	Although an agent might work successfully with an Hyperic Server of a later version, it is strongly recommended that you run the same version of the agent and server.
Build #	Agent build number	
Bundle Version	Agent bundle version	
Creation Time	The date/time that the Hyperic Agent was first started up.	
# Platforms	Number of platforms the agent manages.	Typically, an agent manages one platform - the platform where it runs. Exceptions include: <ul style="list-style-type: none"> <li>• If the agent manages a vSphere vCenter instance, the number of platforms shown is the number of VMs the vCenter server manages.</li> <li>• If the agent manages remote network devices or network host platform types.</li> </ul>
# Metrics	The number of resource metrics the agent collects. This is the total number of metrics that are configured for collection across all resources the agent monitors.	If one agent bears an inordinate metric load, you may be able to distribute it more evenly. See the "Overloaded Agent" section of <a href="#">Section 4, "Troubleshoot Agent and Server Problems"</a> .
Time Offset (ms)	The difference in system clock time between the agent and the Hyperic Server.	A time offset can cause incorrect availability reporting. See the "Out-of-Sync Agent and Server Clocks" section of <a href="#">Section 4, "Troubleshoot Agent and Server Problems"</a> . A question mark in this column indicates that the Hyperic Server is unable to contact the agent. <b>True?</b>

Field	Description	Notes
License Count	The number of platform licenses consumed by the agent.	Typically, a single agent consumes a single license. If an agent manages a vSphere vCenter instance, it consumes a license for the platform that hosts vCenter, a license for each vSphere vHost administered by the vCenter instance, and — if an agent is installed in each VM — a license for each vSphere VM on each vHost.

## 2.9.2. View Hyperic Agent Metrics

Topics marked with\*relate to features available only in vFabric Hyperic.

The Hyperic Agent monitors itself. You can tailor the metric collection settings for an Hyperic Agent, use agent metrics to troubleshoot problems, and base alerts on agent metrics or events. This page describes the metrics and monitoring views for an Hyperic Agent.

- [Agent Monitoring Defaults](#)
- [View Agent Indicators Charts](#)
- [View Agent Metric Data](#)
- [Hyperic Agent Metrics](#)

For information about metrics that indicate Hyperic Agent problems, see [Section 4, “Troubleshoot Agent and Server Problems”](#).

### Agent Monitoring Defaults

The metrics that the agent reports for itself by default are:

- Availability
- JVM Free Memory
- JVM Total Memory
- Number of Metrics Collected Per Minute
- Number of Metrics Sent to the Server Per Minute
- Server Offset
- Total Time Spend Fetching Metrics per Minute

See the [Hyperic Agent Metrics](#) section for a list of all supported Hyperic Agent metrics.

### View Agent Indicators Charts

The **Indicators** page for an Hyperic Agent charts the agent's indicator metrics, by default:

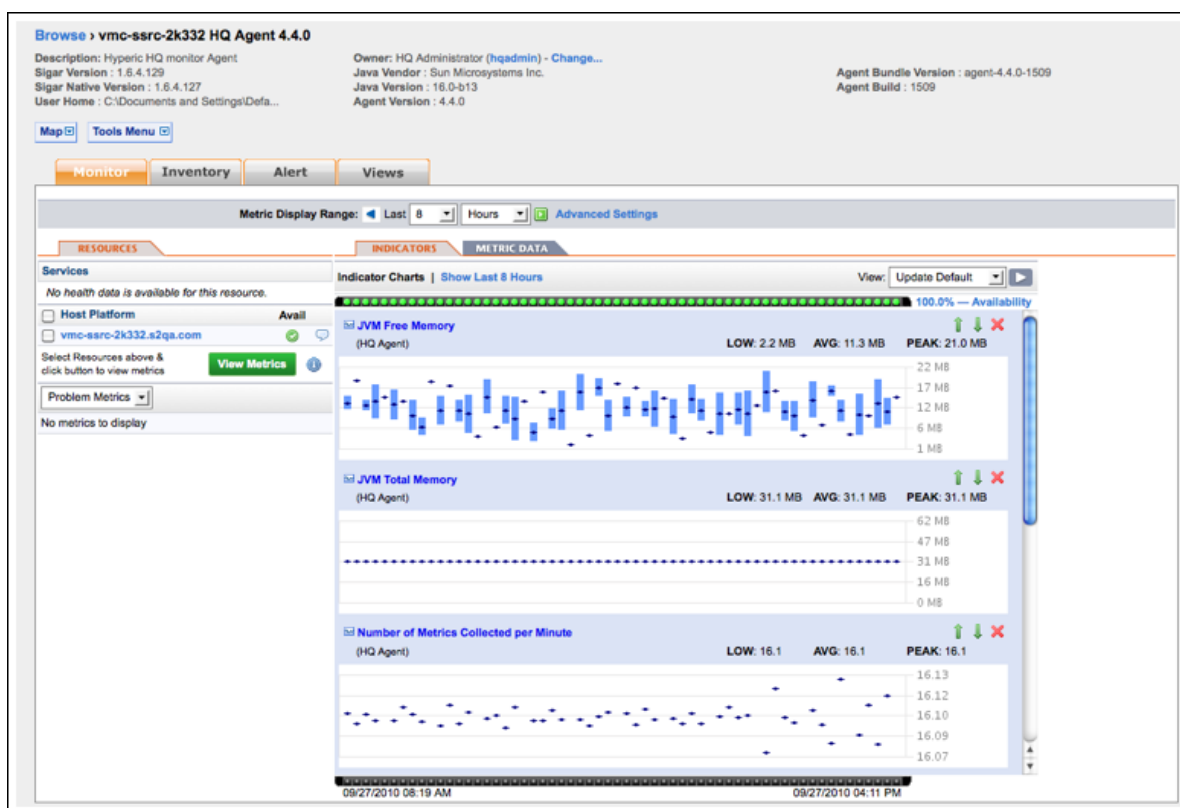
- JVM Free Memory
- JVM Total Memory
- Number of Metrics Collected Per Minute

To view the **Indicators** page for an Hyperic Agent:

1. Click **Resources > Browse**.
2. Click **Servers**.

3. Select **HQ Agent** from the **Server Type** pull-down.

The screenshot below is the **Indicators** page for an Hyperic Agent.



## View Agent Metric Data

The **Metric Data** page for an Hyperic Agent displays all of the metrics collected for the agent in tabular form.

To view the **Metric Data** page for an Hyperic Agent:

1. Click **Resources > Browse**.
2. Click **Servers**.
3. Select **HQ Agent** from the **Server Type** pull-down.



**Browse > vmc-ssrc-2k332 HQ Agent 4.4.0**

Description: Hyperic HQ monitor Agent  
 Sigar Version : 1.6.4.129  
 Sigar Native Version : 1.6.4.127  
 User Home : C:\Documents and Settings\Defa...

Owner: HQ Administrator (hqadmin) - [Change...](#)  
 Java Vendor : Sun Microsystems Inc.  
 Java Version : 16.0-b13  
 Agent Version : 4.4.0

Agent Bundle Version : agent-4.4.0-1509  
 Agent Build : 1509

[Map](#) [Tools Menu](#)

**Monitor** Inventory Alert Views

Metric Display Range:  8  [Advanced Settings](#)

**RESOURCES** **INDICATORS** **METRIC DATA**

Services  
 No health data is available for this resource.

Host Platform Avail  
 vmc-ssrc-2k332.s2qa.com

Show Metrics of Categories  
☒ Availability ☒ Utilization  
☒ Throughput ☒ Performance

Show Metrics with Value Types  
☒ Dynamic ☒ Trends Up  
☒ Trends Down ☒ Static

Keyword Search:

Alerts OOB LOW AVG PEAK LAST Collection Interval

Availability  
☒ Availability 0 0 - 100.0% - 16.1 00:05:00

Performance  
☒ Number of Metrics Sent to Server per Minute 0 0 16.1 16.1 16.1 00:10:00  
☒ Server Offset 0 0 33ms 383.427ms 662ms 00:05:00  
☒ Total Time Spent Fetching Metrics per Minute 0 0 85.808ms 123.635ms 178.298ms 128.122ms 00:10:00

Throughput  
☒ Number of Metrics Collected per Minute 0 0 16.1 16.1 16.1 00:10:00

Utilization  
☒ JVM Free Memory 0 0 2.2 MB 11.3 MB 21.0 MB 14.4 MB 00:05:00  
☒ JVM Total Memory 0 0 31.1 MB 31.1 MB 31.1 MB 31.1 MB 00:05:00

CHART SELECTED METRICS SET BASELINES DISABLE COLLECTION Collection Interval for Selected:  Minutes

## Hyperic Agent Metrics

The table lists all of the metrics that can be collected for an Hyperic Agent. For information about using agent metrics to troubleshoot problems, see [Section 4, “Troubleshoot Agent and Server Problems”](#).

Category	Metric	Notes
Availability		
	Availability	Collected by default.
	Start Time	
	Up Time	
Throughput		
	Number of Active Threads	
	Number of Metrics Collected	
	Number of Metrics Collected per Minute	By default, this is an indicator metric.
	Number of Metrics which Failed to be Collecte	
	Number of Metrics which Failed to be Collected per Minute	
	Number of Requests Served	
	Number of Requests Served per Minute	
Performance		
	Maximum Time Spent Fetching a Metric	
	Maximum Time Spent Processing a Request	

Category	Metric	Notes
	Minimum Time Spent Fetching a Metric	
	Minimum Time Spent Processing a Request	
	Number of Connection Failures	
	Number of Connection Failures per Minute	
	Number of Metric Batches Sent to Server	
	Number of Metric Batches Sent to Server per Minute	
	Number of Metrics Sent to Server	
	Number of Metrics Sent to Server per Minute	Collected by default.
	Server Offset	Collected by default.
	Total Time Spent Fetching Metrics	
	Total Time Spent Fetching Metrics per Minute	Collected by default. High value can indicate overloaded agent or problem with scheduling thread.
	Total Time Spent Processing Requests	
	Total Time Spent Processing Requests per Minute	
	Total Time Spent Sending Metrics to Server	
	Total Time Spent Sending Metrics to Server per Minute	
Utilization		
	Cpu Total Time	
	Cpu Total Time per Minute	
	JVM Free Memory	By default, this is an indicator metric.
	JVM Total Memory	By default, this is an indicator metric.
	Open File Descriptors	
	Resident Memory Used	"Resident Memory" is the amount of memory the Hyperic Agent occupies in memory.
	Time Spent in System Mode	
	Time Spent in System Mode per Minute	
	Time Spent in User Mode	

Category	Metric	Notes
	Time Spent in User Mode per Minute	
	Total Memory Used	

## 2.9.3. Reduce Agent Memory Footprint

Topics marked with\* relate to features available only in vFabric Hyperic.

This page describes options for reducing the amount of memory the Hyperic Agent uses.

- [Limit Plugin Loading](#)
- [Reduce Java Heap](#)
- [Delete Javadocs Folder](#)

### Limit Plugin Loading

The best way to reduce an agent's footprint is to configure it to load only the plugins for the resource types you want to monitor. See [Section 2.6, “Configure Plugin Loading”](#) for instructions.

### Reduce Java Heap

To reduce the Java heap size that an Hyperic Agent allocates for itself on startup, add the `agent.javaOpts` property to the agent's `agent.properties` file. This property does not exist in `agent.properties` - the default behavior is equivalent to the setting shown below. You can reduce the heap from 128m to 64m.

#### agent.javaOpts

##### Description

Additional options to pass to Java.

##### Default

```
-Xmx128m -Xms128m -Djava.net.preferIPv4Stack=true
```

### Delete Javadocs Folder

In an environment where every MB is critical, you can delete the agent's javadocs folder, `agent-4.x.x/bundles/agent-4.x.x-yyyy/pdk/javadoc`; note however that this reduces the agent footprint by only (approximately) 70 MB.

## 2.10. Agent Properties

The properties supported in the `agent.properties` file are defined below. Note that not all supported properties appear in the default `agent.properties` file. To use a property that is not defined in `agent.properties`, you must add it explicitly.

For more information, see [Section 2.1.4, “Agent Configuration”](#).

### Looking for a property that is not listed here?

If your `agent.properties` file contains a property not listed here, please add a comment to this page. One explanation, although uncommon, is the use of special agent properties to override the descriptor-defined value for a `<property>` for resource instances on a particular platform. Such properties have names that reflect a resource type attribute and value. See [Overriding the Value of a Resource property at Platform Level](#).

- [Section 2.10.1, “agent.eventReportBatchSize”](#)
- [Section 2.10.2, “agent.javaOpts”](#)
- [Section 2.10.3, “agent.listenIp”](#)
- [Section 2.10.4, “agent.listenPort”](#)
- [Section 2.10.5, “agent.logDir”](#)
- [Section 2.10.6, “agent.logFile”](#)
- [Section 2.10.7, “agent.logLevel”](#)
- [Section 2.10.8, “agent.logLevel.SystemErr”](#)
- [Section 2.10.9, “agent.logLevel.SystemOut”](#)
- [Section 2.10.10, “agent.maxBatchSize”](#)
- [Section 2.10.11, “agent.proxyHost”](#)
- [Section 2.10.12, “agent.proxyPort”](#)
- [Section 2.10.13, “agent.startupTimeOut”](#)
- [Section 2.10.14, “agent.setup.agentIP”](#)
- [Section 2.10.15, “agent.setup.agentPort”](#)
- [Section 2.10.16, “agent.setup.camIP”](#)
- [Section 2.10.17, “agent.setup.camLogin”](#)
- [Section 2.10.18, “agent.setup.camPword”](#)
- [Section 2.10.19, “agent.setup.camPort”](#)
- [Section 2.10.20, “agent.setup.camSecure”](#)
- [Section 2.10.21, “agent.setup.camSSLPort”](#)

- [Section 2.10.22, “agent.setup.resetToken”](#)
- [Section 2.10.23, “agent.setup.unidirectional”](#)
- [Section 2.10.24, “agent.storageProvider.info”](#)
- [Section 2.10.25, “autoinventory.defaultScan.interval.millis”](#)
- [Section 2.10.26, “autoinventory.runtimeScan.interval.millis”](#)
- [Section 2.10.27, “http.useragent”](#)
- [Section 2.10.28, “jboss.installpath”](#)
- [Section 2.10.29, “log4j Properties”](#)
- [Section 2.10.30, “platform.log\\_track.eventfmt”](#)
- [Section 2.10.31, “plugins.exclude”](#)
- [Section 2.10.32, “plugins.include”](#)
- [Section 2.10.33, “scheduleThread.cancelTimeout”](#)
- [Section 2.10.34, “scheduleThread.fetchLogTimeout”](#)
- [Section 2.10.35, “scheduleThread.poolsize”](#)
- [Section 2.10.36, “scheduleThread.queuesize”](#)
- [Section 2.10.37, “sigar.mirror.procnets”](#)
- [Section 2.10.38, “snmpTrapReceiver.listenAddress”](#)
- [Section 2.10.39, “weblogic.auth.method”](#)
- [Section 2.10.40, “weblogic.installpath”](#)
- [Section 2.10.41, “weblogic.ssl2ways.cert”](#)
- [Section 2.10.42, “weblogic.ssl2ways.key”](#)
- [Section 2.10.43, “weblogic.ssl2ways.key.pass”](#)
- [Section 2.10.44, “websphere.installpath”](#)
- [Section 2.10.45, “websphere.useext”](#)

## 2.10.1. agent.eventReportBatchSize

### Description

The maximum number of events that the HQ Agent will send per contact with the server.

### Default

As installed, `agent.properties` does not contain a line that defines the value of this property. The default behavior of the agent is to send a maximum of 100 events per contact with the server.

## 2.10.2. agent.javaOpts

### Description

Additional options to pass to Java.

