



JBoss Operations Network

3.1

Installation Guide

for installing servers and agents
Edition 3.1.2

Ella Deon Lackey

JBoss Operations Network 3.1 Installation Guide

for installing servers and agents
Edition 3.1.2

Ella Deon Lackey
dlackey@redhat.com

Legal Notice

Copyright © 2012 Red Hat, Inc..

This document is licensed by Red Hat under the [Creative Commons Attribution-ShareAlike 3.0 Unported License](https://creativecommons.org/licenses/by-sa/3.0/). If you distribute this document, or a modified version of it, you must provide attribution to Red Hat, Inc. and provide a link to the original. If the document is modified, all Red Hat trademarks must be removed.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat Software Collections is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This manual covers the installation and setup of JBoss ON 3.1.2 servers and agents and basic tasks for configuring the inventory.

Table of Contents

Preface	3
1. JBoss Operations Network Overview	3
2. Examples and Formatting	3
3. Document History	4
Chapter 1. JBoss Operations Network 3.1.2 Prerequisites	6
1.1. Supported Platforms, Databases, and Other Requirements	6
1.2. Hardware Requirements	6
Chapter 2. Setting up Databases	7
2.1. Configuring PostgreSQL	7
2.2. Setting up Oracle	10
Chapter 3. Installing and Upgrading the JBoss ON Server on Linux	15
3.1. Preparing for Installation	15
3.2. Installing the Server JAR File	16
3.3. Configuring the Server with the Web Installer	17
3.4. Silently Installing the JBoss ON Server	19
3.5. Additional Post-Setup Checklist	22
3.6. Configuring the Server as a Service	24
3.7. Re-Installing the Server	25
3.8. Upgrading JBoss ON	25
Chapter 4. Installing and Upgrading the JBoss ON Server on Windows	36
4.1. Preparing for Installation	36
4.2. Installing the Server JAR File	37
4.3. Going Through the Web Installer	38
4.4. Silently Installing the Server	41
4.5. Additional Post-Setup Checklist	44
4.6. Configuring the Server as a Windows Service	46
4.7. Re-Installing the Server	46
4.8. Upgrading JBoss ON	47
Chapter 5. Installing JBoss Agent Plug-in Packs	58
Chapter 6. Installing the JBoss ON CLI	59
Chapter 7. Troubleshooting Installation and Upgrade	60
7.1. Exceptions and Error Logs	60
7.2. Connection Issues	62
7.3. UI Problems	62
Chapter 8. Installing and Upgrading the Agent from the JAR File	63
8.1. Before Installing the Agent	63
8.2. Installing the Agent from JAR File	65
8.3. Using an Answer File for the Agent Installation	69
8.4. Running the JBoss ON Agent as a Service	70
8.5. Changing Agent Connection Configuration	73
8.6. Installing Multiple Agents with a Shared Directory or Account	73
8.7. About Agent Automatic Updates	75
8.8. Manually Upgrading the JBoss ON Agent	77
8.9. Reinstalling the Agent (JAR)	78
Chapter 9. Installing the Agent from RPM	79

9.1. About Agent RPMs	79
9.2. Installing the Agent from RPM	82
9.3. Changing the Agent Configuration After an RPM Install	83
9.4. Migrating from a JAR Installation to an RPM Installation	85
9.5. Troubleshooting RPM Installs	87
Chapter 10. Uninstalling JBoss ON	88
10.1. Uninstalling an Agent	88
10.2. Uninstalling the Server	89
Index	89

Preface

JBoss Operations Network 3.1.2 provides an integrated solution for managing JBoss middleware, other network infrastructure, and applications built on Red Hat Enterprise Application Platform (EAP).

This manual covers planning and procedures for installing JBoss ON servers and agents and upgrading existing JBoss ON systems. This *Installation Guide* is intended for JBoss ON administrators.

1. JBoss Operations Network Overview

JBoss Operations Network has four major components, which work together to create the management platform:

- The JBoss ON servers, which centralize configuration and connect the components
- An SQL database (PostgreSQL or Oracle) which stores JBoss ON configuration settings and resource-related data, including content packages, the resource inventory, and monitoring data
- Local agents installed on managed platforms, which connect with servers to receive resource configuration updates and which collect and send monitoring data
- The JBoss ON GUI, which is a web-based interface that allows users to connect to any JBoss ON server, from any location, to view resource data and perform management tasks

2. Examples and Formatting

Each of the examples used in this guide, such as file locations and commands, have certain defined conventions.

2.1. Command and File Examples

All of the examples for JBoss ON commands, file locations, and other usage are given for Red Hat Enterprise Linux systems. Be certain to use the appropriate commands and files for your platform.

Example 1. Example Command

To start the JBoss ON server:

```
serverRoot/jon-server-3.1.2.0.GA1/bin/rhq-server.sh start
```

2.2. Text Formatting and Styles

Certain words are represented in different fonts, styles, and weights. Different character formatting is used to indicate the function or purpose of the phrase being highlighted.

Formatting Style	Purpose
Monospace font	Monospace is used for commands, package names and directory paths, and any text displayed in a p

Formatting Style	Purpose
<div style="border: 1px solid black; padding: 5px; width: fit-content;"> Monospace with a background </div>	This type of formatting is used for anything entered and returned in a command prompt.
<i>Italicized text</i>	Any text which is italicized is a variable, such as <i>instance_name</i> or <i>hostname</i> . Occasionally, this is used to emphasize a new term or other phrase.
Bolded text	Most phrases which are in bold are application names, such as Cygwin , or are fields or options in a user interface, such as a User Name Here: field or OK button.

Other formatting styles draw attention to important text.



Note

A note provides additional information that can help illustrate the behavior of the system or provide more detail for a specific issue. Tips provide pointers to helpful information or to easy ways to accomplish something.



Important

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



Warning

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

3. Document History

Revision 3.1.2-1.400	2013-10-31	Rüdiger Landmann
Rebuild with publican 4.0.0		
Revision 3.1.2-1	January 23, 2013	Ella Deon Lackey
Added note for support for Java 7.		
Clarified the agent RPM channels to use for a yum installation, Bugzilla 870329.		
Updating agent plug-in documentation, Bugzilla 864129 and Bugzilla 882745.		
Revision 3.1.1-6	January 8, 2013	Ella Deon Lackey

Removed incorrect reference to Java 7, Bugzilla 878369 and Bugzilla 878951.

Clarified setting up an Oracle database and setting the database name in the connection URL, Bugzilla 823963.

Corrected incorrect command for agent upgrade, Bugzilla 870369.

Revision 3.1.1-5	October 3, 2012	Ella Deon Lackey
-------------------------	------------------------	-------------------------

Adding in new chapter for installing the agent from RPM and general restructuring of the doc.

Bug fixes as part of JBoss Operations Network 3.1.1.

Revision 3.1-0	June 12, 2012	Ella Deon Lackey
-----------------------	----------------------	-------------------------

Initial release of JBoss Operations Network 3.1.

Chapter 1. JBoss Operations Network 3.1.2 Prerequisites

1.1. Supported Platforms, Databases, and Other Requirements

The list of supported platforms, databases, and other requirements such as Java, are listed at <https://access.redhat.com/knowledge/articles/112523>.

1.2. Hardware Requirements

Regardless of the server or database platform, there are certain minimum requirements that must be met to install the JBoss ON server and its associated database.

Table 1.1. Recommended Minimum Hardware

	Minimum
Memory	2 GB
Installation Directory Storage [a]	10 GB
Temporary Directory Storage	10 GB

[a] The server runs as a system user. Make sure that any system limits on user memory are set high enough to accommodate the JBoss ON server and all its data.

Chapter 2. Setting up Databases

2.1. Configuring PostgreSQL

Running JBoss Operations Network on PostgreSQL requires three things:

- ✦ Adequate PostgreSQL settings for memory, timeouts, connections, and related settings
- ✦ A database
- ✦ A user with adequate permissions

JBoss ON supports PostgreSQL 8.2.4 and later 8.2.x versions, 8.3, 8.4, and 9.0.

2.1.1. Installing PostgreSQL

You can download the Microsoft Windows binaries you need from:

<https://www.postgresql.org/download/windows/>

Use **YUM** to install PostgreSQL:

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

To install a specific version of PostgreSQL, go to: <https://yum.postgresql.org/rpmchart.php> and download the **postgresql**, **postgresql-server** and **postgresql-libs** RPM packages and install via **yum** from the download directory. For example:

```
sudo yum install
postgresql91-9.1.24-2PGDG.rhel6.x86_64.rpm
postgresql91-libs-9.1.24-2PGDG.rhel6.x86_64.rpm
postgresql91-server-9.1.24-2PGDG.rhel6.x86_64.rpm
```

2.1.2. Configuring PostgreSQL

For more detailed information about setting up client authentication for PostgreSQL users and databases, see the PostgreSQL documentation at <http://www.postgresql.org/docs/8.4/interactive/client-authentication.html>.



Note

Ensure that the Postgres authentication mechanism is properly configured for the configuration commands to work.

1. *Optional.* Change the password for the Unix user for PostgreSQL:

```
sudo passwd postgres
```

2. Initialize the database. The database must be initialized before starting the server.

```
service postgresql initdb
```

3. Start Postgres. For example, on Red Hat Enterprise Linux:

```
service postgresql start
```

On Windows:

```
net start pgsql-8.3
```

4. Set up a password for the **postgres** user on the database:

```
# su - postgres
$ psql
postgres=# ALTER USER postgres PASSWORD 'password';
ALTER ROLE
postgres=#
```

5. Create a PostgreSQL role named **rhqadmin**, where *'password'* should be replaced with a strong password.

```
postgres=# CREATE USER rhqadmin PASSWORD 'password';
CREATE ROLE
```



Important

Although the default postgresql credentials are user **rhqadmin** and password **rhqadmin**, these credentials should not be used as they present a security risk. Use these credentials are needed for [Section 3.3, “Configuring the Server with the Web Installer”](#) or [Section 3.4, “Silently Installing the JBoss ON Server”](#).

6. Create a PostgreSQL database named **rhq**, specifying the **rhqadmin** role as the owner.

```
postgres=# CREATE DATABASE rhq OWNER rhqadmin;
CREATE DATABASE
```

7. Give users on the computer access to the database. To allow all users, add the appropriate connection settings for each connection type (local, IPv4, and IPv6) to the **data/pg_hba.conf** configuration file, for both local and external connections:

```
# "local" is for Unix domain socket connections only
local all all md5
# IPv4 local connections:
host all all 127.0.0.1/32 md5
host all all 172.31.7.0/24 md5
# IPv6 local connections:
host all all ::1/128 md5
```

Using **all all** sets these settings for every user to every PostgreSQL database. This settings can be applied to only the JBoss ON database by using **rhq all** or even to specific users for JBoss ON, such as **rhq rhqadmin**.

Then, restart the database service.

```
service postgresql restart
```

8. Make the configuration changes in [Section 2.1.3, “Setting PostgreSQL Parameters”](#).

2.1.3. Setting PostgreSQL Parameters

There are several settings in the PostgreSQL server configuration that can be tuned to provide better performance for JBoss ON.

2.1.3.1. Editing the postgresql.conf File

PostgreSQL requires minor changes to the database configuration in the **postgresql.conf** file.

1. Make sure that an adequate amount of memory and system resources are assigned to accommodate the JBoss ON database.

```
## not necessary if the database is started with the -i flag
listen_addresses = '*'

## performance changes for JBoss ON
shared_buffers = 80MB # default is 32MB
work_mem = 2048 # default is 1MB
checkpoint_segments = 10 # default is 3
```

For PostgreSQL 8.2.4 and 8.3, also set the **max_fsm_pages** parameter. (This parameter should not be used on PostgreSQL 8.4 and later databases.)

```
max_fsm_pages = 100000 # default is 204800
```

2. *Optional.* Set the statement timeout period so a size that is adequate to handle data compression in large environments. By default, the default is zero (0) seconds, which means there is no statement timeout set; not having a timeout period is the preferred setting for smaller deployments.

```
statement_timeout = 0s # default is 0s
```



Note

If there is already a global statement timeout period for that database, but you need to use a larger setting for JBoss ON, set a user-level statement timeout value that only applies to the JBoss ON user.

```
ALTER USER rhqadmin SET statement_timeout=600000;
```

3. JBoss ON can use up to 55 database connections for the server. PostgreSQL also allows for connections reserved for administrators. These connections are counted in the pool of **max_connections** and therefore need to be added to the total number of **max_connections**. For example, if there are five connections reserved for the administrator, edit the **postgresql.conf** file as follows:

```
max_connections = 60      # default is 100
superuser_reserved_connections = 5 # default is 3
max_prepared_transactions = 60      # default is 5 (in v8.3) or 0 (in
v8.4)
```



Note

max_prepared_transactions is set to the same value as *max_connections*, as explained in the "max_prepared_transactions (integer)" in the [PostgreSQL documentation](#).

If JBoss ON is also monitoring this database instance, add one more connection per (logical) database that is set up in PostgreSQL. For further information about this plug-in, see the PostgreSQL server section of the *Resource Monitoring Reference*.

2.1.3.2. Setting Kernel Parameters

Consider adjusting the kernel parameters for your system. The PostgreSQL documentation on [Managing Kernel Resources](#) has more information.

2.1.3.3. Editing `pg_hba.conf`

Update the `pg_hba.conf` file to allow the newly-created role to connect from the machine the JBoss ON server is installed on, such as localhost. Adding client connections is covered in the PostgreSQL documentation in the [Client Authentication](#) section.

After editing the `pg_hba.conf` file, restart PostgreSQL for the changes to take effect. If no errors are displayed, the database is now ready to support a JBoss ON installation.

For more information on tuning Postgres, see the PostgreSQL documentation about [Tuning your PostgreSQL Server](#).

2.1.3.4. Fixes for "Relation RHQ_Principal does not exist" Error

Sometimes the database connection is marked as valid but the install still fails with the *Relation RHQ_Principal does not exist* error. This occurs when a new database is created by running `initdb` in a *non-C* locale through PostgreSQL instances.

To fix this error:

1. Using a database explorer, create an empty table called ***RHQ_PRINCIPAL*** in the database used for JBoss ON.
2. Click **Install server**.

The installer displays a warning about an existing schema. Overwrite the existing schema as it only consists of one empty table.

Another option is to specify the encoding of the created database as ***SQL-ASCII*** at creation time. For example:

```
initdb -D /my/test/data -E SQL_ASCII --locale en_US.UTF-8
```

2.2 Setting up Oracle

2.2. Setting up Oracle

Only two things are required to run JBoss ON on Oracle:

- » A database
- » A user with adequate permissions

Basic configuration follows the process of setting up the database and users. There is also an advanced configuration process that gives more control over the database settings, such as increased memory limits, that can improve performance for large JBoss ON deployments.

2.2.1. Prepping Oracle Settings

There are several settings in the Oracle configuration that can be tuned to provide better performance for JBoss ON.

2.2.1.1. Setting SGA and PGA Sizes

Oracle settings for SGA and PGA sizes are very important for JBoss ON performance. If these values are too small, the database will be very slow. There are two specific settings to adjust:

- » `sga_target`
- » `pga_aggregate_target`

Talk to the database administrator to verify the sizing requirements for Oracle's SGA and PGA settings.

2.2.1.2. Tuning Open Cursors

Run the following SQL command to check if the `max_open_cur` setting has a value lower than 300:

```
select max(a.value) as highest_open_cur, p.value as max_open_cur
from v$sesstat a, v$statname b, v$parameter p
where a.statistic# = b.statistic#
and b.name = 'opened cursors current'
and p.name= 'open_cursors'
group by p.value;
```

If the value *is* lower than 300, then open more cursors:

```
ALTER SESSION SET OPEN_CURSORS = 300 SCOPE=BOTH;
```

2.2.1.3. Setting the Number of Processes and Sessions

The `v$resource_limit` limit sets the maximum number of Oracle processes and sessions which JBoss ON is allowed to have. The equation for this calculation has this general flow:

```
calculate the number of processes => add additional processes for Enterprise
Manager => calculate the total number of sessions (final value)
```

There are two ways to calculate the number of processes (one using the number of agents and the other the number of servers). Use whichever method results in a higher number.