### Default

```
-Xmx128m -Xms128m -Djava.net.preferIPv4Stack=true
```

## 2.10.3. agent.listenIp

### Description

The IP address to which the agent binds at startup. The default value allows the agent to listen on all IP addresses on the the agent host.

### Default

As installed, `agent.properties` does not contain a line that defines the value of this property. The default behavior of the agent is to listen on all IP addresses on its host. This behavior is equivalent to to setting this property to an asterisk, like this:

```
*
```

## 2.10.4. agent.listenPort

### Description

The port on the agent's listen address to which the agent binds at startup.

### Default

As installed, `agent.properties` does not contain a line that defines the value of this property. The default behavior of the agent is to listen on port 2144.

## 2.10.5. agent.logDir

### Description

You can add this property to the `agent.properties` file to specify the directory where the HQ Agent will write its log file. Unless you specify a fully qualified path, `agent.logDir` is evaluated relative to the agent installation directory.

This property does not exist in `agent.properties` unless you explicitly add it. To change the location for the agent log file, add `agent.logDir` to `agent.properties` and enter a path relative to the agent installation directory, or if desired, a fully qualified path.

Note that the name of the agent log file is configured with the [Section 2.10.6, “agent.logFile”](#) property.

**Default**

`agent.logDir` does not exist in `agent.properties` unless you explicitly add it. The default behavior is equivalent to this setting:

```
agent.logDir=log
```

resulting in the agent log file being written to the `AgentHome/log` directory.

## 2.10.6. agent.logFile

**Description**

Specifies the path and name of the agent log file.

**Default**

Note that in `agent.properties`, the default setting for `agent.LogFile` is made up of a variable and a string.

```
agent.logFile=${agent.logDir}\agent.log
```

- `agent.logDir` is a variable - it supplies the value of an identically named agent property. By default, the value of `agent.logDir` is `log`, interpreted relative to the agent installation directory.
- `agent.log` is the name for the agent log file.

So, by default, the agent log file is named `agent.log`, and is written to the `AgentHome/log` directory.

If you want the agent to the log to a different directory, you must explicitly add the [Section 2.10.5, “agent.logDir”](#) property to `agent.properties`.

## 2.10.7. agent.logLevel

**Description**

Specifies the level of detail of the messages the Agent writes to the log file. Allowable values are: INFO and DEBUG.

**Default**

INFO

## 2.10.8. agent.logLevel.SystemErr

**Description**

Redirects `System.err` to `agent.log`. Commenting out this setting will cause `System.err` to be directed to `agent.log.startup`.

**Default**

ERROR

## 2.10.9. agent.logLevel.SystemOut

**Description**



Redirects System.out to agent.log. Commenting out this setting will cause System.out to be directed to agent.log.startup.

**Default**

INFO

## 2.10.10. agent.maxBatchSize

**Description**

The maximum number of metrics that the agent will send per contact with the server.

**Default**

As installed, agent.properties does not contain a line that defines the value of this property. The default behavior of the agent is to send a maximum of 500 per contact with the server.

## 2.10.11. agent.proxyHost

**Description**

The host name or IP address of the proxy server that the agent must connect to first when establishing a connection to the HQ server. Supported in HQ Enterprise only.

**Default**

none

## 2.10.12. agent.proxyPort

**Description**

The port number of the proxy server that the agent must connect to first when establishing a connection to the HQ server. Supported in HQ Enterprise only.

**Default**

none

## 2.10.13. agent.startupTimeOut

**Description**

The number of seconds that the agent startup script will wait before determining that the agent did not startup successfully. If the agent is not determined to be listening for requests within this period of time, an error is logged, and the startup script times out.

**Default**

As installed, agent.properties does not contain a line that defines the value of this property. The default behavior of the agent is to timeout after 300 seconds.

## 2.10.14. agent.setup.agentIP

**Description**

This specifies the IP address that the HQ Server will use to contact the HQ Agent. If the agent is on the same host as the server, value of 127.0.0.1 is valid.

If there is a firewall between the server and agent, specify the IP address of the firewall, and configure the firewall to forward traffic intended for the agent to the agent's listen address, which can be configured with agent.listenIP.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.\* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

#### **Default**

As installed, agent.properties contains a commented out statement that sets the value to \*default\*. If you use the agent.setup.\* properties to supply an agent's configuration at first startup, and uncomment this property and leave the value \*default\*, the HQ Server will contact the agent using the IP address that SIGAR detects on the agent host.

## **2.10.15. agent.setup.agentPort**

### **Description**

This specifies the port (on the IP address configured with agent.setup.agentIP) on the agent on which the HQ Server will communication with the agent.

If there is a firewall between the agent and the server, set agent.setup.agentPort to the appropriate port on the firewall, and configure the firewall to forward traffic intended for the agent to the agent listen port, which can be configured with.

The agent reads this value only in the event that it cannot find its connection configuration in its data directory. Specifying this and other agent.setup.\* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

### **Default**

As installed, agent.properties contains a commented out statement that sets the value to \*default\*. If you use the agent.setup.\* properties to supply an agent's configuration at first startup, and uncomment this property and leave the value \*default\*, the HQ Server will contact the agent on port 2144, unless SIGAR detects it is not available, in which case another default is selected.

## **2.10.16. agent.setup.camIP**

### **Description**

You can use this property to define for the agent the IP address of the HQ server. The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.\* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

The value can be provided as an IP address or a fully qualified domain name. To identify an server on the same host as the server, set the value to 127.0.0.1.

If there is a firewall between the agent and server, specify the address of the firewall, and configure the firewall to forward traffic on port 7080, or 7443 if you use the SSL port, to the HQ Server.

### **Default**

Commented out, localhost.

## 2.10.17. agent.setup.camLogin

### Description

You can use this property to define for the agent, at first startup after installation, the HQ username to use when registering itself with the server. The permission required on the server for this initialization is Create, for Platforms.

A login from the agent to the server is only required during the initial configuration of the agent.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.\* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

### Default

Commented out, hqadmin.

## 2.10.18. agent.setup.camPword

### Description

You can use this property to define for the agent, at first startup after installation, the password for the user specified by agent.setup.camLogin.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.\* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

### Default

Commented out, hqadmin.

## 2.10.19. agent.setup.camPort

### Description

You can use this property to define for the agent, at first startup after installation, what server port to use for non-secure communications with the server. The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.\* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

### Default

Commented out, 7080.

## 2.10.20. agent.setup.camSecure

### Description

You can use this property to define for the agent, at first startup after installation, whether to communicate with the server over SSL. If you set this property to yes, all agent-server communications will be use the SSL secure port.

If acceptable in your environment, non-SSL communication offers improved performance for agent-server communications.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.\* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

**Default**

Commented out, value of yes.

## 2.10.21. agent.setup.camSSLPort

**Description**

You can use this property to define for the agent, at first startup after installation, what server port to use for SSL communications with the server. The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.\* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

**Default**

Commented out, 7443.

## 2.10.22. agent.setup.resetupToken

**Description**

You can use this property to define for the agent, at first startup after installation, whether the agent will create a new token to use to authenticate with the server each time it starts up. Regenerating a token is useful if the Agent cannot connect to the server because the token has been deleted or corrupted.

Regardless of the value of this property, an agent will generate a token the first time it is started after installation.

The agent reads this value only in the event that it cannot find connection configuration in its data directory. Specifying this and other agent.setup.\* properties is a way to reduce the user interaction required to configure an agent to communicate with the server.

**Default**

As installed, agent.properties contains a commented out statement that sets the value to no.

## 2.10.23. agent.setup.unidirectional

Available only in **vFabric Hyperic**

**Description**

Enables the unidirectional communications between HQ Agent and HQ Server in vFabric Hyperic. For more information, see [Section 2.2.4, “Configure Unidirectional Agent - Server Communication”](#).

**Default**

Commented out, defaults to no.

## 2.10.24. agent.storageProvider.info

**Description**

Configuration for data storage on the Agent side

**Default**

As installed, agent.properties does not contain a line that defines the value of this property. The default setting of the agent is

```
agent.storageProvider.info=${agent.dataDir}\|m|100|20|50
```

Which means, use the data directory to store the disklists, max size 100MB. The 20 and 50 numbers are used for purging data; check to see if the list can be shortened when the size is greater than 20MB and the list is 50% empty

## 2.10.25. autoinventory.defaultScan.interval.millis

**Description**

Specifies how frequently the agent performs a default autoinventory scan.

The default scan detects servers and platform services, typically using the process table or the Windows registry. Default scans are less resource-intensive than runtime scans.

**Default**

Commented out, set to 86,400,000 milliseconds, or 1 day.

**Note however, that by default, the agent performs the default scan at startup and every 15 minutes thereafter.**

## 2.10.26. autoinventory.runtimeScan.interval.millis

**Description**

Specifies how frequently the agent performs a runtime scan.

A runtime scan may use more resource-intensive methods to detect services than a default scan. For instance, a runtime scan may involve issuing an SQL query or looking up an MBean.

**Default**

86,400,000 milliseconds, or 1 day.

## 2.10.27. http.useragent

**Description**

The http.useragent property defines the value for the User-Agent request header in HTTP requests issued by the HQ Agent. By default, the User-Agent in agent requests includes the HQ Agent version - and so change upon agent upgrade. If a target HTTP server is configured to block requests with an unknown User-Agent, agent requests will fail after agent upgrade.

You can use http.useragent to define a User-Agent value that will be consistent across upgrades.

Note: agent.properties does not contain this property by default. You must explicitly add it to the file.

**Default**

Hyperic-HQ-Agent/Version

For example:

Hyperic-HQ-Agent/4.1.2-EE

## 2.10.28. jboss.installpath

### Description

To enable the agent to monitor JBoss, specify the location of the JBoss root directory.

### Default

/usr/local/jboss-4.0.0

## 2.10.29. log4j Properties

```
log4j.rootLogger=${agent.logLevel}, R
log4j.appender.R.File=${agent.logFile}
log4j.appender.R.MaxBackupIndex=1
log4j.appender.R.MaxFileSize=5000KB
log4j.appender.R.layout.ConversionPattern=%d %-5p [%t] [%c{1}] %m%n
log4j.appender.R.layout=org.apache.log4j.PatternLayout
log4j.appender.R=org.apache.log4j.RollingFileAppender
##
## Disable overly verbose logging
##
log4j.logger.httpclient.wire=ERROR
log4j.logger.org.apache.commons.httpclient=WARN
log4j.logger.org.hyperic.hq.measurement.agent.server.SenderThread=INFO
log4j.logger.org.hyperic.hq.agent.server.AgentDListProvider=INFO
log4j.logger.org.hyperic.hq.agent.server.MeasurementSchedule=INFO
log4j.logger.org.hyperic.hq.measurement.agent.server.ScheduleThreadTrace=INFO
log4j.logger.org.hyperic.util.schedule.ScheduleTrace=INFO
##
## Only log errors from naming context
##
log4j.category.org.jnp.interfaces.NamingContext=ERROR
log4j.category.org.apache.axis=ERROR
```

## 2.10.30. platform.log\_track.eventfmt

### Description

Specifies the content and format of the Windows event attributes that an HQ Agent includes when logging a Windows event as an event in HQ. `agent.properties` does not contain the `platform.log_track.eventfmt` property, you must explicitly add it if you want to tailor the data logged for Windows events.

### Default Behavior

When Windows log tracking is enabled, an entry of this form is logged for events that match the criteria you specified on the resource's **Configuration Properties** page:

[Timestamp] Log Message (EventLogName):EventLogName:EventAttributes

where:

- **Timestamp** - is when the event occurred
- **Log Message** - is an text string

- **EventLogName** - is the Windows event log type, "System", "Security", or "Application".
- **EventAttributes** - a colon delimited string made of the Windows event **Source** and **Message** attributes.

For example, this log entry:

04/19/2010 06:06 AM Log Message (SYSTEM): SYSTEM: Print: Printer HP LaserJet 6P was paused.

is for an Windows event written to the Windows System event log at 6:06 AM on 04/19/2010. The Windows event **Source** and **Message** attributes, are "Print" and "Printer HP LaserJet 6P was paused.", respectively.

### Configuration

You can use the parameters below to configure the Windows event attributes that the agent writes for a Windows event. Each parameter maps to Windows event attribute of the same name.

- **%user%** - The name of the user on whose behalf the event occurred.
- **%computer%** - The name of the computer on which the event occurred.
- **%source%** - The software that logged the Windows event.
- **%event%** - A number identifying the particular event type.
- **%message%** - The event message.
- **%category%** - An application-specific value used for grouping events.

For example, with this property setting:

```
platform.log_track.eventfmt=%user%@%computer% %source%:%event%:%message%
```

the HQ Agent will write the following data when logging Windows event:

```
04/19/2010 06:06 AM Log Message (SYSTEM): SYSTEM: HP_Administrator@Office Print:7:Printer HP
LaserJet 6P was paused.
```

This entry is for as for an Windows event written to the Windows System event log at 6:06 AM on 04/19/2010. The software associated with the event was running as "HP\_Administrator" on the host "Office". The Windows event's **Source**, **Event**, and **Message** attributes, are "Print", "7", and "Printer HP LaserJet 6P was paused.", respectively.

After you configure the content of the log entry written for a Windows event, when you configure an alert definition for a Windows resource, you can create an alert condition based on the message content, including the custom fields you have configured. For more information, see the "Define Alert Condition Set" section in *Configure Monitoring Options*.

## 2.10.31. plugins.exclude

### Description

Use this property to specify plugins that you do not wish the HQ Agent to load at startup. This is useful for reducing the agents memory footprint.

### Usage

Supply a comma-separated list of plugins to exclude, for example:

```
plugins.exclude=jboss,apache,mysql
```

### 2.10.32. plugins.include

#### Description

Use this property to specify plugins that you do wish the HQ Agent to load at startup. This is useful for reducing the agents memory footprint.

#### Usage

Supply a comma-separated list of plugins to include, for example:

```
plugins.include=weblogic,apache
```

### 2.10.33. scheduleThread.cancelTimeout

**This property was added in Hyperic 4.5.1.**

#### Description

The maximum time, in milliseconds, the `ScheduleThread` will allow a metric collection process to run before attempting to interrupt it. When the timeout is exceeded, collection of the metric will be interrupted, if it is in an interruptible state, that is, in a `wait()`, `sleep()` or non-blocking `read()` state.

Default is 5000.

#### Usage

```
scheduleThread.cancelTimeout=5000
```

### 2.10.34. scheduleThread.fetchLogTimeout

**This property was added in Hyperic 4.5.1.**

#### Description

The property controls when a warning message is issued for a long-running metric collection process. If a metric collection process exceeds the value of this property, measured in milliseconds, the agent writes a warning message to the `agent.log` file.

Default is 2000.

#### Usage

```
scheduleThread.fetchLogTimeout=2000
```

### 2.10.35. scheduleThread.poolsize

**This property was added in Hyperic 4.5.1.**

#### Description



This property allows a plugin to use multiple threads for metric collection. This property may increase metric throughput for plugins known to be thread safe.

Default is 1.

**Usage**

Specify the plugin by name and the number of threads to allocate for metric collection:

```
scheduleThread.poolsize.PluginName=2
```

where PluginName is the name of the plugin to which you are allocating threads.

for example:

```
scheduleThread.poolsize.vsphere=2
```

## 2.10.36. scheduleThread.queueSize

**This property was added in Hyperic 4.5.1.**

**Description**

This property can be used to limit the metric collection queue size (the number of metrics) for a plugin.

Default is 10000.