Table 2.1. Calculating Oracle Processes

Calculation Type	Equation	Example
Agents	$1.5 * \text{number_of_agents}$	$1.5 * 100 \text{ agents} = 150$
Servers	$60 * \text{number_of_servers}$	$60 * 2 \text{ servers} = 120$
with Oracle Enterprise Manager	$\text{highest_number_of_processes} + 40$	$1.5 * 100 \text{ agents} + 40 = 190$

As noted in [Table 2.1, "Calculating Oracle Processes"](#), the calculation is slightly different for systems using Oracle Enterprise Manager. In that situation, first calculate the processes for agents and servers. Then, take whichever value is highest and add another 40, and that yields the number of processes to set.

After calculating the total number of processes, then take that number and multiply it by 1.1 to determine the total number of sessions (and the final value for `v$resource_limit`).

Example 2.1. Calculating Oracle Processes and Sessions for JBoss ON

Example Corp. is planning to deploy 175 agents and 3 servers. They will be using Oracle Enterprise Manager to manage their Oracle instance.

The first step is to calculate the number of processes based on agents and based on servers:

```
1.5 * 175 agents = 262.5 processes
60 * 3 servers = 180 process
```

So the method to use for processes is the agent's method, since that value is higher.

They add another 40 to the number of processes to accommodate the Oracle Enterprise Manager.

```
262.5 + 40 = 302.5
```

The total number of process is 302.5. From there, they calculate the number of sessions:

```
302.5 * 1.1 = 332.75
```

The final value for their Oracle `v$resource_limit` limit database setting is 333.

2.2.2. Configuring Oracle

A specific Oracle database and user need to be configured for JBoss ON to access to store its data.

1. Create a dedicated Oracle instance to be used for JBoss ON. This process is described in the Oracle documentation.
2. Log into Oracle as the system user.

```
[jsmith@server ~]$ sqlplus
SQL> CONNECT sys/your_sys_password AS sysdba;
```

3. Create a database for JBoss ON. In this example, the database is named `rhq`. This process is described in more detail in the Oracle documentation.


```
SQL> CREATE DATABASE rhq;
SQL> @?/rdbms/admin/catalog.sql
SQL> @?/rdbms/admin/catproc.sql
```

4. Create a user that JBoss ON will use to access Oracle. Create the user named **rhqadmin** with the password **rhqadmin**. For example:

```
SQL> CREATE USER rhqadmin IDENTIFIED BY rhqadmin;
```

5. Grant the required permissions to the Oracle user. At a minimum, this user must have the **connect** and **resource** roles. For example:

```
SQL> GRANT connect, resource TO rhqadmin;
```

6. Set additional permissions for the JBoss ON Oracle user that define parameters to handle database commits.

JBoss ON uses internally two phase commit for some of database actions. To recover from two phase commit failures, the Oracle user has to have appropriate permissions, otherwise the database will return **XAException.XAER_RMERR** errors.

Set these four privileges for the user:

```
GRANT SELECT ON sys.dba_pending_transactions TO user;
GRANT SELECT ON sys.pending_trans$ TO user;
GRANT SELECT ON sys.dba_2pc_pending TO user;
GRANT EXECUTE ON sys.dbms_xa TO user;
```

The **GRANT EXECUTE** line assumes that the Oracle server is version 11g R1. For an unpatched version of Oracle older than 11g R1, then use this line instead:

```
GRANT EXECUTE ON sys.dbms_system TO user;
```

7. Make sure that the **db_block_size** value is at least 8 KB.

```
SQL> show parameter db_block_size;
NAME                                TYPE                                VALUE
-----                                -                                -
db_block_size                        integer                             8192
```

2.2.3. Configuring Oracle (Advanced)

There are optional configurations that can help Oracle perform effectively with large JBoss ON environments, such as deployments with hundreds of JBoss ON agents. This configuration is not necessary for smaller environments.

**Note**

For advanced configuration, install Oracle using the graphical wizard rather than SQL command-line tools.

1. Create a new database.
 - a. Open the **Oracle Database Configuration Assistant**.
 - b. Select **New Database**.
 - c. Set the ***Includes datafiles*** parameter to **No**.
 - d. Decline to install the example schemas to save space.
 - e. Select **Typical Memory** configuration, and then set the database sizing type to **OLTP**.
 - f. Allocate the highest percentage of system resources that the system can afford. This should be between 70% and 90%, with the highest value preferred.

**Warning**

Locally manage all tablespaces.

2. Create the JBoss ON user.

```
CREATE USER rhqadmin IDENTIFIED BY rhqadmin;
```

3. Grant the required permissions to the new user.

```
GRANT CONNECT, RESOURCE TO rhqadmin;
```

4. Set additional permissions for the JBoss ON Oracle user that define parameters to handle database commits.

JBoss ON uses internally two phase commit for some of database actions. To recover from two phase commit failures, the Oracle user has to have appropriate permissions, otherwise the database will return **XAException.XAER_RMERR** errors.

Set these four privileges for the user:

```
GRANT SELECT ON sys.dba_pending_transactions TO rhqadmin;
GRANT SELECT ON sys.pending_trans$ TO rhqadmin;
GRANT SELECT ON sys.dba_2pc_pending TO rhqadmin;
GRANT EXECUTE ON sys.dbms_system TO rhqadmin;
```

Chapter 3. Installing and Upgrading the JBoss ON Server on Linux

The core of JBoss Operations Network is the server, which communicates with agents, maintains the inventory, manages resource settings, interacts with content providers, and provides a central management UI. JBoss ON has other components which are required in order for JBoss ON to carry out its functions — agents which are installed on platforms, a CLI which allows administrators to script configuration, and plug-ins which integrate JBoss ON with other JBoss products. Each component has to be installed and configured independently, to match the needs of the specific network.

3.1. Preparing for Installation

3.1.1. Setting up the JDK for the JBoss ON Server

The JBoss ON server requires Java 6 or Java 7 JDK.

1. Download and install the appropriate version of Java, if necessary.
2. Set the **JAVA_HOME** environment variable to the installation directory.
 - a. Open the **.bashrc** for the system user that will run JBoss ON. For example:

```
vim /home/jon/.bashrc
```

- b. Add a line to set the **JAVA_HOME** environment variable to the specific JDK directory. For example:

```
export JAVA_HOME=/usr/lib/jvm/java-1.6.0-openjdk-1.6.0.0/
```

3. Set the system to use the correct version of the JDK using the system **alternatives** command. The selected version has the ***+** symbols by it.

```
/usr/sbin/alternatives --config javac

There are 2 programs which provide 'javac'.

  Selection    Command
-----
     1         /usr/lib/jvm/java-1.6.0-bea/bin/javac
*+  2         /usr/lib/jvm/java-1.6.0-openjdk/bin/javac

Enter to keep the current selection[+], or type selection number:
```

3.1.2. Preparing the Host Machine

For JBoss ON servers and agents to be able to communicate, they have to be able to connect to each other's host machines. Make sure that the machines are configured so that the server and agent machines are able to locate and connect with each other. There are three areas that commonly need to be configured:

- ✦ **Install the *urw-fonts* package.** Java requires certain system fonts packages in order to label charts and graphs properly. If the *urw-fonts* package is not installed, then the monitoring graphics cannot be rendered.

- ✦ **Synchronize machine clocks.** All JBoss ON servers and agents must have synchronized clocks. Clock variations cause issues in availability reporting, metric measurements, graphing, and even identifying and importing resources into inventory. The Network Time Protocol project, <http://www.ntp.org/>, has information on installing and configuring NTP to ensure your clocks are synchronized.
- ✦ **Configure DNS.** Both forward and reverse DNS mapping entries must be present for all hosts involved in monitoring. This includes all JBoss ON servers and all machines running agents.
- ✦ **Configure the firewall to allow communication over the server and agent ports.** Ensure the necessary ports have been opened to prevent the firewall from blocking the JBoss ON server and agents from communicating. The JBoss ON server typically uses port 7080, and the JBoss ON agents typically use port 16163.

3.2. Installing the Server JAR File

1. Stop any currently running JBoss ON instances.

```
serverRoot/jon-server-3.1.2.0.GA1/bin/rhq-server.sh stop
```



Warning

If the new JBoss ON server will use a database that existing JBoss ON instances are also using, then all of the existing JBoss ON instances have to be stopped. Otherwise, the installer will hang when it tries to contact the database and the database is unavailable because it is in use by another JBoss ON server.

2. Download the JBoss ON binaries from the [Customer Support Portal](#).
 - a. In the Customer Support Portal, click **Software**, and then select **JBoss Operations Network** in the product drop-down box.
 - b. Download the **JBoss Operations Network 3.1.2 Base Distribution** package by clicking the **Download** icon.
 - c. There are additional plug-in packs available for EAP, EDS, EWS, and SOA-P. If any of those plug-ins will be used with the JBoss ON server, then download them as well.
3. Unzip the server distribution to the directory where will be executed from.

```
cd /opt  
  
unzip jon-server-3.1.2.0.GA1.zip
```

This creates a version-specific installation directory, `/opt/jon-server-3.1.2.0.GA1`. A directory with this name should not exist prior to the unzip operation.

4. Run the JBoss ON server:

```
serverRoot/jon-server-3.1.2.0.GA1/bin/rhq-server.sh start
```

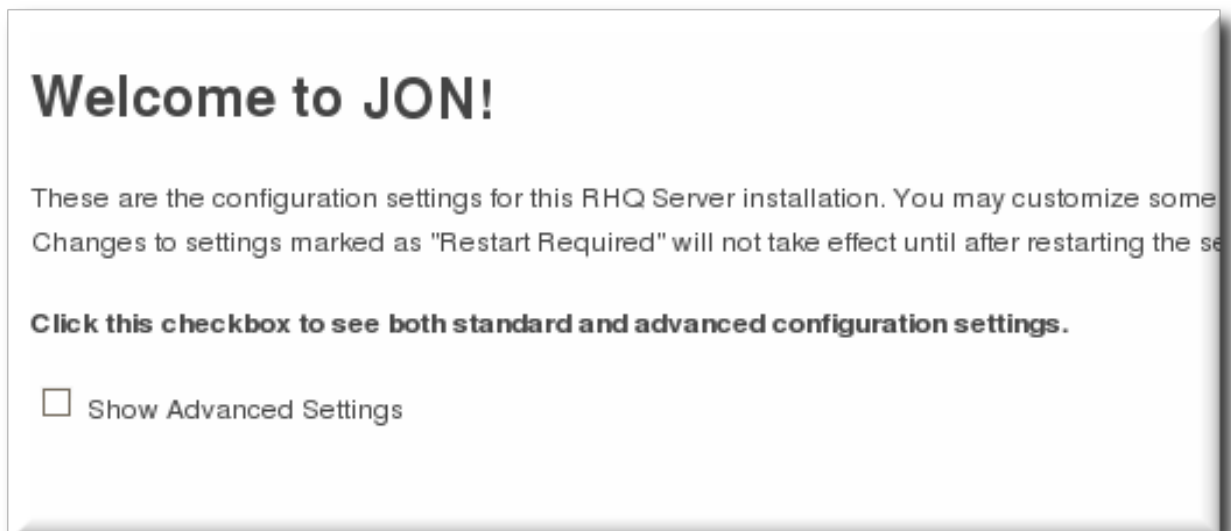
5. Set up the JBoss ON server using the web installer ([Section 3.3, "Configuring the Server with the Web Installer"](#)) or by editing the configuration file ([Section 3.4, "Silently Installing the JBoss ON Server"](#)).

3.3. Configuring the Server with the Web Installer

1. Open the server UI at `http://localhost:7080/`.



2. Clicking the **Click here to continue the installation** link brings you to the main installer page.
3. By default, the installer displays only the typical settings required for installation. For more custom environments, click the **Show Advanced Settings** check box to display the advanced settings in the next two sections.



Important

Do not unselect the advanced settings checkbox after setting advanced values, or the advanced settings will be reset to the default values.

4. Set the database connection properties. This should include the database type, the database hostname and port, and the name of the database created for JBoss ON (in this example, `rhq`).

The installation options are slightly different depending on the database configuration.

Database Settings define the database configured for this installation. All are required. Use the "Test Connection" button to validate the settings.

Database Type	PostgreSQL
Database Connection URL	jdbc:postgresql://127.0.0.1:5432/rhq
Database JDBC Driver Class	org.postgresql.Driver
Database XA DataSource Class	org.postgresql.xa.PGXADDataSource
Database User Name	rhqadmin
Database Password	

Confirm database settings

Test Connection



Important

With Oracle, selecting the overwrite tables option when there is nothing to overwrite causes an error message with **ErrorCode=[2289]**. This can be ignored.

- Set the basic connection information for the new JBoss ON server, such as its HTTP and HTTPS ports.

Installation Settings define the required server endpoint for this installation.

	Requires Restart?	
Server Name	localhost.localdomain	No
Server Public Address	localhost.localdomain	No
HTTP Port	7080	Yes
Secure HTTPS Port	7443	Yes

- Set preliminary notification information for the JBoss ON server for alerts to use for SNMP and email notifications.

Server Settings configure the server for this installation. All server settings are required.

	Requires Restart?	
Server Bind Address	0.0.0.0	Yes
Embedded Agent Enabled	<input type="radio"/> Yes <input checked="" type="radio"/> No	No
Email SMTP Hostname	localhost	No
Email From Address	rhqadmin@localhost	No

Install Server!

- Click the **INSTALL** button to begin configuring the server. This can take several minutes, and a loading screen will be up until the server is configured and a redirect is available.

loading screen will be up until the server is configured and a redirect is available.



8. When the server is configured, click the link to go to the UI.
9. Log into the GUI using the user account with the default superuser and password, **rhqadmin/rhqadmin**. The default username and password are predefined for the superuser; the password can be reset later.



Note

Change the password for the superuser account, **rhqadmin**, when you first log in. Passwords are changed in the **Administration > Users** area.

10. *Optional.* Install any additional agent plug-ins, as in [Chapter 5, Installing JBoss Agent Plug-in Packs](#).
11. *Optional.* Install the JBoss ON CLI, as described in [Chapter 6, Installing the JBoss ON CLI](#).
12. Begin configuring JBoss ON agent and resources, as in [Section 3.5, “Additional Post-Setup Checklist”](#).

3.4. Silently Installing the JBoss ON Server

The initial setup of the server is the same as in [Section 3.3, “Configuring the Server with the Web Installer”](#). Instead of using the visual, web-based installer, however, a silent installation loads a pre-configured properties file and starts the server with all of its configuration in place.

1. Stop any currently running JBoss ON instances.

```
[root@server ~]# serverRoot/jon-server-3.1.2.0.GA1/bin/rhq-server.sh
stop
```



Warning

If the new JBoss ON server will use a database that existing JBoss ON instances are also using, then all of the existing JBoss ON instances have to be stopped. Otherwise, the installer will hang when it tries to contact the database and the database is unavailable because it is in use by another JBoss ON server.

2. Download the JBoss ON binaries from the [Customer Support Portal](#).
 - a. In the Customer Support Portal, click **Software**, and then click the **Product** drop-down box arrow to open the **JBoss Operations Network** software download.
 - b. Download the **JBoss Operations Network 3.1.2 Base Distribution** package by clicking the **Download** icon.
 - c. There are additional plug-in packs available for EDS, EAP, EWS, and SOA-P. If any of those plug-ins will be used with the JBoss ON server, then download them as well.
3. Unzip the server distribution to the directory from where it will be executed.

```
[root@server ~]# cd /opt  
[root@server ~]# unzip jon-server-3.1.2.0.GA1.zip
```

4. Install the agent plug-ins.

```
[root@server ~]# unzip jon-plugin-pack-agent_plugin_name-  
3.1.2.0.GA1.zip  
[root@server ~]# cd /opt/jon/jon-server-3.1.2.0.GA1/plugins
```

5. Open the JBoss ON server configuration properties file:

```
[root@server ~]# vim serverRoot/jon-server-3.1.2.0.GA1/bin/rhq-  
server.properties
```

6. Set the **rhq.autoinstall.enabled** parameter to true to instruct the server to load the configuration automatically from file.

```
rhq.autoinstall.enabled=true
```

7. Set the parameter to tell the server whether preserve the any existing data in the database. For a new installation, this can be set to **auto** so the installer will supply the database schema. If a JBoss ON server has been installed before so that there is existing data in the database, then this can be set to **overwrite**, to apply the new schema.

```
rhq.autoinstall.database=auto
```



Important

The autoinstaller options are only evaluated once, when the JBoss ON server is first installed. After that initial configuration, the autoinstaller is disabled. These properties are ignored once the server is set up and cannot be used to initiate a re-install of an existing instance.