**Usage**

Specify the plugin by name and the number of maximum metric queue length:

```
scheduleThread.queueSize.PluginName=15000
```

where PluginName is the name of the plugin upon which you are imposing a metric limit.

for example:

```
scheduleThread.queueSize.vsphere=15000
```

## 2.10.37. sigar.mirror.procnets

**Description**

mirror /proc/net/tcp on linux

**Default**

true

## 2.10.38. snmpTrapReceiver.listenAddress

**Description**

Use this property to specify the port on which the HQ Agent listens for SNMP traps. By default, the agent is not configured to listen for SNMP traps. You must add this property to `agent.properties` to enable the agent to receive traps.

Typically SNMP uses the UDP port 162 for trap messages. This port is in the privileged range, so an agent listening for trap messages on it must run as root (or as an Administrative user on Windows).

If you prefer to run the the agent under the context of a non-administrative user, you can configure it to listen for trap messages on an unprivileged port.

**Usage**

Specify an IP address (or 0.0.0.0 to specify all interfaces on the platform) and the port for UDP communications in this format:

```
snmpTrapReceiver.listenAddress=udp:IP_address/port
```

To enable the HQ Agent to receive SNMP traps on an unprivileged port, specify port 1024 or above. The following setting allows the agent to receive traps on any interface on the platform, on UDP port 1620.

```
snmpTrapReceiver.listenAddress=udp:0.0.0.0/1620
```

## 2.10.39. weblogic.auth.method

**Description**

`weblogic.auth.method` is one of four properties you define to enable an HQ Agent to communicate with a WebLogic Administration Server using Two-Way-SSL.

Add the following line to the `agent.properties` file to specify that the agent will use Two-Way-SSL for communications with the Administration Server.

```
weblogic.auth.method=ssl2ways
```

**Default**

None.

## 2.10.40. weblogic.installpath

**Description**

To enable the agent to monitor WebLogic 8.1, specify the location `server/lib/weblogic.jar`

**Default**

```
/usr/local/boa/weblogic-8.1
```

## 2.10.41. weblogic.ssl2ways.cert

**Description**

`weblogic.ssl2ways.cert` is one of four properties you define to enable an HQ Agent to communicate with a WebLogic Administration Server using Two-Way-SSL.

Add `weblogic.ssl2ways.cert` to the `agent.properties` file and set its value to the location of the client certificate that the HQ Agent will present to the Administration Server:

```
weblogic.ssl2ways.cert=Client2Cert.pem
```

where *Client2Cert.pem* is the **path to the client certificate** the HQ Agent presents to the Administration Server it manages.

**Default**

None.

## 2.10.42. weblogic.ssl2ways.key

### Description

`weblogic.ssl2ways.key` is one of four properties you define to enable an HQ Agent to communicate with a WebLogic Administration Server using Two-Way-SSL.

`weblogic.ssl2ways.key` to the `agent.properties` file and set its value to the location of client's private key:

```
weblogic.ssl2ways.key=clientKey.pem
```

where:

*clientkey.pem* is the **path to the private key** the HQ Agent presents to the Administration Server that the agent manages.

### Default

None.

## 2.10.43. weblogic.ssl2ways.key.pass

### Description

`weblogic.ssl2ways.key.pass` is one of four properties you define to enable an HQ Agent to communicate with a WebLogic Administration Server using Two-Way-SSL.

Add `weblogic.ssl2ways.key.pass` to the `agent.properties` file and set its value to the passphrase for the client private key:

```
weblogic.ssl2ways.key.pass=ClientKey
```

where *ClientKey* is the passphrase for the client private key.

### Default

None.

## 2.10.44. websphere.installpath

### Description

To enable the agent to monitor WebSphere, specify the location of the WebSphere jars.

### Default

```
/opt/WebSphere/AppServer
```

## 2.10.45. websphere.useext

### Description

This property is required to enable management of WebSphere 6.0 and 6.1.

Do **not** define the `websphere.useext` property to monitor WebSphere 7.

**Usage**

Add the following property definition to the `agent.properties` file for an an HQ Agent that will manage WebSphere 6.0 or 6.1.

```
websphere.useext=true
```

## 3. Configure and Run the Hyperic Server

This page is a linked table of contents for *Configure and Run the Hyperic Server*.

- [Section 3.1, “Start and Stop Hyperic Server”](#)
- [Section 3.2, “Configure Hyperic Server Help and Announcement Behavior”](#)
- [Section 3.3, “Configure Metric Baseline and Alert Processing Behavior”](#)
- [Section 3.4, “Managing the HQ Database”](#)
- [Section 3.5, “Scaling and Tuning Hyperic Performance”](#)
- [Section 3.6, “Integrate HQ Server with Other Systems”](#)
- [Section 3.7, “Clustering HQ Servers for Failover”](#)
- [Section 3.8, “Hyperic Server Properties”](#)
- [Section 3.9, “HQ Database Table Schemas”](#)

## 3.1. Start and Stop Hyperic Server

These topics have information about starting and stopping Hyperic Server:

- [Section 3.1.1, “Starting the Server on Unix-Based Platforms”](#)
- [Section 3.1.2, “Starting the Server on Windows To Run as a Service”](#)
- [Section 3.1.3, “Updating SSL Certs for Hyperic Server”](#)

### 3.1.1. Starting the Server on Unix-Based Platforms

Start the server with this command:

```
<Server Installation Directory>/bin/hq-server.sh start
```

The script will display some startup information on stdout, then it will detach and run in the background

The server.log file and bootstrap.log files in the logs directory under the server installation directory will have detailed startup information.

### 3.1.2. Starting the Server on Windows To Run as a Service

The first time you start up the server after installation, use this command to start it as a Windows Service:

```
<Server Installation directory>\bin\hq-server.bat install
```

Henceforth, use the Windows Service control panel to start and stop the server.

### 3.1.3. Updating SSL Certs for Hyperic Server

When Hyperic is configured to use SSL for agent-server communications, it uses the built-in certs in `hq-engine/hq-server/conf/hyperic.keystore`. You can replace the built-in certs with your own.

If your certificate is not PKCS12 format, you can use the **openssl** tool to convert it. For more information, see <http://community.jboss.org/wiki/sslsetup>.

After ensuring your certificate is in the correct format, use **java-keytool** to install it. For more information, see <http://java.sun.com/j2se/1.4.2/docs/tooldocs/windows/keytool.html>.

## 3.2. Configure Hyperic Server Help and Announcement Behavior

Topics marked with\*relate to features available only in vFabric Hyperic.

- [Section 3.2.1, “Configure User Interface Help Location”](#)
- [Section 3.2.2, “Configure Hyperic Version and Security Announcements”](#)

### 3.2.1. Configure User Interface Help Location

By default, the online help for Hyperic user interface is served remotely from the Hyperic support site. Help is also built-in to the Hyperic Server. To configure the server to present local help, toggle the value of the **Context-Sensitive Help** property in the **Announcement Properties** section of the **Administration > HQ Server Settings** page, from "Remote" to "Internal".

### 3.2.2. Configure Hyperic Version and Security Announcements

Hyperic sends email announcements to Hyperic administrators when a key release is upcoming, or to distribute important product information. You can configure the level of messages you wish to receive or disable receipt of Hyperic notifications with the **HQ Version and Security Announcements** property, in the **Announcement Properties** section of the **Administration > HQ Server Settings** page. You can choose:

- **All**
- **Major** — default value
- **None**

## 3.3. Configure Metric Baseline and Alert Processing Behavior

These topics have instructions for configuring HQ Server baselining and alert processing behaviors:

- [Section 3.3.1, “Configure Global Alert Properties”](#)
- [Section 3.3.2, “Configure Alert Notification Throttling”](#)
- [Section 3.3.3, “Configure Alert Notification Email Properties”](#)
- [Section 3.3.4, “Configure Metric Baseline Properties”](#)



### 3.3.1. Configure Global Alert Properties

The settings in the **Global Alert Properties** section of the **Administration > HQ Server Settings** page enable immediate and global control of alert processing.

- **Alerts** - Disable or enable all alert definitions for all resources immediately. Disabling stops any alerts from firing; notifications defined in escalations that are currently in progress will be completed.
- **Alert Notifications** - Disable or enable alert notifications for all resources immediately. Disabling stops all notifications, include those for alerts with escalations currently in progress.
- **Hierarchical Alerting\*** - This setting controls whether alerts are evaluated using the hierarchical alerting method. When hierarchical alerting is enabled, before firing an alert for a resource, HQ considers the availability and alert status of the resource's parent. The purpose of hierarchical alerting is to avoid firing alerts for every resource affected by a single root cause.

**Note:** You can extend the effect of hierarchical alerting by configuring the relationship between a network device or virtual host and the platforms that depend on it using the **Network and Host Dependency Manager** available in the "Plugins" section of the **Administration** tab. For more information see [Configure Network Host Dependencies for Hierarchical Alerting](#).

### 3.3.2. Configure Alert Notification Throttling

Available only in **vFabric Hyperic**

You can use notification throttling to limit the number of alert email actions (notifications sent by email for a fired alert) that HQ will issue in a 15 second interval. When the threshold you specify is reached, HQ stops sending email alert notifications and instead sends a summary of alert activity every ten minutes to the recipients you specify.

After starting to throttle, HQ re-evaluates notification volume for fired alerts every 10 minutes; when it determines that the per interval volume of individual notifications that fired alerts would generate is less than the configured threshold, HQ resumes sending individual notifications.

In the **Notification Throttling Configuration Properties** section of the **Administration > HQ Server Settings** page:

1. Click the **Notification Throttling** ON control.
2. In the "Threshold" field, enter the maximum number of notifications you want sent in a 15 second interval.
3. Enter one or more email addresses in the "Notification Email(s) field".

For related information, see [Controlling Alert and Notification Volume](#).

### 3.3.3. Configure Alert Notification Email Properties

The settings in the **Email Configuration Properties** section of the **Administration > HQ Server Settings** are used to form notifications that Hyperic sends for a fired alert.

Property	Description
Base URL	The address:port where the Hyperic Server listens for web application requests. The initial value of <b>Base URL</b> is the web application listen port configured when the Hyperic Server was installed, for example:

Property	Description
	<p><code>http://Ms-MacBook-Pro-15.local:7080</code></p> <p><b>Base URL</b> forms the prefix of the URL to which Hyperic appends the remainder of the URL, which points to the <b>Alert Detail</b> page for the fired alert. For example:</p> <p><code>http://Ms-MacBook-Pro-15.local:7080/alerts/Alerts.do?mode=viewAlert&amp;eid=5:10611&amp;a=16431</code></p>
From Email Address	<p>The email address listed as the sender of the alert emails. For example:</p> <p><code>hq@demo2.vmware.com</code></p>

### 3.3.4. Configure Metric Baseline Properties

Available only in **vFabric Hyperic**

In vFabric Hyperic, the properties in the **Automatic Baseline Configuration Properties** section of the **Administration > HQ Server Settings** page control the Hyperic baselining process and the accuracy of the baseline.

Server Setting	Description	Default
Baseline Frequency	The frequency with which HQ calculates a baseline for each metric.	3 days
Baseline Dataset	The time range of metric data used in calculating the baseline.	7 days
Baseline Minimum Data Points	The minimum number of data points used in calculating a baseline.	40
Track Out-of-Bounds Metrics	Controls whether or not HQ tracks <a href="#">OOB metrics</a> .	off

## 3.4. Managing the HQ Database

This section has topics related to the HQ database. It includes information about database maintenance and optional configurations.

- [Section 3.4.1, “Hyperic Database Backup and Recovery”](#)
- [Section 3.4.2, “Building a Metric Data Warehouse”](#)
- [Section 3.4.3, “Configure HQ Server Data Compression and Purge Behavior”](#)
- [Section 3.4.4, “Monitoring the HQ Database”](#)

### 3.4.1. Hyperic Database Backup and Recovery

The Hyperic database contains most of the data necessary to recreate your Hyperic Server environment after a failure, or to move the database to a different host. In addition to historical metrics, the database contains configuration settings, such as Hyperic Agent connection information, collection intervals, portlet configurations, groups, roles, and users. Some server configuration data, such as database connection information, the mail server for alerts, and Java arguments used at server startup, is stored in external files.

Like any other database, your Hyperic database should be backed up on a regular basis, so that you can restore the data in the event of a failure that corrupts or destroys the database. It is also good practice to backup the database prior to upgrading Hyperic, your database server, or other software that resides on the server machine.

You should define Hyperic backup procedures and incorporate them into your overall backup processes. Your local requirements and practices will dictate backup frequency, timing, naming conventions, and retention policies. A daily backup is sufficient for most environments.

- [Backing up Built-In PostgreSQL Database](#)
- [Backup and Recovery of MySQL and Oracle Databases](#)
- [Hyperic Files to Backup](#)

#### Backing up Built-In PostgreSQL Database

If you use Hyperic's built-in PostgreSQL database, back it up with the PostgreSQL `pg_dump` command:

```
pg_dump hqdb | gzip > hqdb-MM.DD.YY.dump.gz
```

Copy the dump file to your backup location.

Always use this method to back up the built-in database; do not simply copy the contents of the database's data directory.

#### Backup and Recovery of MySQL and Oracle Databases

If you use MySQL or Oracle for your Hyperic database, extend your existing database backup and recovery procedures to the Hyperic database. Perform a full database backup on a scheduled basis.

#### Hyperic Files to Backup

In addition to the Hyperic database, you may want to create a backup of the following directories and files in your server directory:

```
conf/  
bin/hq-server.sh  
hqdb/data/postgresql.conf
```

You can back up these files while Hyperic Server is running.

The contents of these files are stable. Changes are infrequent once your Hyperic Server is installed at configured. Back them up at that time and after making changes to the sever configuration.

### 3.4.2. Building a Metric Data Warehouse

Hyperic's retention strategy for measurement data is it to store the minimum amount of data that enables it to pinpoint when change in performance or availability occur. Detailed measurement data is stored for a limited period of time - two days, by default - after which the data is compressed and archived as hourly averages with highs and lows. You can configure Hyperic to keep detailed measurement data for longer, up to a maximum of 7 days.

To support requirements for trend analysis over a longer time frame, vFabric Hyperic versions 3.2 and later provide the MetricDataReplicator class, which you can use to replicate uncompressed measurement data in a secondary database.

- [Metric Replication Strategy Overview](#)
- [Instructions for Establishing Secondary MySQL Database for Metrics](#)
- [Set Up the Secondary Database](#)
- [Test the New Views](#)
- [Set up the Metric Data Replicator](#)
- [Create log4j Properties File](#)
- [Create Script to Run the Replication Process](#)
- [Run the Replication Process](#)
- [Verify the Replication Process Results](#)

#### Metric Replication Strategy Overview

Detailed steps for creating and populating a secondary database for detailed metrics are provided in the sections that follow. This is a summary of the approach:

- A secondary database instance is configured to store detailed measurement data replicated from the primary Hyperic database. The secondary database contains one table, EAM\_MEASUREMENT\_DATA. This guide currently only covers MySQL, adjustments will need to be made for Oracle and PostgreSQL.
- The secondary database has a database link to the primary Hyperic database, and five views that point to the primary Hyperic database for resource inventory data. The resource inventory data does not physically reside on the secondary database. The database link to the main database allows views on the secondary database to access inventory data in the primary Hyperic database. These are the views that are required on the secondary database:
  - EAM\_PLATFORM
  - EAM\_SERVER
  - EAM\_SERVICE
  - EAM\_RESOURCE
  - EAM\_MEASUREMENT\_TEMPL
  - EAM\_MEASUREMENT

For documentation on these database tables, see [Section 3.9, “HQ Database Table Schemas”](#).

## Instructions for Establishing Secondary MySQL Database for Metrics

These sections below have instructions for configuring a secondary Hyperic database for metrics on MySQL.

Note: MySQL's query optimizer has limitations that have negative impact on the the performance of Hyperic's metric replicator class. This performance degradation can have a ripple effect on the performance of your primary Hyperic database during metric replication as such the replicator class should be ran against a slave mysql db for optimum performance. This document is written around the use of the slave DB.