8. Optionally, set the IP address or hostname for the server. If this is not set, then the server will detect it when it starts.

```
rhq.autoinstall.public-endpoint-address=server1.example.com
```

9. Set the port numbers for the instance.


```
rhq.server.startup.web.http.port=7080
rhq.server.startup.web.https.port=7443
```

- Set the username and password to connect to the backend database. The password is stored in the properties file in an encoded form. Use the **generate-db-password.sh** script to hash the password.

```
serverRoot/jon-server-3.1.2.0.GA1/bin/generate-db-password.sh
mypassword
Encoded password: 1d31b70b3650168f79edee9e04977e34
```

Then, set the hashed password as the **rhq.server.database.password** value.

```
rhq.server.database.user-name=rhqadmin
rhq.server.database.password=1d31b70b3650168f79edee9e04977e34
```

- Scan the rest of the database configuration to make sure that it corresponds to your specific database type and instance. The parameters, and some possible values for PostgreSQL and Oracle databases, are described in [Table 3.1, “rhq-server.properties Parameters for Database Configuration”](#).

```
# Database
rhq.server.database.connection-
url=jdbc:postgresql://127.0.0.1:5432/rhq
rhq.server.database.driver-class=org.postgresql.Driver
rhq.server.database.xa-datasource-
class=org.postgresql.xa.PGXADatasource
rhq.server.database.type-mapping=PostgreSQL
rhq.server.database.server-name=127.0.0.1
rhq.server.database.port=5432
rhq.server.database.db-name=rhq
hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect

# Quartz
rhq.server.quartz.driverDelegateClass=org.quartz.impl.jdbcjobstore.Pos
tgreSQLDelegate
rhq.server.quartz.selectWithLockSQL=SELECT * FROM {0}LOCKS ROWLOCK
WHERE LOCK_NAME = ? FORUPDATE
rhq.server.quartz.lockHandlerClass=org.quartz.impl.jdbcjobstore.StdRow
LockSemaphore
```

- Start the JBoss ON server:

```
[root@server ~]# serverRoot/jon-server-3.1.2.0.GA1/bin/rhq-server.sh
start
```

When the server starts, it loads the edited **rhq-server.properties** file and is fully configured.

Table 3.1. rhq-server.properties Parameters for Database Configuration

Parameter	Description
-----------	-------------

Parameter	Description
rhq.server.database.type-mapping	Gives the type or vendor of the database that is used by the JBoss ON server. This is either PostgreSQL or Oracle.
rhq.server.database.connection-url	The JDBC URL that the JBoss ON server uses when connecting to the database. This has the format (roughly) of <i>jdbc:db-type:hostname:port[:\]/db-name</i> . An example is <i>jdbc:postgresql://localhost:5432/rhq</i> or <i>jdbc:oracle:oci:@localhost:1521:orcl</i> .
rhq.server.database.driver-class	The fully qualified class name of the JDBC driver that the JBoss ON server uses to communicate with the database. An example is <i>oracle.jdbc.driver.OracleDriver</i> .
rhq.server.database.xa-datasource-class	The fully qualified class name of the JDBC driver that the JBoss ON server uses to communicate with the database. Examples of this are <i>org.postgresql.xa.PGXADatasource</i> or <i>oracle.jdbc.xa.client.OracleXADatasource</i> .
rhq.server.database.user-name	The name of the user that the JBoss ON server uses when logging into the database
rhq.server.database.password	The password of the database user that is used by the JBoss ON server when logging into the database. This password is stored in a hash.
rhq.server.database.server-name	The server name where the database is found. This must match the server in the connection URL. This is currently only used when connecting to PostgreSQL.
rhq.server.database.port	The port on which the database is listening. This must match the port in the connection URL. This is currently only used when connecting to PostgreSQL.
rhq.server.database.db-name	The name of the database. This must match the name found in the connection URL. This is currently only used when connecting to PostgreSQL.
rhq.server.quartz.driverDelegateClass	The Quartz driver used for connections between the server and the database. The value of this is set by the installer and depends on the type of database used to store the JBoss ON information. For PostgreSQL, this is org.quartz.impl.jdbcjobstore.PostgreSQLDelegate , and for Oracle, this is org.quartz.impl.jdbcjobstore.oracle.OracleDelegate .

3.5. Additional Post-Setup Checklist

Once the JBoss ON server is configured, there are a number of different areas that administrators can configure to set up and manage resource in JBoss ON and to configure JBoss ON itself.

Most of the initial setup — building out the JBoss ON inventory with managed resources, setting up roles and access control, and creating resource groups — is covered in [Initial Setup: the Resource Inventory, Groups, and Users](#). JBoss ON has a number of other features like drift monitoring, alerting, application and content deployment, server high availability, and configuration management. The way that these are configured

depends on the demands of your infrastructure. [Table 3.2, “Configuration and Features to Set Up”](#) has links to procedures for features which are commonly set up soon after JBoss ON is set up; browse the JBoss ON GUI and documentation for a more comprehensive view of JBoss ON's capabilities.

Table 3.2. Configuration and Features to Set Up

Feature	Description	Doc Link
Creating resource groups	For simplicity and effectiveness with managing resources, JBoss ON has resource groups. There are a number of different types of groups, including compatible groups (groups of the same type of resource) and dynamically-created groups. This makes it possible to apply configuration changes, alert definitions, drift definitions, and other settings all at once and to create useful groups for better monitoring and inventory tracking.	"Managing Groups"
Configuring roles and access control	JBoss ON defines access to resources based on <i>roles</i> . Both users and resources are assigned to roles, and then access control rights are assigned to the roles.	"About Security in JBoss ON: Roles and Access Control"
LDAP user authentication	By default, users are created in JBoss ON and then stored in the JBoss ON database. However, JBoss ON can be configured to check an LDAP server first for user accounts, so existing LDAP users can be authenticated in JBoss ON — without having to create users in JBoss ON first.	"How JBoss ON Uses LDAP for Authentication"
LDAP group authorization	LDAP groups can be associated with JBoss ON roles. This means that the members in the LDAP group are automatically granted the rights and can manage the resources defined in the JBoss ON role.	"How JBoss ON Roles Work with LDAP User Groups"
Creating alerts	An alert is a way of informing administrators that some event or condition has occurred on a resource. JBoss ON provides a number of different ways to set and respond to alerts, both by sending notifications and by taking a specified action.	"Brief Introduction to Alerts and Notifications"

Feature	Description	Doc Link
Configuring drift monitoring	When configuration settings change, the configuration <i>drifts</i> from its designated state. Drift monitoring provides a mechanism for administrators to define and manage that designated configuration state, to apply it to multiple resources, and to be informed if any resource drifts from that state.	"Understanding Drift"
Setting up bundles for provisioning	Provisioning is a way of deploying content through JBoss ON. Provisioning takes a <i>bundle</i> , an archive file, and deploys it on a platform or a JBoss resource. This bundle can be a configuration file, a set of EARs or WARs, or even a full application server. JBoss ON supports multiple versions of the same bundle and can deploy these bundles to different resources, all managed through a single point in JBoss ON.	"Creating Bundles"
Setting affinity and high availability	When multiple JBoss ON servers are added, they naturally establish a high availability and failover topology. Agents can be managed by any server in the cloud, but it is possible to set a preference, or <i>affinity</i> , for agents to be managed by selected servers. This can improve performance or be used to reflect the infrastructure topology.	"Configuring High Availability"

3.6. Configuring the Server as a Service

The `rhq-server.sh` script can be managed by the `init` process so that the server starts automatically when the system boots. This also allows the server process to be managed by services like `service` and `chkconfig`.

1. Copy the `rhq-server.sh` script into the `/etc/init.d/` directory.

```
cp serverRoot/bin/rhq-server.sh /etc/init.d/
```

2. Edit the `/etc/init.d/rhq-server.sh` script to set the `RHQ_SERVER_HOME` variable to the JBoss ON server install directory and the `RHQ_SERVER_JAVA_HOME` variable to the appropriate directory for the JVM. For example:

```
RHQ_SERVER_HOME=serverRoot/jon-server-3.1.2.0.GA1/
```

```
RHQ_SERVER_JAVA_HOME=/usr/
```

3. Edit the `/etc/init.d/rhq-server.sh` script, and add the following lines to the top of the file, directly under `#!/bin/sh`.

```
#!/bin/sh
#chkconfig: 2345 95 20
#description: JBoss Operations Network Server
#processname: run.sh
```

The last two numbers in the `#chkconfig: 2345 95 20` line specify the start and stop priority, respectively, for the JBoss ON server.

4. Add the service to the `chkconfig` service management command, and verify that it was added properly.

```
chkconfig --add rhq-server.sh
chkconfig rhq-server.sh --list
```

5. Set the `rhq-server.sh` service to run at run level 5.

```
chkconfig --level 5 rhq-server.sh on
```

3.7. Re-Installing the Server

Whether it is configured through the web UI or silently, the JBoss ON server uses an autoinstaller to configure itself initially. Once that initial configuration is complete, the autoinstaller is disabled (so even setting the `rhq.autoinstaller.*` properties in `rhq-server.properties` does not re-initiate the server configuration).

To re-install the server, remove the entire home directory for the server, and then unzip the original JBoss ON server archive and configure the server as if it were all new, as in [Section 3.3, “Configuring the Server with the Web Installer”](#) or [Section 3.4, “Silently Installing the JBoss ON Server”](#).

3.8. Upgrading JBoss ON

An upgrade procedure for JBoss Operations Network essentially overlays the new JBoss ON packages and libraries over the existing configuration and databases. The upgrade procedure, then, is very similar to the installation process. The new packages need to be installed, and then the server is set up in the same setup wizard. The difference is that the server reuses its existing databases and data so that the configuration from the previous installation is preserved.

Upgrade to JBoss Operations Network 3.1.2 is only supported from JBoss ON 2.x versions and later.



Note

The JBoss ON servers must be upgraded before the JBoss ON agents can be upgraded.



Warning

There will be a minimal loss monitoring data because of the downtime required when the server and agents are being upgraded. Additionally, any monitoring data for the JBoss ON server will be lost, if the server is included in the inventory.

3.8.1. Upgrading the JBoss ON Server

Not every step in this upgrade procedure applies to every JBoss Operations Network installation. Just run through the steps in order, and perform the ones necessary for your deployment.



Warning

Upgrading the JBoss ON server essentially creates a new server instance that replaces the old instance. If the JBoss ON server was added to the inventory, then the old JBoss ON server resource must be deleted from the inventory because it will not be a usable resource after upgrade. Once the upgrade process is complete, then the JBoss ON server must be added to the inventory again and all of the previous configuration for that resource (like alerts, scheduled operations, and group membership) must be redone.



Note

See [Chapter 7, *Troubleshooting Installation and Upgrade*](#) if there are any problems during the upgrade process.

1. First, do some prep work on the JBoss ON configuration. It is easier to clean up the configuration before migration than it is after.
 - a. Remove any unused or out of service platforms from the inventory.
 - b. Remove any alert definitions which use conditions for obsolete metrics.

For migrating to JBoss ON 3.1.2, there are four alert conditions — all for PostgreSQL databases — which should be removed:

- » User Time
 - » Kernel Time
 - » Physical Memory
 - » Virtual Memory
2. Prepare the JBoss ON agents for upgrade. Agents will auto-upgrade, meaning that when they detect that the server has a new version, the agent will request an update. Follow the instructions at [Section 8.7, “About Agent Automatic Updates”](#) to prepare the agent, and then just leave it running. The agent should be running in the background to upgrade properly, as in [Section 8.4, “Running the JBoss ON Agent as a Service”](#).

In some rare cases, the agent will be upgraded manually instead of upgrading itself. In that case, stop the agent before upgrading the server, and follow the instructions at [Section 8.8, “Manually Upgrading the JBoss ON Agent”](#).

3. Stop the JBoss ON server which is being upgraded *as well* as any currently running JBoss ON instances. For example:

```
serverRoot/jon-server-3.1.2.0.GA1/bin/rhq-server.sh stop
```



Warning

If the upgraded JBoss ON server will use a database that existing JBoss ON instances are also using, then all of the existing JBoss ON instances have to be stopped. Otherwise, the installer will hang when it tries to contact the database and the database is unavailable because it is in use by another JBoss ON server.

4. Open the server root directory. For example:

```
cd /opt/jon
```

5. Unzip the server packages.

```
unzip jon-server-3.1.2.0.GA1.zip
```



Important

Do not copy the new server installation on top of a previous server installation.

The directory structure within the server package gives the new server installation directory a version-specific name, such as `/opt/jon/jon-server-3.1.2.0.GA1`.

6. Copy over any changes in your original `rhq-server.properties` file to the new file in `serverRoot/jon-server-3.1.2.0.GA1/bin`. Changes to this file include things like setting up SSL and enabling SMTP for email notifications.



Note

In JBoss ON 2.3.1 and older versions, the password to access the database is stored in plaintext. In JBoss ON 3.1.2, this password is hashed for security.



Note

If you don't want to edit the `rhq-server.properties` file manually, you can change the server settings to the proper configuration in the **Advanced Settings** form during the server setup.

7. Additional plug-in packs for specific needs (such as supporting management tasks for EWS, EAP, and SOA-P) are available to be installed separate from the core JBoss ON agent packages. Each plug-in pack as at least one (and sometimes more than one) agent plug-in. Each zip file for the plug-ins has a README.txt file with specific setup instructions.

The plug-in files can be unzipped anywhere. For example:

```
cd /opt/jon/jon-server-3.1.2.0.GA1  
unzip jon-plugin-pack-agent_plugin_name-3.1.2.0.GA1.zip
```



Note

If there are multiple JBoss ON servers in a high availability setup, the agent plug-in pack only has to be installed once. The other servers will pick up the plug-ins as part of the high availability polls.

8. Start the JBoss ON server. For example:

```
serverRoot/jon-server-3.1.2.0.GA1/bin/rhq-server.sh start
```

9. Back up your server database before going through the setup wizard. In case there is a problem with the upgrade process, the backup allows you to restore to its previous state.

10. Open the web UI.

```
http://hostname:7080
```

As with a new installation, the installer opens after you log in.

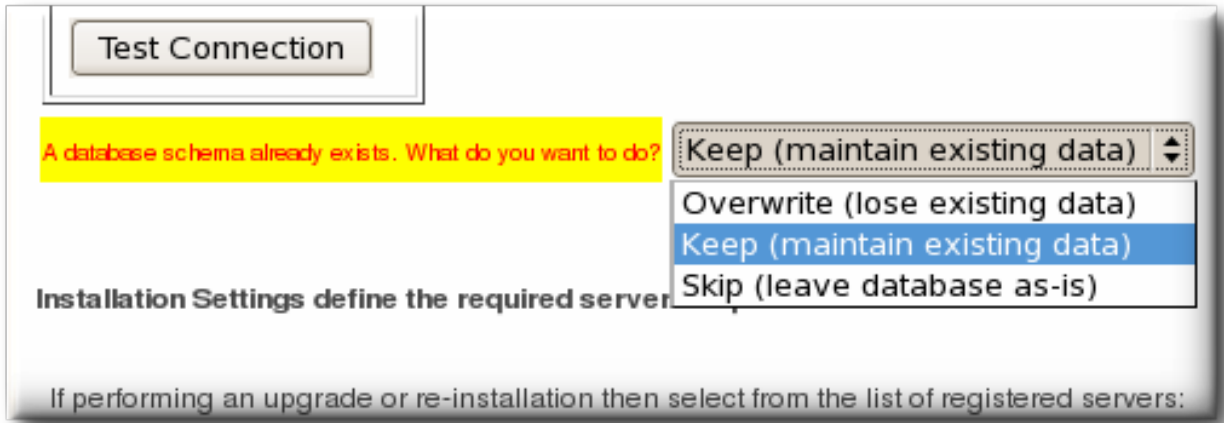
11. The setup process is the same as the initial setup procedure in [Chapter 3, Installing and Upgrading the JBoss ON Server on Linux](#).