### Set Up the Secondary Database

Perform the steps below to create the secondary database and configure access to your primary Hyperic database. All of the steps in this section apply to the secondary database.

Another option would be to use MySQL federated tables

1. Install your secondary MySQL Database Server and setup replication.
2. Create user and database for warehouse

```
CREATE DATABASE hqdata;  
GRANT ALL ON hqdata.* to hqdata identified by 'password';  
FLUSH PRIVILEGES;  
USE hqdata;
```

3. Create table for measurements

```
create table EAM_MEASUREMENT_DATA  
(  
    TIMESTAMP bigint,  
    MEASUREMENT_ID int,  
    VALUE numeric(24, 5),  
    primary key (TIMESTAMP, MEASUREMENT_ID)  
);
```

4. Create view measurements, replace hqdb with your Hyperic Db name

```
create view EAM_MEASUREMENT  
as select ID,  
    VERSION_COL,  
    INSTANCE_ID,  
    TEMPLATE_ID,  
    MTIME,  
    ENABLED,  
    COLL_INTERVAL,  
    DSN  
from hqdb.EAM_MEASUREMENT;
```

5. Create view for platforms, replace hqdb with your Hyperic DB name

```
create view EAM_PLATFORM  
as select ID,  
    VERSION_COL,  
    FQDN,  
    CERTDN,  
    DESCRIPTION,
```

```
CTIME,  
MTIME,  
MODIFIED_BY,  
LOCATION,  
COMMENT_TEXT,  
CPU_COUNT,  
PLATFORM_TYPE_ID,  
CONFIG_RESPONSE_ID,  
AGENT_ID,  
RESOURCE_ID  
from hqdb.EAM_PLATFORM;
```

6. Create view for servers, replace hqdb with your Hypierc DB name

```
create view EAM_SERVER as  
select ID,  
        VERSION_COL,  
        DESCRIPTION,  
        CTIME,  
        MTIME,  
        MODIFIED_BY,  
        LOCATION,  
        PLATFORM_ID,  
        AUTOINVENTORYIDENTIFIER,  
        RUNTIMEAUTODISCOVERY,  
        WASAUTODISCOVERED,  
        SERVICESAUTOMANAGED,  
        AUTODISCOVERY_ZOMBIE,  
        INSTALLPATH,  
        SERVER_TYPE_ID,  
        CONFIG_RESPONSE_ID,  
        RESOURCE_ID  
from hqdb.EAM_SERVER;
```

7. Create view for services, replace hqdb with your Hyperic DB name

```
create view EAM_SERVICE as  
select ID,  
        VERSION_COL,  
        DESCRIPTION,  
        CTIME,  
        MTIME,  
        MODIFIED_BY,  
        LOCATION,  
        AUTODISCOVERY_ZOMBIE,  
        SERVICE_RT,  
        ENDUSER_RT,  
        PARENT_SERVICE_ID,  
        SERVER_ID,  
        SERVICE_TYPE_ID,  
        CONFIG_RESPONSE_ID,  
        RESOURCE_ID  
from hqdb.EAM_SERVICE;
```

8. Create view for resources, replace hqdb with your Hyperic Db name

```
create view EAM_RESOURCE  
as select ID,  
        VERSION_COL,  
        RESOURCE_TYPE_ID,  
        INSTANCE_ID,  
        SUBJECT_ID,  
        PROTO_ID,  
        NAME,
```

```

        SORT_NAME,
        FSYSTEM,
        MTIME
    from hqdb.EAM_RESOURCE;

```

9. Create view for measurement templates, replace hqdb with your Hyperic DB name

```

create view EAM_MEASUREMENT_TEMPL as
select ID,
        VERSION_COL,
        NAME,
        ALIAS,
        UNITS,
        COLLECTION_TYPE,
        DEFAULT_ON,
        DEFAULT_INTERVAL,
        DESIGNATE,
        TEMPLATE,
        PLUGIN,
        CTIME,
        MTIME,
        MONITORABLE_TYPE_ID,
        CATEGORY_ID
    from hqdb.EAM_MEASUREMENT_TEMPL;

```

10.OPTIONAL: Create view for Availability data, replace hqdb with your Hyperic DB name

```

create view HQ_AVAIL_DATA_RLE
as select MEASUREMENT_ID,
        STARTTIME,
        ENDTIME,
        AVAILVAL
    from hqdb.HQ_AVAIL_DATA_RLE;

```

Availability isn't replicated as it is stored differently and doesn't have compression. This view is provided as an example if you want to query a single db for availability data.

## Test the New Views

To verify the views you created work, you can run a query that lists all servers in the database. Enter the following query at the mysql prompt of the secondary database and to list all of the servers. This query runs against the hqdb database.

```

SELECT * FROM EAM_SERVER;

```

## Set up the Metric Data Replicator

Once your secondary database is ready to store and view the Hypierc data, you must set up the metric\_replicator.properties file with the appropriate parameters. Create a directory where the replicator files will be stored, for instance:

/usr/hyperic/replicator

Property Setting	Description
pri_user=hqadmin	Primary database username
pri_pass=hqadmin	Primary database password
pri_url=jdbc:mysql://<ipaddress>:<port>/hqdb	Connection string for primary server, including IP Address and port



Property Setting	Description
sec_user=hqdata	Secondary database username
sec_pass=password	Secondary database password
sec_url=jdbc:mysql://<ipaddress>:<port>/hqdata?	Connection string for secondary server, including IP Address and port.
interval	Time interval in minutes. Use to specify the interval the script runs at. Default: 90
time_chunk	Amount of minutes to do during this run. Default: 90
batch_size	Number of metrics do to in a batch. Default: 2000

## Create log4j Properties File

Create a file called `log4j.properties` in the replicator directory you created in the previous step and paste this text into the file:

```
log4j.rootLogger=DEBUG, R
log4j.appender.R=org.apache.log4j.ConsoleAppender
log4j.appender.R.layout=org.apache.log4j.PatternLayout
log4j.appender.R.layout.ConversionPattern=%d \[PRIVATE:%t\] %-5p %c - %m%n
```

## Create Script to Run the Replication Process

Create a file called `run.sh`, which will be the script that runs the replication process. Copy the commands shown below, changing the values of `JAVA_HOME` and `SERVER_HOME` to point to your Java and Hyperic Server installations respectively.

```
#!/bin/bash

#update these params
SERVER_HOME="/opt/vmware/vfabric/hyperic/server/active/hq-engine/hq-server"

# props file which configures the replicator
PROPS="metric_replicator"

HQ_ROOT="$SERVER_HOME/webapps/ROOT"

# path to the prop files
# update to reflect appropriate database
PROP_FILES="."
DB_PKGS="$HQ_ROOT/WEB-INF/lib/mysql-connector-java-commercial-5.1.10.jar"
HQ_PKGS="$HQ_ROOT/WEB-INF/lib/hq-common-4.5.jar"
LOG_PKGS="$HQ_ROOT/WEB-INF/lib/commons-logging-1.0.4.jar:$HQ_ROOT/WEB-INF/lib/log4j-1.2.14.jar"
HQEE_PKGS="$HQ_ROOT/WEB-INF/lib/hqee-server-4.5.jar"
HQUTIL_PKGS="$HQ_ROOT/WEB-INF/lib/hq-util-4.5.jar"
PKGS="$PROP_FILES:$DB_PKGS:$HQ_PKGS:$HQEE_PKGS:$LOG_PKGS:$HQUTIL_PKGS"
ARGS="$LOG_ARGS -Dreplprops=$PROPS -Djdbc.drivers=com.mysql.jdbc.Driver -cp $PKGS"
JAVA="$JAVA_HOME/bin/java"

set -x

$JAVA $ARGS com.hyperic.hq.measurement.shared.MetricDataReplicator
```

## Run the Replication Process

To run the replication process, open a terminal window and enter:

```
run.sh
```

## Verify the Replication Process Results

Run the queries in the following sections to verify that the replication process succeeded.

### Query 1: Show all the disk stats

Run the following query to show all metrics whose name contains the string "disk", replacing the your.platform.name below with a valid platform name in your resource inventory.

```
SELECT p.fqdn,
r.name,
t.name,
d.value,
d.timestamp
from EAM_MEASUREMENT_TEMPL t,
EAM_MEASUREMENT m,
EAM_MEASUREMENT_DATA d,
EAM_SERVER s,
EAM_PLATFORM p,
EAM_RESOURCE r
where t.id = m.template_id AND
m.id = d.measurement_id AND
p.id = s.platform_id AND
r.instance_id = m.instance_id AND
lower(p.fqdn) = 'your.platform.name' AND
lower(r.name) LIKE '%Mount%'
ORDER BY d.timestamp DESC;
```

### Query 2: Component Service Usage Information for Servers and Services

Once the appropriate join has been made to access the server or service layer, filtering by server or service name or metric template is the easiest way to select specific metrics of interest per server or service. (Meaning "server" and "service", as defined in the HQ inventory model.) For example, JBoss is a server while the individual web applications running within the container are services. Metrics specific to the JBoss container are available from the server layer while the internal web applications are available via the service layer.

```
SELECT platform.fqdn,
resource.name,
template.name,
data.value,
data.timestamp
FROM EAM_MEASUREMENT_TEMPL template,
EAM_MEASUREMENT measurement,
EAM_MEASUREMENT_DATA data,
EAM_SERVICE service,
EAM_SERVER server,
EAM_PLATFORM platform,
EAM_RESOURCE resource
WHERE 1=1 AND
template.id = measurement.template_id AND
measurement.id = data.measurement_id AND
platform.id = server.platform_id AND
server.id = service.server_id AND
service.id = measurement.instance_id AND
lower(platform.fqdn) = 'your.platform.name' AND
lower(resource.name) like '%jboss%'
AND lower(template.name) like '%transaction count%'
ORDER BY data.timestamp desc;
```



### 3.4.3. Configure HQ Server Data Compression and Purge Behavior

#### HQ Server Data Management Processes

HQ Server stores monitoring results using a tiered model to minimize the volume of data stored, while still providing sufficient data granularity. Periodically, the HQ Server removes detailed metric data from the database and archives it. Alerts and events older than a specified age are removed from the database, and not archived.

The server performs the following periodic data management functions:

- **Compress and archive measurement data** — HQ Server stores detailed metric data (all data points reported) in the HQ database for a configurable period (up to 7 days) of time, after which the metrics are eligible for compression and archival. On a (configurable) periodic basis, the server removes the aged individual metric data points from the database, and archives the metric data in compressed form: hourly metric averages, highs, and lows. HQ Server retains the archived metric data for 2 years.
- **Purge alert data** — HQ Server retains fired alert data for a configurable period, after which the alerts are deleted.
- **Purge event data** — HQ Server retains event data for a configurable period, after which the events are deleted.
- **Rebuild metric table indexes** — During normal HQ operation, the metric data tables in the HQ database contain a lot of frequently changing data. The HQ Server rebuilds the metric table indexes on a (configurable) periodic basis to avoid performance problems that heavily fragmented indexes can cause.

#### Configure HQ Data Management Settings

You can configure how HQ condenses and purges the contents of the HQ database on the **Administration > Server Settings** page. Retaining fewer days of detailed metric data and deleting alerts and other events on a timely basis can improve HQ performance.

Option	Description	Default	Notes
Run Database Maintenance Every	Controls how frequently HQ compresses and archives detailed metric data that is older than the age specified by the following property.	Hourly	
Delete Detailed Metric Data Older Than	Controls how many days of detailed metric data HQ retains before compressing it into hourly averages with highs and lows and archiving those values.	2 days	You cannot enter a value greater than 7.
Reindex Metric Data Tables Nightly	Controls whether HQ reindexes metric data tables every night. If configured to re-index nightly, HQ re-indexes the tables around midnight.	Nightly	

Option	Description	Default	Notes
Delete Alerts Older Than	Controls how long HQ stores alert event data.	31 days	
Delete Events and Logs Older Than	Controls how long HQ stores other HQ event and log data.	31 days	

**Warning: Data Management Changes Require Server Restart**

Any changes made in this section require the HQ Server to be restarted before they take effect.

### 3.4.4. Monitoring the HQ Database

HQ administrators can view real-time HQ Server and database health and load data by clicking **HQ Health** on the Administration page in the HQ user interface.

The information on the HQ Health page is useful to HQ internal experts; Hyperic support engineers may use the HQ Health data and diagnostics to diagnose and troubleshoot HQ Server and database problems. For more information, see the associated [help page](#).

For database configuration options that may be useful in large HQ deployments, see the topics in [Section 3.5, “Scaling and Tuning Hyperic Performance”](#).

## 3.5. Scaling and Tuning Hyperic Performance

- [Section 3.5.1, “Sizing Considerations”](#)
- [Section 3.5.2, “Increasing Java Heap and Changing GC Settings”](#)
- [Section 3.5.3, “Increase JMS Memory”](#)
- [Section 3.5.4, “Increase the Number Connections to Database”](#)
- [Section 3.5.5, “Configure Hyperic Cache Settings for Improved Performance”](#)
  - [Where Are Caches Kept](#)
  - [Default Cache Settings](#)
  - [Monitoring Caches to Identify Areas for Improvement](#)
  - [Changing Caches Settings](#)
  - [Configuring Permission Cache for Improved Performance \\*](#)

### 3.5.1. Sizing Considerations

The number of platforms the Hyperic Server can manage depends on the hardware it runs on, the number of Hyperic Agents reporting to the server, the volume of metrics that are collected, and the size of the Hyperic database.

As a general rule, assume a minimal system configuration will support at least 25 Agents or more. On a high performance platform, a properly tuned Hyperic server can support from 100's to 1000's of agents.

This page has instructions for tuning Hyperic for large environments.

**Note:** See [Section 1.1, “Installation Requirements”](#) for Hyperic Server system requirements.

### 3.5.2. Increasing Java Heap and Changing GC Settings

Heap size startup options are set in the `ServerHome/conf/hq-server.conf` file using the `server.java.opts` property. Default settings are described in [server.java.opts](#).

How much you can increase the default heap options depends on the amount of RAM on the Hyperic Server host. Given sufficient RAM, you could use these settings:

```
server.java.opts=-XX:MaxPermSize=192m -Xmx4096m -Xms4096m -XX:+UseConcMarkSweepGC
```

If you increase heap size, you may need to change JMS memory settings as well. For more information, see the [Increase JMS Memory Settings](#) below.

**Note:** If you are running Hyperic Server on a 64-bit system with 4GB (4096 MB) or less memory, Hyperic recommends you use 32-bit JVM. A 64-bit JVM is not recommended unless you have more memory. You might need twice as much heap on a 64-bit system as on a 32-bit system to achieve the same performance.

### 3.5.3. Increase JMS Memory

The JMS memory settings are configured using `server.HighMemory` and `server.MaxMemory` in `ServerHome/conf/hq-server.conf`, by default, 350MB and 400MB, respectively.

The values for `server.HighMemory` and `server.MaxMemory` should be 80% and 90% of the Java heap size, respectively. Erratic alert behavior or missed alerts may indicate the settings are too low.

### 3.5.4. Increase the Number Connections to Database

The size of the connection pool for communication between Hyperic Server and the database is configured with `server.database-minpoolsize` and `server.database-maxpoolsize` in `Server-Home/conf/hq-server.conf`, by default, 5 and 100, respectively.

In large environments, it may be appropriate to increase the size of the connection pool. If you are managing between 800 and 1500 platforms, set both `server.database-minpoolsize` and `server.database-maxpoolsize` to 300.

Given properly configured caches, it should not be necessary to increase connection pool size to more than 700.

**Note:** If you increase `server.database-minpoolsize` to 100 or more, set `server.database-maxpoolsize` equal to that value.

### 3.5.5. Configure Hyperic Cache Settings for Improved Performance

The default cache settings in Hyperic are suitable for most deployments. However, depending on your deployment and use of Hyperic, especially in large environments, changing the cache settings might improve performance.