Warning

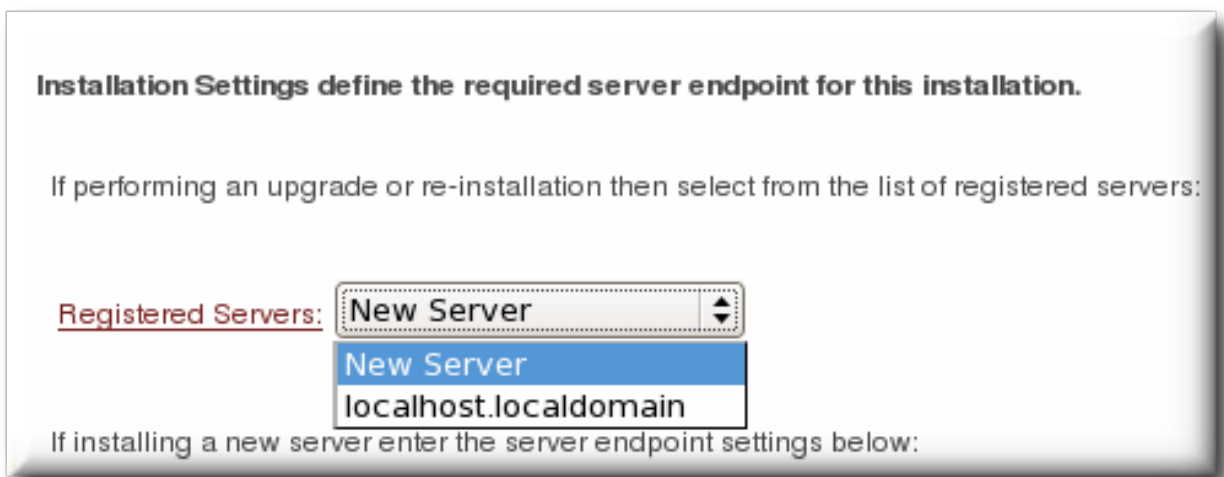
Do not change any of the settings for the server, especially identifying information such as the **Server Name** field. This can cause errors during the upgrade process.

When the database connection information is entered, the JBoss ON installer detects the existing JBoss ON database. This introduces a new field to the installer, prompting you for what to do with the existing database.



Choose the default, **Keep (maintain existing data)**. Do *not* choose **Overwrite (lose existing data)**, or the installer will delete all of your JBoss ON data, including your inventory, monitoring history, alerts, and metrics.

12. The **Registered Servers** lists every server in the server cloud. For upgrades and re-installs, this gives you the option to keep the existing server configuration (such as ports and notification settings) or to set new values. To preserve the settings, select the server from the registered servers list; otherwise, select **New Server**.



13. Restart the browser after upgrade completes or force the browser to refresh, using **Ctrl+F5**.



Note

The browser caches an old session ID for the previous JBoss ON server installation. Attempting to open the GUI without refreshing the cache causes the login to fail and throws an error in the JBoss ON server logs:

```
Exception:
2012-05-23 16:37:08,046 ERROR
[org.apache.catalina.connector.CoyoteAdapter] An exception or
error occurred in the container during the request processing
java.lang.NullPointerException
    at
org.apache.catalina.connector.CoyoteAdapter.parseSessionCookiesId
(CoyoteAdapter.java:507)
    at
org.apache.catalina.connector.CoyoteAdapter.postParseRequest(Coyo
teAdapter.java:449)
    at
org.apache.catalina.connector.CoyoteAdapter.service(CoyoteAdapter
.java:239)
... 8< ...
```

14. Start any JBoss ON agents that were stopped for the upgrade process.
15. If the older JBoss ON server was added to the JBoss ON inventory, then remove it. The old JBoss ON server must be removed from the inventory because it is no longer a usable resource.
16. *Optional.* Add the new JBoss ON server as a resource in the inventory.

3.8.2. Migrating SNMP Settings

The SNMP settings are not migrated with the other server settings. These settings have to be restored manually after migration, and there are two ways to do this:

- ✦ Run a SQL command to copy the settings from the database table for the old server instance into the table for the new instance.
- ✦ Set the SNMP settings in the JBoss ON GUI.

3.8.2.1. Running SQL Commands to Migrate the SNMP Settings

1. Open the administrative page, with the location **admin/test/sql.jsp**. For example:

```
http://server.example.com:7080/admin/test/sql.jsp
```

2. Run the command to migrate the SNMP settings:

```
select property_key,property_value from RHQ_SYSTEM_CONFIG where
property_key like 'SNMP%';
```

3. Click the **Execute SQL** button.

4. Reconfigure all of the SNMP alert notification senders.

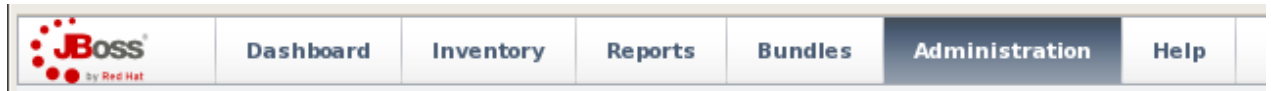
3.8.2.2. Configuring SNMP Settings in the GUI



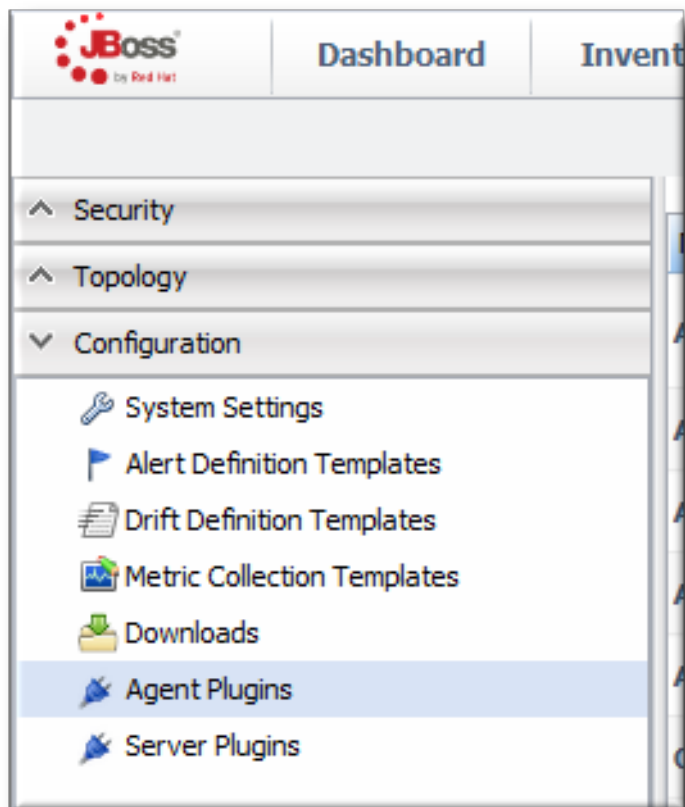
Note

Take a screenshot of the SNMP settings *before* beginning the upgrade procedure, so that the image is available as a reference to re-do the SNMP settings.

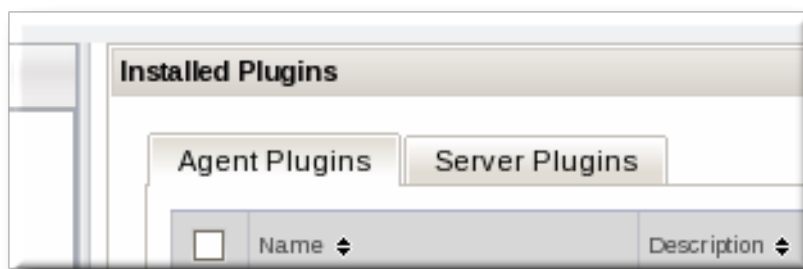
1. In the top menu, click the **Administration** tab.