#### Where Are Caches Kept

All internal caches are kept in EHCache and are accessible via the "HQ Health" screen. EHCache provides:

- Automatic integration with Hibernate
- Optional distributed caching when running in a cluster.
- A variety of proven, well-tested eviction algorithms
- Better cache management and configuration

#### Default Cache Settings

By default, the cache settings in Hyperic are set for a medium to large deployment:

- 100 Platforms
- 500 Servers
- 5000 Services

These settings should suffice for most deployments.

#### Monitoring Caches to Identify Areas for Improvement

You can monitor Hyperic's internal caches through the `server.log` or via the "HQ Health" screen. The log contains diagnostic information, which is written to the log periodically. The cache information should look like:

Cache	Size	Hits	Misses
-------	------	------	--------



```

=====
Agent.findByAgentToken      2      4184      2
AgentScheduleInQueue       0      0      0
Alert.findByCreateTime     10     318     694
Alert.findByEntity         0      0      0
....

```

This listing shows each cache, its size, and its hit counts. At the end of the listing there is a summary for the total size of all Hyperic caches.

If a cache has an unusually high miss rate or becomes full, you can manually configure it to improve performance. For example, consider this cache:

```

Measurement.findByTemplateForInstance      10000      6766      25772

```

The cache has 10,000 elements. With only 6766 hits and 25,772 misses, the hit ratio is around 20-25%. Ideally the number of misses should peak at about the maximum size of the cache. (There are some exceptions to this: the UpdateTimestampsCache and the PermissionCache have items invalidated from the cache frequently, so these rules do not apply.) So, increasing this cache's size would probably improve Hyperic performance.

## Changing Caches Settings

After you identify a cache whose configuration should be changed, you can change its configuration in the file `server-n.n.n-EE\hq-engine\hq-server\webapps\ROOT\WEB-INF\ehcache.xml`. In general, only the cache sizes should need to be changed.

To continue with the example cache from above, the entry for the cache in this file looks like this:

```

<cache name="DerivedMeasurement.findByTemplateForInstance"
  maxElementsInMemory="10000"
  eternal="true"
  timeToIdleSeconds="0"
  timeToLiveSeconds="0"
  memoryStoreEvictionPolicy="LRU" />

```

You may need to iterate on the cache size to find the optimal setting.

The format of this file is described in EHCACHE's documentation.

## Configuring Permission Cache for Improved Performance \*

The PermissionCache can be changed to improve performance. This cache caches recently evaluated permission checks:

```

<cache name="PermissionCache"
  maxElementsInMemory="1000"
  timeToIdleSeconds="0"
  timeToLiveSeconds="60"
  memoryStoreEvictionPolicy="LRU" />

```

This cache is set to expire the checks after 60 seconds. In an environment where the user and role definitions are not updated frequently, you can increase the `timeToLiveSeconds` value to effect a significant performance increase.

While the right value for `timeToLiveSeconds` depends on the environment, the rule of thumb is that it should be set no higher than the acceptable time for a permission-related operation to take effect. Perhaps this is even zero, but that can be costly because the Server needs to go to the database to check permission on every resource the user views.

## 3.6. Integrate HQ Server with Other Systems

These topics have instructions for enabling HQ Server to communication with other enterprise systems:

- [Section 3.6.1, “Configure LDAP Properties”](#)
- [Section 3.6.2, “Configure Kerberos Properties”](#)
- [Section 3.6.3, “Configuring Hyperic Server for SMTP Server”](#)
- [Section 3.6.4, “Enable vFabric Hyperic to Send SNMP Traps”](#)
- [Section 3.6.5, “Configure HQ Server for Hyperic IQ”](#)

### 3.6.1. Configure LDAP Properties

You configure HQ Server for your LDAP server in the "LDAP Configuration Properties" section of the **Administration > HQ Server Settings** page.

By default Hyperic HQ uses its database for storing information about users and authenticating those users. HQ can be configured to use an external LDAP directory for authenticating users in addition to its own database.

In HQ Enterprise, these properties configure HQ to use an external LDAP directory — in addition to HQ's own database — for authenticating users.

Property	Description
Use LDAP Authentication	Checkmark this option to enable LDAP authentication.
URL	Location of your LDAP or Active Directory server. If other than the standard LDAP port is used, specify it the URL. Add the port to the end of the URL, after a colon (:) character. For example: <code>ldap://YourLDAPHost:44389</code>
SSL	Indicates whether the LDAP directory requires SSL connections.
Username and Password	Used if the LDAP directory does not allow anonymous searching, as is common in secure environments. The username must be an LDAP user with sufficient privileges to view at least the sections of the directory containing the information for HQ users. The full node path is required, for example: <code>cn=admin,dc=example,dc=com</code>
Search Base	Also known as the suffix. Required for an LDAP connection. The full path to the branch required, for example: <code>ou=people,dc=example,dc=com</code> If you are unsure of this setting, check with your LDAP administrator.
Search Filter	Limits the users in LDAP to a subset of the entire directory. For example, <code>(!(location=SFO*))</code>
Login Property	The LDAP property that HQ will use as the username. Very important. Examples of common login properties are "cn" and "uid".

After LDAP authentication has been successfully configured, users will be able to log into HQ with their LDAP password (using the value specified as the Login Property as their username). The first time LDAP users log in to HQ, they will be asked to provide some identifying information before they can continue: their first and last name, email address, phone number, etc. This is for display purposes and alert notification purposes.

**Note:** LDAP users must also have one or more HQ roles. As with all new users in HQ, LDAP users are not assigned to any roles by default and therefore they will not be able to see any resources in HQ until they are added to a role. HQ administrators will need to assign the appropriate role or roles to the user in HQ. (On the **HQ Administration** page, click **List Users**, select the user, and click **Add to List** in the "Roles Assigned To" section of the page.)

### 3.6.2. Configure Kerberos Properties

In HQ Enterprise, these properties on the **Admin > HQ Server Settings** page configure HQ to use Kerberos authentication.

- **Realm** - Identifies the Kerberos realm.
- **KDC** - Identifies the Kerberos kdc
- **Debug** - Enables debug logging.

### 3.6.3. Configuring Hyperic Server for SMTP Server

Hyperic sends emails using the SMTP server specified during Hyperic Server installation. On many Unix and Linux machines, the default — `localhost` is satisfactory. In this case, no additional configuration is required. To use a remote SMTP server, you configure the Hyperic Server with the remote host connection information, and set up authentication in `hq-server.conf`.

1. Define SMTP properties in the "Email Settings section" of the `<HQ Server directory>/conf/hq-server.conf` file. As installed, `hq-server.conf` does not contain the mail properties - you must add the properties to override Hyperic's default settings. The properties you define depend on whether you wish to use plain text or SSL communication. Note that changes to `hq-server.conf` take effect only after restart Hyperic Server restart.

- To configure plain text communication, add the mail properties shown below to `hq-server.conf`. Hyperic's default behavior is equivalent to the values shown below - replace the values as appropriate for your environment.

```
# Change to the SMTP gateway server
# maps to mail.smtp.host,
server.mail.host=localhost
# Change to SMTP port
mail.smtp.port=25
# SMTP properties
mail.smtp.auth=false
mail.smtp.socketFactory.class=javax.net.SocketFactory
mail.smtp.socketFactory.fallback=false
mail.smtp.socketFactory.port=25
mail.smtp.starttls.enable=false
```

- To configure SSL communication, define the following properties:

```
server.mail.hostserver.mail.host=SmtServerHost
mail.user=SmtUser
mail.password=SmtPassword
mail.smtp.port=587
mail.smtp.auth=true
mail.smtp.socketFactory.class=javax.net.ssl.SSLSocketFactory
mail.smtp.socketFactory.fallback=false
mail.smtp.socketFactory.port=465
mail.smtp.starttls.enable=true
mail.user=EAM Application
mail.password=password
mail.smtp.port=587
mail.smtp.auth=true
mail.smtp.socketFactory.class=javax.net.ssl.SSLSocketFactory
mail.smtp.socketFactory.fallback=false
mail.smtp.socketFactory.port=465
mail.smtp.starttls.enable=true
```

2. Add the SMTP Server's TLS certificate to the JRE keystore:

- a. Obtain a copy of the public certificate for the SMTP server's TLS configuration (not the private key) on the Hyperic Server.
- b. As the user that owns the Hyperic installation, execute the following in the server installation directory:

```
jre/bin/keytool -keystore jre/lib/security/cacerts -import -storepass changeit -file /
path/to/smtp_server_tls.cert
```

- c. When asked if you want to trust the certificate, answer "yes."

**Note:** The certificate import example above assumes the use of a JRE that is bundled with the Hyperic Server. When using a non-bundled JRE, use that JRE's keytool and cacerts file. For more information, see [Updating SSL Certs for Hyperic Server](#)

### 3.6.4. Enable vFabric Hyperic to Send SNMP Traps

Available only in **vFabric Hyperic**

- [Configure HQ Server to Send SNMP Traps](#)
- [Configure HQ Server Enterprise for SNMP v1](#)
- [Configure HQ Server Enterprise for SNMP v2c](#)
- [Configure HQ Server Enterprise for SNMP v3](#)
- [Using SNMP Traps in Alert Definitions](#)

This page has information about enabling vFabric Hyperic to send SNMP traps to an SNMP management system.

**Note:** For information about enabling Hyperic to *receive traps*, see [Configuring HQ as an SNMP Trap Receiver](#).

#### Configure HQ Server to Send SNMP Traps

1. Click **HQ Server Settings** on the **Administration** page.
2. At the bottom of the page, in the "SNMP Server Configuration Properties" section, define the properties for your version of SNMP. See the appropriate section below.

#### Configure HQ Server Enterprise for SNMP v1

Select "v1" from the **SNMP Protocol Version** pulldown and supply values for the properties defined in the table below.

The table below defines the properties for configuring HQ Server for SNMP V1 communications with an NMS.

Configuration Option	Description	Allowable Values
SNMP Trap OID	The OID of the notification to be sent. Supplies the value of <code>snmpTrapOID.0</code> - the second varbind in a trap or inform that HQ Server generates. (The first varbind is <code>sysUpTime.0</code> .)	
Default Notification Mechanism	Your selection governs the notification type that will appear as the default notification type option in the "Notification Mechanism" pull-down list that is presented in configuration dialogs when user configures an SNMP notification as an alert action, or as a step in an escalation.	For v1 of the SNMP protocol, choose V1 Trap. This is the only trap type you can generate for SNMP v1.
Enterprise OID	Enterprise OID.	
Community	The community name to be sent with the trap.	
Generic ID	Single digit identifier of the trap type.	0 - coldStart 1 - warmStart

Configuration Option	Description	Allowable Values
		2 - linkDown 3 - linkUp 4 - authenticationFailure 5 - egpNeighborLoss 6 - enterpriseSpecific
Specific ID	The specific trap code for an enterprise-specific trap (when <b>Generic ID</b> is set to 6).	
Agent Address	Address of the managed object that generates the trap.	

### Configure HQ Server Enterprise for SNMP v2c

Configuration Option	Description	Allowable Values
SNMP Trap OID	The OID of the notification to be sent. Supplies the value of <code>snmpTrapOID.0</code> - the second varbind in a trap or inform that HQ Server generates. (The first varbind is <code>sysUpTime.0</code> .)	
Default Notification Mechanism	Specifies the default notification type that will appear in configuration dialogs when an authorized user configures an SNMP notification as an alert action, or as a step in an escalation. This choice simply defines the default option - the user configuring an alert action or escalation can choose a different message type.	<ul style="list-style-type: none"> <li>• V1 Trap</li> <li>• V2c Trap</li> <li>• Inform</li> </ul>
Community	The community name to be sent with the trap.	

### Configure HQ Server Enterprise for SNMP v3

This section lists the properties for enabling HQ Enterprise to send SNMP notifications to an NMS. When HQ is so enabled, you can use SNMP notifications in alert definitions - as alert actions and escalation steps.

Configuration Option	Description	Allowable Values
SNMP Trap OID	The OID of the notification to be sent. Supplies the value of <code>snmpTrapOID.0</code> - the second varbind in a trap or inform that HQ Server generates. (The first varbind is <code>sysUpTime.0</code> .)	
Default Notification Mechanism	Specifies the default notification type that will appear in configuration dialogs when an authorized user	<ul style="list-style-type: none"> <li>• V1 Trap</li> <li>• V2c Trap</li> </ul>



Configuration Option	Description	Allowable Values
	er configures an SNMP notification as an alert action, or as a step in an escalation. This choice simply defines the default option - the user configuring an alert action or escalation can choose a different message type.	<ul style="list-style-type: none"> <li>• Inform</li> </ul>
Security Name	The username HQ's SNMP agent should use when sending notifications to the NMS.	Required.
Local Engine ID	ID of HQ's SNMP agent; this value appears automatically, and is not user-configurable.	
Auth Protocol	The SNMP authentication protocol HQ Server should use for communications with the NMS.	<ul style="list-style-type: none"> <li>• none</li> <li>• MD5</li> </ul>
Auth Passphrase	The SNMP authorization passphrase configured for use when communication with the NMS.	<ul style="list-style-type: none"> <li>• SHA</li> </ul>
Privacy Protocol	The SNMP Privacy Protocol HQ Server should use for communication with the NMS.	<ul style="list-style-type: none"> <li>• none</li> <li>• DES</li> <li>• 3DES</li> <li>• AES-128,</li> <li>• AES-192</li> </ul>
Privacy Passphrase	The SNMP privacy passphrase configured for use when communication with the NMS.	<ul style="list-style-type: none"> <li>• AES-256</li> </ul>
Context Engine ID	The EngineID of the NMS. This, along with Context Name, identifies the SNMP context for accessing management data.	Required for v1 and v2c traps. Do not supply for Inform.
Context Name	The name of the SNMP context that provides access to management information on the NMS. A context is identified by the Context Name and Context Engine ID.	

## Using SNMP Traps in Alert Definitions

After the configuration above is complete, the "SNMP Trap" notification tab is available when you define or edit an alert definition.

### 3.6.5. Configure HQ Server for Hyperic IQ

The **Hyperic IQ Server URL** property, on the **Administration > Server Settings** page identifies the URL of a Hyperic IQ installation. You must specify the IQ server to enable users to view IQ reports in the Hyperic IQ Reports dashboard portlet. The URL is:

```
http://IQ_host:9080/arc
```

where *IQ\_host* is the address of the IQ server.

## 3.7. Clustering HQ Servers for Failover

Available only in **vFabric Hyperic**

- [Section 3.7.1, “Overview”](#)
- [Section 3.7.2, “Requirements for an Failover Deployment”](#)
- [Section 3.7.3, “Configuring a Server Cluster”](#)
  - [Step 1: Install the First HQ Server Instance](#)
  - [Step 2: Install Additional HQ Server Nodes](#)
  - [Step 3: Configure Cluster Name and Communications Properties](#)
  - [Step 4: Configure the Load Balancer](#)
  - [Step 5: Configure Agents to Communicate with HQ Server Cluster](#)
  - [Step 6: Start the Nodes](#)
  - [Troubleshooting a Failover Configuration](#)

### 3.7.1. Overview

To avoid interruption of HQ Server operation in the case of failure, you can configure a cluster of HQ Servers. The failover configuration uses:

- EHCACHE's distributed caching for replicating changes throughout the cluster.
- The `nodeStatus.hqu` plugin for monitoring the availability of nodes.
- A hardware load balancer for managing failover when an node becomes unavailable. The load balancer checks the status of each node every 10 seconds, by issuing an HTTP request to the node's `nodeStatus.hqu` plugin. The check will return a response of `master=true` for the primary node, and a response of `master=false` for other nodes in the cluster.

A HQ Server cluster contains multiple nodes; two are generally sufficient. One HQ Server, automatically selected by HQ, serves as the primary node. The other node or nodes serve as hot backups---they do not share the workload with the primary node.

A failover configuration is transparent to users and HQ administrators; it is not apparent that the active HQ server instance is clustered, or which node is currently active.

### 3.7.2. Requirements for an Failover Deployment

- A hardware-based load balancer.
- Only one HQ Server in an HQ Server cluster should receive agent communications at a time. The load balancer should not direct agent connections to an HQ server instance that serves as the secondary node.
- Database Considerations---All nodes in the HQ cluster must share the same database. You cannot use HQ's internal PostgreSQL database in a failover configuration. You must use an external database; MySQL, Oracle, and PostgreSQL are supported.

### 3.7.3. Configuring a Server Cluster

These instructions assume that you do not already have an HQ Server installation.

#### Step 1: Install the First HQ Server Instance

Run the full installer in the appropriate mode for the type of database server you will use (-mysql, -postgresql, or -oracle). You **must** choose one of these options, because clustering requires the use of an external HQ database. The installer will create the HQ database schema.

#### Step 2: Install Additional HQ Server Nodes

For each additional node:

- Run the full installer in the appropriate mode for the type of database created during installation of the first server instance (-mysql, -postgresql, or -oracle).
- When the installer prompts for the location of the HQ database, specify the location of the database created for the first server instance.
- When the installer asks if you want to upgrade, overwrite, or exit the process, select the choice for "upgrade".

#### Step 3: Configure Cluster Name and Communications Properties

Configure the cluster-related properties on each of the HQ Servers in the cluster, in the "Cluster Settings" section of its conf/hq-server.conf file.

#### Default hq-server.conf File

```
# Cluster Settings
#####
#
# Property: ha.partition
#
# This property defines the name of the HQ cluster. Each HQ server with the
# same ha.partition name will join the same cluster. This property is required
# for proper cluster initialization.
#
#ha.partition=

#
# Property: ha.node.address
#
# This property defines the IP address or hostname to bind the multicast listener
# to. This property is required for proper cluster initialization.
#
#ha.node.address=

#
# Property: ha.node.mcast_addr
#
# This property defines the multicast address to use. This property is not required
# and defaults to 238.1.2.3.
#
#ha.node.mcast_addr=238.1.2.3

#
# Property ha.node.mcast_port
```

```
#
# This property defines the multicast port to use. This property is not required
# and defaults to 45566.
#
#ha.node.mcast_port=45566
#
# Property ha.node.cacheListener.port
#
# This property defines the multicast port that is used to discover cache peers. This
# property is not required and defaults to 45567
#ha.node.cacheListener.port=45567
#
# Property ha.node.cacheProvider.port
#
# This property defines the multicast port that is used to synchronize caches throughout
# the HQ cluster. This property is not required and defaults to 45568.
#ha.node.cacheProvider.port=45568
```

## Required Cluster Properties

For each HQ Server in the cluster you must specify:

ha.partition	Name of the cluster---this value is identical for each node in the cluster
ha.node.address	Multicast listen address---specifies IP address or host-name upon which the node listens for multicast traffic; this value is unique to each node in the cluster.

**Note:** If you are upgrading from a pre-v3.0 failover configuration, the each server's .conf file will contain obsolete cluster properties, including server.cluster.mode and server.ha.bind\_addr properties. Delete these properties and replace with the current failover properties described below.

## Optional Cluster Properties

If desired, you can control these communication behaviors for the nodes in the cluster:

ha.node.mcast_addr and ha.node.mcast_port	Address and port for sending multicast messages to other nodes. Note: ha.node.mcast_addr must be the same on each node.
ha.node.cacheListener.port and ha.node.cacheProvider.port	Ports used for discovering and synchronizing with cache peers.

## Step 4: Configure the Load Balancer

Configure the load balancer, according to the vendor or supplier instructions. Procedures vary, but at a minimum you will identify the HQ Server nodes in the cluster and the failover behavior.

1. Identify the Hyperic Server nodes in the cluster.
2. Configure the load balancer to check the nodeStatus.hqu URL every 10 seconds. For example, in a 2-node cluster, if the the IP addresses of the nodes are 10.0.0.1 and 10.0.0.2, configure the load balancer to check these URLs every 10 seconds:

- `http://hqadmin:hqadmin@10.0.0.1:7080/hqu/health/status/nodeStatus.hqu`

- `http://hqadmin:hqadmin@10.0.0.2:7080/hqu/health/status/nodeStatus.hqu`

3. Configure the load balancer to direct all traffic to the node whose status is `master=true`.

## Step 5: Configure Agents to Communicate with HQ Server Cluster

The HQ Agents in your environment communicate with the HQ Server cluster through the load balancer. When you startup a newly installed agent, either supply the load balancer listen address and port interactively, or specify the connection information in `agent.properties`.

For existing agents, you can run `hq-agent.sh setup`, to force the setup dialog.

## Step 6: Start the Nodes

Start the HQ Servers.

## Troubleshooting a Failover Configuration

This section describes the most common sources of problems the failover configuration.

- Multicast blocking---The cluster detection and cache peer detection relies on multicast. Make sure your router isn't blocking multicast packets; otherwise the HQ cluster will fail to initialize properly. It's also common for virtualization technologies like VMware and Xen to not enable multicast by default.
- Don't register agents using the loopback address---If install an HQ Agent on the same machine as an HQ Server node, when you specify the IP address the server should use to contact the agent, don't specify loopback address (127.0.0.1).
- Alerts that were currently firing or in escalation were "lost"--A failover to another cluster node occurred in the middle of the alerts being fired or escalated. The alert state could be lost.

## 3.8. Hyperic Server Properties

- [Section 3.8.1, “About Hyperic Server Configuration”](#)
  - [Configuration Settings in hq-server.conf](#)
  - [Configuration Settings in the Database](#)
- [Section 3.8.2, “Server Property Definitions”](#)
  - [server.connection-validation-sql](#)
  - [server.database](#)
  - [server.database-blockingtimeout](#)
  - [server.database-driver](#)
  - [server.database-maxpoolsize](#)
  - [server.database-minpoolsize](#)
  - [server.database-password](#)
  - [server.database-url](#)
  - [server.database-user](#)
  - [server.encryption-key](#)
  - [server.hibernate.dialect](#)
  - [server.java.opts](#)
  - [server.jms.highmemory](#)
  - [server.jms.maxmemory](#)
  - [server.mail.host](#)
  - [server.quartzDelegate](#)
  - [server.webapp.port](#)
  - [server.webapp.secure.port](#)
- [Clustering Properties in Hyperic Enterprise](#)

## 3.8.1. About Hyperic Server Configuration

### Configuration Settings in `hq-server.conf`

`hq-server.conf` contains the configuration settings that Hyperic Server requires to start up and get ready for work. For instance, `hq-server.conf` has properties that tell the server how to connect to the database and where to listen for agent and web application communications.

When you install Hyperic Server, the selections you can make - port selections, use of plaintext or SSL communications, and so on - correspond to properties in `hq-server.conf`. The configuration settings you supply during installation are persisted in `ServerHome/conf/hq-server.conf`.

In addition to the properties that reflect installation choices, `hq-server.conf` contains properties with default values that you can modify, after installation, based on the your environment and the size of your Hyperic deployment. For example, there are properties in `hq-server.conf` that set defaults for database and JMS configuration options.

Each time Hyperic Server starts up, it reads the values of the properties in `hq-server.conf`.

**Note:** Hyperic Server supports some properties that do not appear in `hq-server.conf` unless you add them explicitly.

After you change the values of properties in `hq-server.conf` or add new properties to the file, you must restart the server for the new settings to take effect.

### Configuration Settings in the Database

Some of the configuration data that governs Hyperic Server behavior is stored in the Hyperic Server database. For example, the data Hyperic Server needs to contact an Hyperic Agent is stored in in the Hyperic Database. For information about how Hyperic Server obtains Hyperic Agent address information, see [Section 2.1.2, “Agent Server Communications Diagram”](#).

## 3.8.2. Server Property Definitions

### `server.connection-validation-sql`

#### Description

The SQL query to run in order to validate a connection from the pool.

#### Default

```
server.connection-validation-sql=select 1
```

### `server.database`

#### Description

The kind of database the HQ server will use. The HQ server adjusts its interactions with the database according to the value of this property.

Valid values are:

- PostgreSQL



- Oracle8
- Oracle9i

**Default**

PostgreSQL

**server.database-blockingtimeout****Description**

Maximum time in milliseconds to wait for a connection from the pool.

**Default**

10000

**server.database-driver****Description**

The JDBC driver to use. You shouldn't change this unless you really know what you're doing.

**Default**

None. The value is set as a result of the database selected during HQ Server installation.

**server.database-maxpoolsize****Description**

The maximum number of database connections to keep in the pool. This must be set lower than the total number of connections allowed to the backend database.

**Default**

100

**server.database-minpoolsize****Description**

The minimum number of database connections to keep in the pool

**Default**

5

**server.database-password****Description**

The database user's password.

**Default**

hqadmin

**server.database-url****Description**

The JDBC URL to connect to.

**Default**

None. The value is set as a result of the database selected during HQ Server installation.

If you select...	the default database URL is...
HQ Built-in Databasel PostgreSQL	postgresql://127.0.0.1:9432/hqdb? protocolVersion=2
Oracle	jdbc:oracle:thin:@localhost:1521:HYPERIC_HQ
MySql	jdbc:mysql://localhost:3306/HQ
PostgreSQL (external)	jdbc:postgresql://localhost:5432/HQ? protocolVersion=2

**server.database-user****Description**

The database user to connect as.

**Default**

hqadmin

**server.encryption-key****Description**

The key for decrypting the HQ database user password. The key must be at least 8 characters long, and can contain letters and numbers.

**Default**

None. The HQ installer prompts for `server.encryption-key` during HQ Server installation.

**server.hibernate.dialect****Description**

The database-specific dialect class used by Hibernate in HQ

**Default**

`org.hyperic.hibernate.dialect.PostgreSQLDialect`

**server.java.opts****Description**

Additional options to pass to Java.

**Default**

```
-Djava.awt.headless=true -XX:MaxPermSize=192m -Xmx512m -Xms512m -XX:+Heap-  
DumpOnOutOfMemoryError
```

For information about using different Java options when starting Hyperic Server, see [Increasing Java Heap and Changing GC Settings](#).

For information about how to change Java heap settings on Windows, see [Changing Heap Size on Windows](#).

## server.jms.highmemory

### Description

The high memory mark for the JMS queue.

### Default

350

## server.jms.maxmemory

### Description

Configures the JMS broker memory limit.

If the broker memory limit is reached, the broker will block the `send ( )` call until some messages are consumed and space becomes available on the broker.

The recommended setting for `server.jms.maxmemory` is 90% of the Java heap size. Erratic alert behavior or missed alerts may indicate the settings are too low.

### Default

400

## server.mail.host

### Description

The IP or hostname of the SMTP server that the HQ server will use for sending alerts and other HQ-related emails. Most UNIX platforms have a local SMTP server, in which case localhost or 127.0.0.1 can be used here.

### Default

127.0.0.1

## server.quartzDelegate

### Description

The database-specific plugin class used by HQ's internal scheduler service.

If you use Oracle as your HQ database, specify:

```
org.quartz.impl.jdbcjobstore.oracle.OracleDelegate
```

If you use the either HQ's internal PostgreSQL database or an external PostgreSQL database as your HQ database, specify:

```
org.quartz.impl.jdbcjobstore.PostgreSQLDelegate
```

**Default**

None. The value is set as a result of the database selected during HQ Server installation.

**server.webapp.port****Description**

The HTTP listen port. This is for the HQ web-based GUI and also HQ agents that communicate with the HQ server in non-secure mode.

**Default**

7080

**server.webapp.secure.port****Description**

The HTTPS listen port. This is for the HQ web-based GUI and also HQ agents that communicate with the HQ server in secure mode.

**Default**

7443

**Clustering Properties in Hyperic Enterprise**

For information about properties for configuring an Hyperic Server cluster, see Clustering Hyperic Servers for Failover.

## 3.9. HQ Database Table Schemas

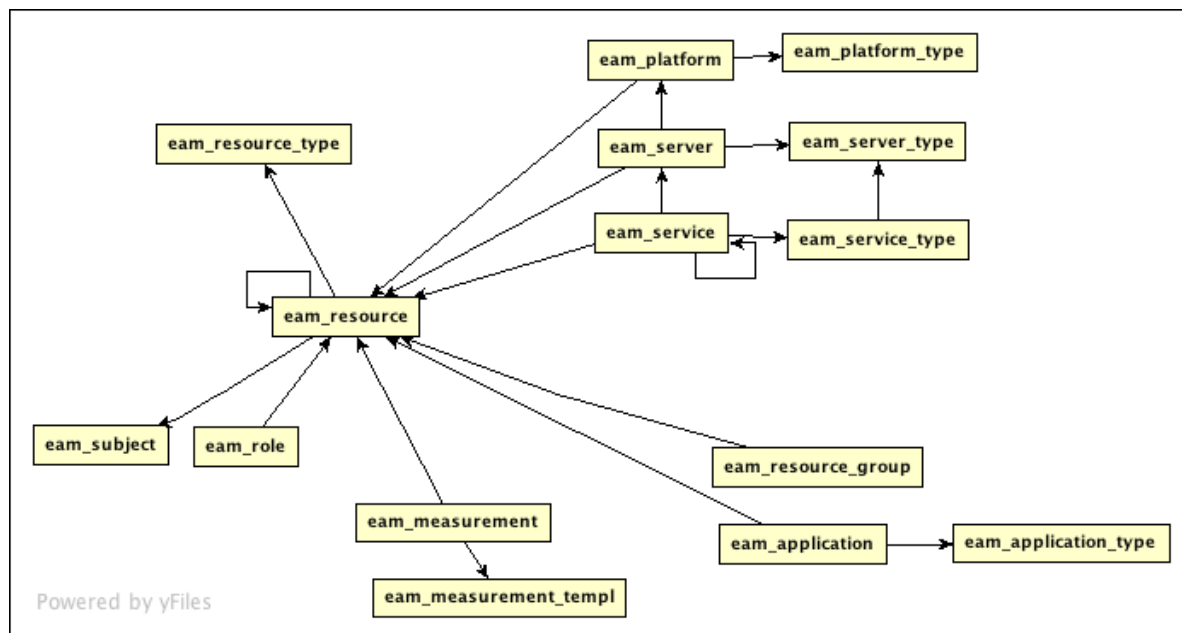
Topics marked with\* relate to features available only in vFabric Hyperic.

- [Section 3.9.1, “Key Resource and Measurement Tables”](#)
  - [EAM\\_RESOURCE Table: All Resource Types and Instances](#)
  - [Tables for Inventory Resources](#)
  - [Tables for Platform, Server, and Service Types](#)
  - [Tables for Measurement Information](#)
- [Section 3.9.2, “Table Documentation”](#)
  - [EAM\\_PLATFORM](#)
  - [EAM\\_PLATFORM\\_TYPE](#)
  - [EAM\\_SERVER](#)
  - [EAM\\_SERVICE](#)
  - [EAM\\_RESOURCE](#)

- [EAM\\_MEASUREMENT](#)
- [EAM\\_MEASUREMENT\\_TEMPL](#)

### 3.9.1. Key Resource and Measurement Tables

This page has information about key HQ database tables that contain information about resources, metric collection, and measurements.



#### EAM\_RESOURCE Table: All Resource Types and Instances

The EAM\_RESOURCE table contains information about the types in the [HQ Inventory Model](#) and instances of those types in the database. This table has a row for *every* managed resource in the HQ database, including

- Operating system platforms, and the servers and services that run on them.
- Virtual or network host platforms, and the servers and services that run on them.
- Groups and applications
- Roles and users
- Escalations

#### Tables for Inventory Resources

The following tables have information about resource instances of a particular inventory type:

- EAM\_PLATFORM - Contains a row for each platform in inventory.
- EAM\_SERVER - Contains a row for each server in inventory.
- EAM\_SERVICE - Contains a row for each service in inventory.
- EAM\_RESOURCE\_GROUP - Contains a row for each group in inventory.
- EAM\_APPLICATION - Contains a row for each application in inventory.

## Tables for Platform, Server, and Service Types

The following tables have information about resource types for an inventory type.

- EAM\_PLATFORM\_TYPE - Contains a row for every platform type that HQ can manage.
- EAM\_SERVER\_TYPE - Contains a row for every server type that HQ can manage.
- EAM\_SERVICE\_TYPE - Contains a row for every service type that HQ can manage.

## Tables for Measurement Information

The following tables have information about the measurements that HQ can collect.

EAM\_MEASUREMENT\_TEMPL - Contains a row for a every metric available for every inventory resource type with its metric template and default metric collection settings.

EAM\_MEASUREMENT - Contains a row for every metric available for every resource in inventory, with metric collection configuration information: whether collection is enabled and the collection interval for enabled metrics.

**Note:** These tables do not store metric values. Metric data is stored in the EAM\_MEASUREMENT\_DATA\_1H, EAM\_MEASUREMENTt\_DATA\_6H, and EAM\_MEASUREMENT\_DATA\_1D tables.

## 3.9.2. Table Documentation

The sections below document the structure for key resource and measurement tables in the HQ database.

### EAM\_PLATFORM

Contains a row for each platform in inventory. Click the thumbnail to see example column data.

The screenshot shows the 'Table: eam\_platform' in the HQ/hqdb/public/TABLE/eam\_platform schema. The table has 15 columns: id, version\_col, fqdn, certdn, cid, description, ctime, mtime, modified\_by, location, comment\_text, cpu\_count, platform\_type\_id, config\_response\_id, agent\_id, and resource\_id. A single row of data is displayed with the following values: 1, 10001, 2 Marie-McGarrys-..., CAM-AGENT-..., (null), Mac OS X Snow Le..., 12747..., 1274900..., hqadmin, SF, comment\_text, 1, 10006, 10001, 10001, 10779.

Field	Type	Description
ID	int4	An ID for the platform, unique among platforms.
VERSION_COL	int8	Version of the row. Increments when the row is modified. increments with any change to configuration of this row
FQDN	varchar(200)	Fully qualified domain name of the platform.
CERTDN	varchar(200)	SSL Certificate for the agent which is monitoring this platform.
CID	int4	not used
DESCRIPTION	varchar(256)	Description of platform.
CTIME	int8	Creation time of platform.

MTIME	int8	Last modification time of the platform.
MODIFIED_BY	varchar(100)	Last modification user.
LOCATION	varchar(100)	String entered by user, optionally.
COMMENT_TEXT	varchar(256)	String entered by user, optionally.
CPU_COUNT	int4	Number of CPUs on this platform.
PLATFORM_TYPE_ID	int4	ID for the platform type. Points to EAM_PLATFORM_TYPE table.
CONFIG_RESPONSE_ID	int4	Link to configuration string in plugin xml file.
AGENT_ID	Int4	A unique identifier to the agent which is monitoring this platform.
RESOURCE_ID	int4	Uniquely identifies the resource, unique across platforms, servers, services. Points to the EAM_RESOURCE table.

## EAM\_PLATFORM\_TYPE

Contains a row table for each HQ-supported platform type. Click the thumbnail to see column names and example column data.

Table: eam\_platform\_type  
HQ/hqdb/public/TABLE/eam\_platform\_type

Info Columns Data Row Count Primary Key Indexes Grants Row Id References

	id	version_col	name	sort_name	cid	description	ctime	mtime	os	osversion	arch	plugin
1	10001	0	Cisco PIXOS	CISCO PIXOS	(null) (null)		1274728129055	1274728129055	(null)	(null)	(null)	netdevice
2	10002	0	Cisco IOS	CISCO IOS	(null) (null)		1274728129119	1274728129119	(null)	(null)	(null)	netdevice
3	10003	0	Network Host	NETWORK HOST	(null) (null)		1274728129246	1274728129246	(null)	(null)	(null)	netdevice
4	10004	0	Network Device	NETWORK DEVICE	(null) (null)		1274728129278	1274728129278	(null)	(null)	(null)	netdevice
5	10005	0	Linux	LINUX	(null) (null)		1274728131003	1274728131003	(null)	(null)	(null)	system
6	10006	0	MacOSX	MACOSX	(null) (null)		1274728131012	1274728131012	(null)	(null)	(null)	system
7	10007	0	FreeBSD	FREEBSD	(null) (null)		1274728131027	1274728131027	(null)	(null)	(null)	system
8	10008	0	NetBSD	NETBSD	(null) (null)		1274728131039	1274728131039	(null)	(null)	(null)	system
9	10009	0	Win32	WIN32	(null) (null)		1274728131085	1274728131085	(null)	(null)	(null)	system
10	10010	0	AIX	AIX	(null) (null)		1274728131109	1274728131109	(null)	(null)	(null)	system
11	10011	0	HPUX	HPUX	(null) (null)		1274728131126	1274728131126	(null)	(null)	(null)	system
12	10012	0	Solaris	SOLARIS	(null) (null)		1274728131142	1274728131142	(null)	(null)	(null)	system
13	10013	0	OpenBSD	OPENBSD	(null) (null)		1274728131168	1274728131168	(null)	(null)	(null)	system
14	10014	0	VMware VI3 Host	VMWARE VI3 HOST	(null) (null)		1274728133130	1274728133130	(null)	(null)	(null)	vim
15	10015	0	Xen Host	XEN HOST	(null) (null)		1274728133302	1274728133302	(null)	(null)	(null)	xen

## EAM\_SERVER

Contains a row for each server in HQ inventory.

Field	Type	Description
ID	int4	A unique identifier of the server.
VERSION_COL	int8	A column which increments with any change to configuration of this row.
CID	int4	
DESCRIPTION	varchar(300)	Description of server.



CTIME	int8	Creation time of server.
MTIME	int8	Last modification time of the server.
MODIFIED_BY	varchar(100)	Last modification user
LOCATION	varchar(100)	
PLATFORM_ID	int4	The Unique ID of the platform on which this server is installed
AUTOINVENTORYIDENTIFIER	varchar(250)	A unique ID describing this server via the plugin XML.
RUNTIMEAUTODISCOVERY	bool	Is runtime autodiscovery enabled on this server?
WASAUTODISCOVERED	bool	Was this server autodiscovered?
SERVICESAUTOMANAGED	bool	Not used.
AUTODISCOVERY_ZOMBIE	bool	Were there deletions on the client side for this server?
INSTALLPATH	varchar(200)	Install path of this server on the platform.
SERVER_TYPE_ID	int4	Unique ID of the server type that describes this server.
CONFIG_RESPONSE_ID	int4	Link to configuration string in plugin xml file.
RESOURCE_ID	int4	Uniquely identifies the resource, unique across platforms, servers, services. Points to the EAM_RESOURCE table.

## EAM\_SERVICE

Contains a row for each service in HQ inventory

Field	Type	Description
ID	int4	An ID for the service, unique among services.
VERSION_COL	int8	A column which increments with any change to configuration of this row
CID	int4	
DESCRIPTION	varchar(200)	Description of service.
CTIME	int8	Creation time of service.
MTIME	int8	Last modification time of the service.
MODIFIED_BY	varchar(100)	Last modification user.
LOCATION	varchar(100)	Not used.
AUTODISCOVERY_ZOMBIE	bool	Were there deletions on the client side for this service?

SERVICE_RT	bool	Is response time enabled for this service?
ENDUSER_RT	bool	Is end user response time enabled for this service?
PARENT_SERVICE_ID	int4	Unique ID into the parent service for this service.
SERVER_ID	int4	Were there deletions on the client side for this server?
AUTOINVENTORYIDENTIFIER	varchar(500)	A unique ID describing this server via the plugin XML.
SERVICE_TYPE_ID	Int4	Unique ID of service type for this service.
CONFIG_RESPONSE_ID	int4	Link to configuration string in plugin xml file.
RESOURCE_ID	int4	Uniquely identifies the resource, unique across platforms, servers, services. Points to the EAM_RESOURCE table.

## EAM\_RESOURCE

This table contains a row for each type in the HQ inventory and access control model, and a row for each instance of each type in the HQ database, including:

- basic inventory types - Platforms, Servers, and Services
- configurable inventory types - Groups and Applications
- Users and Roles
- Escalations

Click the thumbnail to see example column data.

Table: eam\_resource  
HQ/hqdb/public/TABLE/eam\_resource

Info Columns Data Row Count Primary Key Indexes Grants Row Id References

	id	version	resource_type_id	instance_id	subject_id	proto..	name	sort_name
797	10774	0	603	10603	0	0	Sendmail 8.x Message Submission Process	SENDMAIL 8.X MESSAGE SUBMISSION PROCESS
798	10775	0	603	10604	0	0	Sendmail 8.x Root Daemon Process	SENDMAIL 8.X ROOT DAEMON PROCESS
799	10776	0	603	10605	0	0	Sendmail 8.x SMTP	SENDMAIL 8.X SMTP
800	10777	0	603	10606	0	0	JBoss 4.2 HQ Internals	JBoss 4.2 HQ INTERNALS
801	10778	0	603	10607	0	0	JBoss 4.0 HQ Internals	JBoss 4.0 HQ INTERNALS
802	10779	0	301	10001	1	10015	Marie-McGarrys-MacBook-Pro-15.local	MARIE-MCGARRYS-MACBOOK-PRO-15.LOCAL
803	10784	0	303	10005	1	10024	Marie-McGarrys-MacBook-Pro-15.local MacOSX FileServer	MARIE-MCGARRYS-MACBOOK-PRO-15.LOCA...
804	10785	0	303	10006	1	10025	Marie-McGarrys-MacBook-Pro-15.local MacOSX ProcessServer	MARIE-MCGARRYS-MACBOOK-PRO-15.LOCA...
805	10780	0	303	10001	1	10026	Marie-McGarrys-MacBook-Pro-15.local MacOSX NetworkServer	MARIE-MCGARRYS-MACBOOK-PRO-15.LOCA...
806	10923	0	305	10136	1	10027	Marie-McGarrys-MacBook-Pro-15.local MacOSX Network Interface lo0...	MARIE-MCGARRYS-MACBOOK-PRO-15.LOCA...
807	10924	0	305	10137	1	10027	Marie-McGarrys-MacBook-Pro-15.local MacOSX Network Interface en0...	MARIE-MCGARRYS-MACBOOK-PRO-15.LOCA...
808	10925	0	305	10138	1	10027	Marie-McGarrys-MacBook-Pro-15.local MacOSX Network Interface en1...	MARIE-MCGARRYS-MACBOOK-PRO-15.LOCA...
809	10788	0	305	10001	1	10032	Marie-McGarrys-MacBook-Pro-15.local MacOSX File System /dev/disk...	MARIE-MCGARRYS-MACBOOK-PRO-15.LOCA...

Field	Type	Description
ID	int4	Uniquely identifies a type or an instance of a type.
VERSION_COL	int8	Increments with any change to configuration of this row.

RESOURCE_TYPE_ID	int4	Identifies a type in the HQ inventory and access control model. The values this attribute identify a resource as one of the following:
INSTANCE_ID	int4	Uniquely identifies a type or an instance of a particular type in the inventory and access control model. For a type, corresponds to the ID column in one of the following tables: EAM_PLATFORM_TYPE, EAM_SERVER_TYPE, EAM_SERVICE_TYPE, EAM_APPLICATION_TYPE, or EAM_RESOURCE_TYPE. For an instance of a type, corresponds to the ID column in one of the following tables: EAM_PLATFORM, EAM_SERVER, EAM_SERVICE, EAM_RESOURCE_GROUP, EAM_APPLICATION, EAM_ROLE, EAM_SUBJECT, EAM_ESCALATION
SUBJECT_ID	int4	Identifies the HQ user owns the resource.
PROTO_ID	int4	For a type, value is zero. For an instance of a type, contains the value of the ID column for the type in this table.
NAME	varchar(500)	Display name for a resource, for example, "My-Office-Mac-Book-Pro-15.local JBoss 4.2 default ServiceManager Stateless Session EJB".
SORT_NAME	varchar(500)	Same as the NAME column but all in upper case, for example, "MY-OFFICE-MAC-BOOK-PRO-15.LOCAL JBOSS 4.2 DEFAULT SERVICEMANAGER STATELESS SESSION EJB".
FSYSTEM	boolean	
MTIME	int8	Last modification time of the resource.

## EAM\_MEASUREMENT

Each row contains information about a measurement for a resource under management. Click the thumbnail to see example column data.

Table: eam\_measurement  
HQ/hqdb/public/TABLE/eam\_measurement

Info Columns Data Row Count Primary Key Indexes Grants Row Id References

	id	version_col	instance_id	template_id	mtime	enabled	coll_int	dsn	resource_id
897	11113	0	10054	20519	1274728694550	false	0	PostgreSQL 8.2 Table:postgresql:Type=Table,table=eam_pl...	10841
898	11114	0	10054	20514	1274728694551	false	0	PostgreSQL 8.2 Table:postgresql:Type=Table,table=eam_pl...	10841
899	11115	0	10054	20513	1274728694554	false	0	PostgreSQL 8.2 Table:postgresql:Type=Table,table=eam_pl...	10841
900	11116	0	10054	20523	1274728694555	false	0	PostgreSQL 8.2 Table:postgresql:Type=Table,table=eam_pl...	10841
901	11117	0	10054	20512	1274728694556	false	0	PostgreSQL 8.2 Table:postgresql:Type=Table,table=eam_pl...	10841
902	12208	1	10119	20521	1274730114027	true	300000	PostgreSQL 8.2 Table:postgresql:Type=Table,table=qrtz_pa...	10906
903	13412	2	10239	17045	1274819749074	true	300000	JBoss 4.2 JMS Destination:jboss.mq.destination:service=Queu...	11026
904	13415	2	10240	17046	1274819750362	true	300000	JBoss 4.2 JMS Destination:jboss.mq.destination:service=Queu...	11027
905	13407	2	10238	17046	1274819749224	true	300000	JBoss 4.2 JMS Destination:jboss.mq.destination:service=Queu...	11025
906	13408	2	10238	17045	1274819749224	true	300000	JBoss 4.2 JMS Destination:jboss.mq.destination:service=Queu...	11025

Max Rows: 1000 Max Chars: 0 0.014/0.002 sec 1000/9 896-906

Field	Type	Description
ID	int4	Unique ID for a metric that can be collected for a resource. Points to actually measurements in EAM_MEASUREMENT_DATA_* tables.
VERSION_COL	int8	Indicates version of the row, increments upon each change to the row.
INSTANCE_ID	int4	The resource type the measurement is for. Uniquely identifies a resource type of a given inventory level - platform, server, service. For example, the ID 10001 uniquely identifies the platform type "MacOSX".
TEMPLATE_ID	int4	ID of a template points to the EAM_MEASUREMENT_TEMPL table.
MTIME	int8	Time modified.
ENABLED	boolean	Is this metric enabled?
COLL_INTERVAL	int8	How often this metric is collected.
DSN	varchar(2048)	A string which describes the measurement from the plugin-xml text.
RESOURCE_ID	int4	Uniquely identifies the resource for which the metric is associated, unique across platforms, servers, services. Points to the EAM_RESOURCE table.

## EAM\_MEASUREMENT\_TEMPL

Contains a row for a every measurement that HQ can collect, for every resource type it can manage, with information about the default metric collection settings.

Field	Type	Description
-------	------	-------------

ID	int4	A unique identifier of a measurement template for a metric for a resource.
VERSION_COL	int8	A column which increments with any change to configuration of this row
NAME	varchar(100)	Name of this measurement template.
ALIAS	varchar(100)	String that describes the alias portion of XML file.
UNITS	varchar(50)	Units of this measurement.
COLLECTION_TYPE	int4	Static/Dynamic data.
DEFAULT_ON	bool	Does this measurement collect by default?
DEFAULT_INTERVAL	int8	The default collection interval of this metric.
DESIGNATE	bool	Is this metric on the indicator page by default?
TEMPLATE	varchar(2048)	Template string from plugin XML.
PLUGIN	varchar(250)	Name of the plugin which houses this measurement template.
CTIME	int8	Creation time of server.
MTIME	int8	Last modification time of the server.
MONITORABLE_TYPE_ID	int4	Key into the monitorable type data.
CATEGORY_ID	int4	Key into the category ID table.

## 4. Troubleshoot Agent and Server Problems

This page has tips for troubleshooting problems in a Hyperic deployment.

- [Section 4.1, “Looking for Clues”](#)
  - [Section 4.1.1, “HQ Health”](#)
  - [Section 4.1.2, “Agent Metrics”](#)
  - [Section 4.1.3, “Log Files”](#)
  - [Section 4.1.4, “Thread Dumps”](#)
  - [Section 4.1.5, “Check Agent Port Availability”](#)
- [Section 4.2, “Hyperic Server Problems”](#)
  - [Section 4.2.1, “Server Startup Problems”](#)
  - [Section 4.2.2, “Backlogged Hyperic Server”](#)
- [Section 4.3, “Agent Startup Problems”](#)
  - [Section 4.3.1, “Agent Failed to Connect to Server at First Startup”](#)
  - [Section 4.3.2, “Agent Start Script Timeout”](#)
  - [Section 4.3.3, “Java Service Wrapper Timeout”](#)
- [Section 4.4, “Invalid or Unknown Availability”](#)
  - [Section 4.4.1, “Out-of-Sync Agent and Server Clocks”](#)
  - [Section 4.4.2, “Overloaded Backend”](#)
  - [Section 4.4.3, “Overloaded Agent”](#)
- [Section 4.5, “Slow User Interface”](#)
- [Section 4.6, “Warning Messages in the Agent Log”](#)
  - [Section 4.6.1, “Connection Timeout Messages”](#)

## 4.1. Looking for Clues

This section describes options for getting information that might help you diagnose problems in an Hyperic deployment.

### 4.1.1. HQ Health

The **HQ Health** page, available in the "Plugins" section of the **Administration** page, displays a variety of metrics and status about your Hyperic deployment, including server host statistics and Hyperic Server process information.

**HQ Health** provides views, queries, and diagnostic tools that provide visibility into metric loads, caches, the Hyperic database, and agents across your deployment.

For more information about the **Agents** tab in **HQ Health**, see [Section 2.9.1, “View Status of all Hyperic Agents”](#). For information about all of the data available on the **HQ Health** page, see its help page.

### 4.1.2. Agent Metrics

Hyperic Agent metrics are helpful in diagnosing many problems that can occur. By default, these metrics are reported:

- Availability
- JVM Free Memory - Indicator
- JVM Total Memory - Indicator
- Number of Metrics Collected Per Minute - Indicator
- Number of Metrics Sent to the Server Per Minute
- Server Offset
- Total Time Spend Fetching Metrics per Minute

Depending on your environment, you may find it useful to track other agent metrics, such as:

In addition to the default metrics

- Number of Metrics which Failed to be Collected
- Number of Metrics which Failed to be Collected per Minute
- Maximum Time Spent Processing a Request
- Number of Connection Failures
- Total Time Spent Fetching Metrics per Minute

For more information about default and available agent metrics see View Agent Metrics.

### 4.1.3. Log Files

The following log files can be a useful source of information in the event that a problem occurs in a Hyperic deployment:

- `ServerHome/logs/wrapper.log`
- `ServerHome/logs/bootstrap.log`
- `ServerHome/logs/server.log`
- `ServerHome/logs/hqdb.log`
- `AgentHome/logs/wrapper.log`
- `AgentHome/logs/agent.log`
- `AgentHome/logs/agent.log.startup`

You can increase the level of agent logging by setting the `agent.logLevel=DEBUG` properties in `agent.properties`. Note however that debug logging is very verbose and uses more system resources. Hyperic recommends running the agent in DEBUG mode only when troubleshooting a problem. For more information, see [Section 2.5, “Configure Agent Logging”](#).

#### 4.1.4. Thread Dumps

This section has instructions for generating thread dumps for the Hyperic Server and Hyperic Agent.

##### Generate a Hyperic Server Thread Dump from User Interface

Follow these steps to output a server thread dump to your browser.

1. Click the **Administration** tab.
2. Click **HQ Health** in the **Plugins** section of the **Administration** page.
3. Click **Print** in the **HQ Process Information** section on the **HQ Health** page.

##### Generate a Hyperic Server Thread Dump from Command Line

- Windows
  - If Hyperic Server is running in a terminal window — Try `<ctrl><break>` in the terminal window.
  - If Hyperic Server is running as a service — Use a tool like [StackTrace](#).
- Unix-like systems
  - Hyperic 4.2, 4.3, and 4.4 — Run `Kill -3` on the HQ server process.
  - Hyperic 4.5 — use [jstack](#) on the Hyperic Server process. For example, if the server's PID is 215:

```
jstack 215 >mydumpfile.txt
```

##### How to find the Hyperic Server PID

You can run `jps` in a shell to determine the Hyperic Server's process ID — look for the process named "Bootstrap". For example:

```
$ jps
```



```
187 WrapperStartStopApp
408 Jps
215 Bootstrap
```

## Generate Agent Thread Dump from User Interface

Run the agent launcher with the dump option. For more information see [Section 2.3, “Start, Stop, and Other Agent Operations”](#)

### 4.1.5. Check Agent Port Availability

The Hyperic Server must be able to access the agent's listen port — you can verify it can by running `telnet <Address> <Port>`. For example:

```
$ telnet 192.168.1.114 2144
```

For a successful connection, the results are similar to:

```
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^\''.
GET
Connection closed by foreign host.
```

## 4.2. Hyperic Server Problems

This section describes problems that could prevent the Hyperic Server from starting up.

### 4.2.1. Server Startup Problems

In vFabric, the Hyperic Server will not start if it does not find a valid `license.xml` file in its `/conf` directory. Check the **HQ License Information** section of the **Administration** page to make sure your license has not expired.

### 4.2.2. Backlogged Hyperic Server

When the Hyperic Server starts after a period of downtime, it can be inundated with metric reports from agents that continued to run while the server was down. When the server is processing a large metric backlog from many agents, the maximum size of the agent-server connection pool size can become a bottleneck and affect server performance.

You can check the "Current Thread Busy" metric for the Hyperic Server's internal Tomcat server to determine whether connection pool size is an issue.

To enable a restarted server catch up with a metric backlog you can increase the maximum size of the agent-server connection pool — typically this is the only time that increasing the size of the connection pool is indicated.

The maximum number of agent-server connection is configured with the `org.hyperic.lather.maxConns` property in `web.xml`. Note however that, in effect, the number of connections is limited by the maximum number of server execution threads, which is configured in the `server.xml` file for Hyperic's internal Tomcat server. So, although the default value of `org.hyperic.lather.maxConns` is 3000, by default the number of agent-server connections is effectively limited to 500 — the default value of `maxThreads` in `server.xml`.

To enable the server to catch up, enable 20% more connections than there are agents reporting to the server. For example, if you have 1000 agents, enable 1200 agent-server connections.

To change the size of the agent-server connection pool:

1. As necessary, update the maximum number of agent-server connections — `org.hyperic.lather.maxConns` — in `<Server installation directory>/hq-engine/hq-server/webapps/ROOT/WEB-INF/web.xml` file, in the stanza shown below. Ensure that the value of `maxConns` is 20% greater than the number of agents that report to the server.

```
<init-param>
<param-name>org.hyperic.lather.maxConns</param-name>
<param-value>3000</param-value>
</init-param>
```

2. As necessary, configure the maximum number of Tomcat threads — `maxThreads` — in the Catalina service element in `<Server installation directory>/hq-engine/hq-server/conf/server.xml` file, excerpted below. Ensure that the value of `maxThreads` greater than the value of `maxConns`.

```
<Service name="Catalina">

<Executor name="tomcatThreadPool" namePrefix="tomcat-http--" maxThreads="500"
minSpareThreads="50"/>
```

3. Restart the Hyperic Server to enable the changes to take effect.

## 4.3. Agent Startup Problems

This section describes problems that could prevent the Hyperic Agent from starting up.

### 4.3.1. Agent Failed to Connect to Server at First Startup

Every time an agent starts up it attempts to contact the Hyperic Server. If, the first time you start up an agent, it cannot connect to the server, the agent will continue to have problems connecting to the server, even after the server is reachable.

The first time a Hyperic Agent successfully connects with the Hyperic Server, the agent saves the server connection settings in its `/data` directory. If the server is not available (because the wrong address/port was configured for the agent, or because the server hasn't been started or is still in the process of starting up) the agent will fail to connect, and hence fail to persist the server connection data. Upon agent restart, the agent will not be able to find the connection data it requires, and fail to connect to the server. In this case, `server.log` will contain a message similar to:

```
2010-04-20 11:04:26,640 ERROR [Thread-1|Thread-1] [AutoinventoryCommandsServer] Unable to send autoinventory platform data to server, sleeping for 33 secs before retrying. Error: Unable to communicate with server – provider not yet setup
```

To solve this problem:

1. Delete the agent's `/data` directory.
  - This forces the agent to obtain new agent - server communication properties.
2. Verify the address and listen port for the Hyperic Server.
  - Is there a firewall between agent and server? See the instructions in [Section 2.2.1, “Configure Agent - Server Communication Interactively”](#) or [Section 2.2.2, “Configure Agent - Server Communication in Properties File”](#).
3. Verify the Hyperic Server is up.
4. Start the agent, supplying the correct server connection properties, either in `agent.properties` or interactively. See the instructions referenced in step 2 above.

### 4.3.2. Agent Start Script Timeout

By default, the agent start script times out after five minutes if the startup sequence is not successful. Check the `agent.log.startup` file for messages.

If desired, you can configure a longer timeout period – to give the agent more time to connect to the server – by adding the `agent.startupTimeout` property, defined below, to the `agent.properties` file.

#### **agent.startupTimeout**

##### **Description**

The number of seconds that the agent startup script will wait before determining that the agent did not startup successfully. If the agent is not determined to be listening for requests within this period of time, an error is logged, and the startup script times out.

**Default**

As installed, `agent.properties` does not contain a line that defines the value of this property. The default behavior of the agent is to timeout after 300 seconds.

After editing the `agent.properties` file, save your changes and restart the agent.

### 4.3.3. Java Service Wrapper Timeout

Under high load, the agent may become unresponsive. If this occurs and there are no coincident errors or warnings in the agent log that indicate another explanation, it may be that the agent JVM was starved for memory, and unresponsive to a ping from the Java Service Wrapper (JSW).

In that case the `wrapper.log` file will contain an entry like this:

```
ERROR | wrapper | 2009/01/15 02:15:18 | JVM appears hung: Timed out waiting  
for signal from JVM.
```

To resolve the problem, you can configure the JSW to give the agent more time to respond to startup and ping requests.

Increase the JSW's timeout period from 30 seconds to 300. To do so, add this property to `AGENT_HOME/bundles/agent-4.x.xxxx/conf/wrapper.conf`.

```
wrapper.ping.timeout=300
```

This will cause the JSW to wait longer for a ping response before determining that the JVM is hung.

Increase the agent's startup timeout from 30 seconds to 300. This will give the agent more time to start up before wrapper gives up on it and kills the process. To do so, add this property value:

```
wrapper.startup.timeout=300
```

to

```
AgentHome/bundles/agent-4.x.xxxx/conf/wrapper.conf
```

## 4.4. Invalid or Unknown Availability

This section describes reasons that Hyperic might incorrectly show an agent (or agent-managed resource) as unavailable, or show its availability status as "Unknown" (availability icon is grey).

### 4.4.1. Out-of-Sync Agent and Server Clocks

If Hyperic erroneously indicates that resources are unavailable, it may be because the system clocks on the agent and server hosts are out-of-sync. By default, Hyperic monitors the offset between the server and an agent — see the "Server Offset" metric on the **Monitor** tab. An offset of less than one minute is unlikely to pose problems; with a larger offset, problems may occur.

To solve an offset problem, install NTP and synchronize system clocks on the agent and server hosts.

To prevent agent and server becoming significantly out-of-sync, you can run NTP on each system, use Hyperic to monitor the offset on each system, and set alerts based on the offset metric. To do so, configure a platform service of type "NTP" to monitor each NTP service, and set alerts to fire when an offset from the time authority grows unacceptably high. For more information, see the "Monitoring Network Services" section in *Configure Resources for Monitoring*.

### 4.4.2. Overloaded Backend

If the Hyperic database cannot process metrics at the rate it receives them, resources that are available may be incorrectly shown as unavailable.

You can view Hyperic Server process statistics, as well as load and utilization data - load, process, system and JVM memory, and so on - on the **HQ Health** page.

Based on your load, it may be appropriate to tune Hyperic Server, the Hyperic database, or both. For more information, see the "Configuring HQ for Large Environments and Improved Performance" section in *Configure and Run the HQ Server*.

**Important:** First and foremost: run the Hyperic database on dedicated hardware. Competition for system resources can slow down metric processing.

### 4.4.3. Overloaded Agent

If an agent's queue of metrics grows to a certain level over a period of time, the following warning message is written to the `agent.log` file:

The Agent is having a hard time keeping up with the frequency of metrics taken. Consider increasing your collection interval.

To investigate, you can configure the agent to report the "Total Time Spent Fetching Metrics per Minute" metric — if the agent spends more than half its time fetching metrics — it is overloaded.

You can alleviate the problem by

- Increase metric collection intervals — For most metrics, the default is every 5 or 10 minutes. You can change the collection interval for all metrics collected for a resource type on the **Administration > Monitoring Defaults** page for the resource type.

#### Want to Change a Whole Bunch of Metric Collection Intervals?

If you want to change metric templates in bulk, or without using the Hyperic user interface, you can change metric collection settings - including collection intervals - for a resource type with the `HQApi`

**metricTemplate** command. You can use the **metricTemplate sync** option from the command line or in a script, as desired. For more information, see the "HQAapi metricTemplate command" section in *Web Services API*.

- Try to redistribute load — If an agent that logs metric volume warnings is monitoring a large number of remote services over the network (for example, HTTP, FTP, SNMP, or another service type whose protocol the agent supports), you can spread the load around — configure a different agent to monitor some of the network services. You can compare the agent loads on the **Agents** tab of **HQ Health**.

## 4.5. Slow User Interface

If Hyperic's web user interface is slow, the cause may be an overloaded backend - the Hyperic database, or the Hyperic Server itself. See [Overloaded Backend](#).

## 4.6. Warning Messages in the Agent Log

This section has information about the significance of selected warning messages that might be written to the `agent.log` file.

### 4.6.1. Connection Timeout Messages

Lather, the connection protocol for agent-to-server communication, is configured such that agent connections time out after five minutes, and a timeout message is written to `agent.log`. You can increase the timeout period from 300000 to 900000 in this file:

```
hq-engine/server/default/deploy/lather-jboss.sar/jboss-lather.war/WEB-INF/web.xml
```

in this stanza:

```
<init-param>  
<param-name>org.hyperic.lather.execTimeout</param-name>  
<param-value>900000</param-value>  
</init-param>
```