2. In the **Configuration** box on the left navigation bar, click the **Plugins** link.

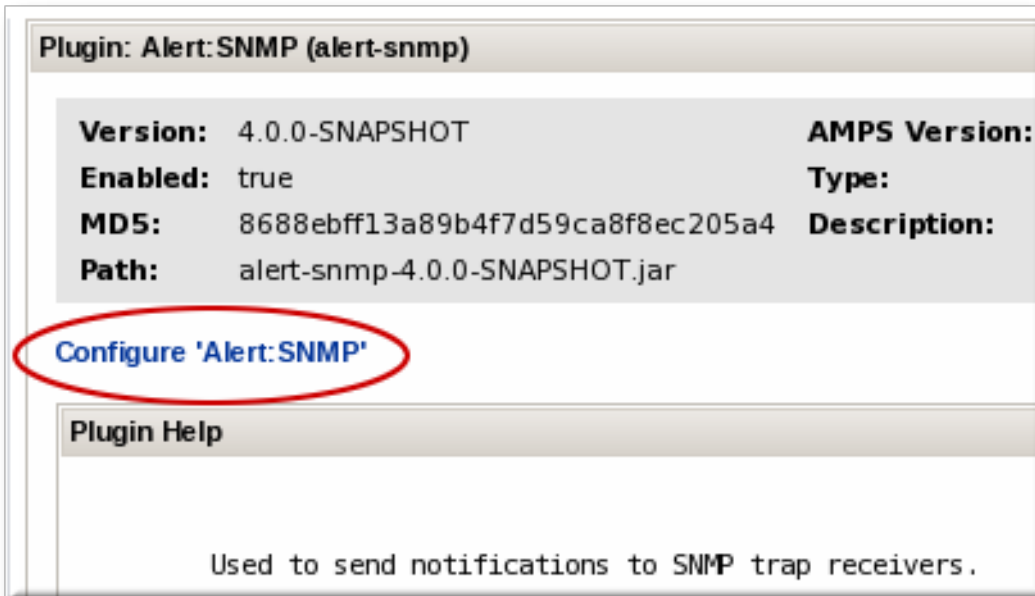


3. Click the **Server Plugins** tab.

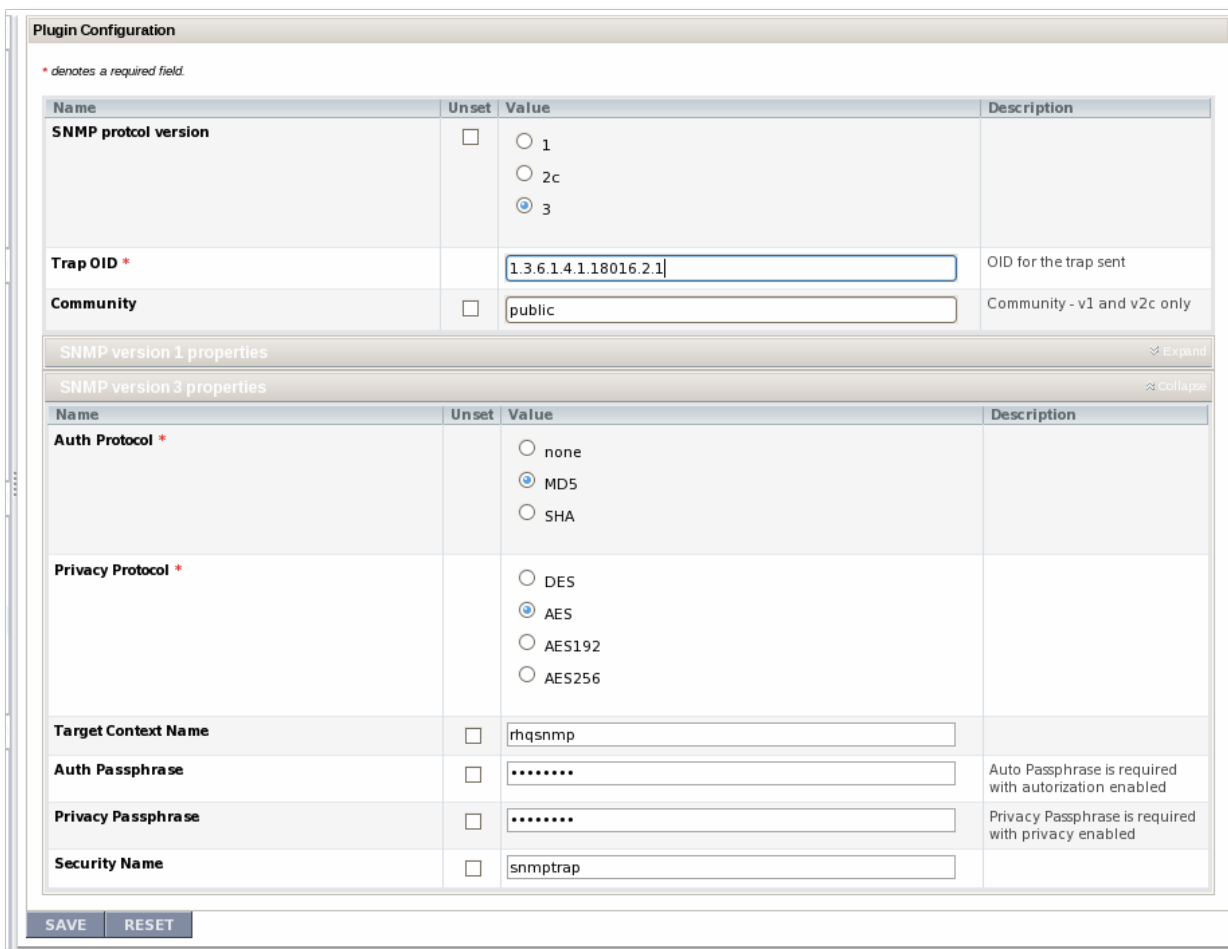


4. Click the name of the SNMP plug-in in the list.

- In the plug-in details page, click the **Configure 'Alert:SNMP'** link to open the configuration page for the plug-in.



- Click the **EDIT** button at the bottom of the configuration screen to make the fields active.
- All SNMP versions require the JBoss ON MIB OID (**1.3.6.1.4.1.18016.2.1**) and selected version, plus optional information to access the trap. Expand the version-specific configuration section and fill in the information about the SNMP agent.



- Once SNMP is set up for the server, reconfigure all of the SNMP alert notification senders.

3.8.3. Resetting Passwords for Configured Content Repositories

Password obfuscation was introduced in JBoss Operations Network 3.1 for stored content providers credentials, including the credentials used to access the JBoss Customer Portal (a default repository in JBoss ON). The methods used to obfuscate, store, and retrieve the content repository credentials in the database are updated in JBoss ON 3.1.1.

When upgrading from JBoss ON 3.1 to JBoss ON 3.1.2, any stored credentials for an existing content repository must be updated in order to apply the new obfuscation method.

Passwords must be changed both for the user account on the repository itself and for the repository configuration in JBoss ON.



Note

The password change is only necessary when upgrading from JBoss ON 3.1. It is not required when upgrading from an earlier version of JBoss ON to JBoss ON 3.1.2.

3.8.4. Upgrading the JBoss EAP 6 Resource Plug-in

JBoss Operations Network 3.0 and 3.0.1 had a tech preview version of the JBoss Enterprise Application Platform (EAP) 6 agent plug-in. With normal upgrade processes, agent plug-ins are updated when the JBoss ON server is updated. **However, tech preview plug-ins are not upgraded. The tech preview version of the plug-in must be deleted, and then the new plug-in installed.**

The EAP 6 tech preview plug-in and the EAP 6 GA plug-in are treated as two separate plug-ins by JBoss ON. If the tech preview version is not deleted, then any EAP 6 resource will be discovered and listed twice in the inventory, once discovered by the tech preview plug-in and discovered again by the GA plug-in.

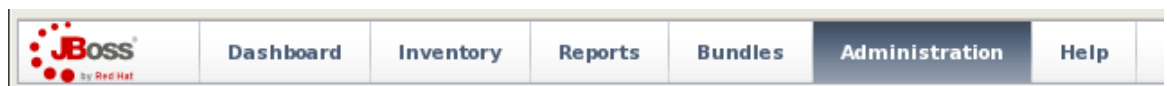


Note

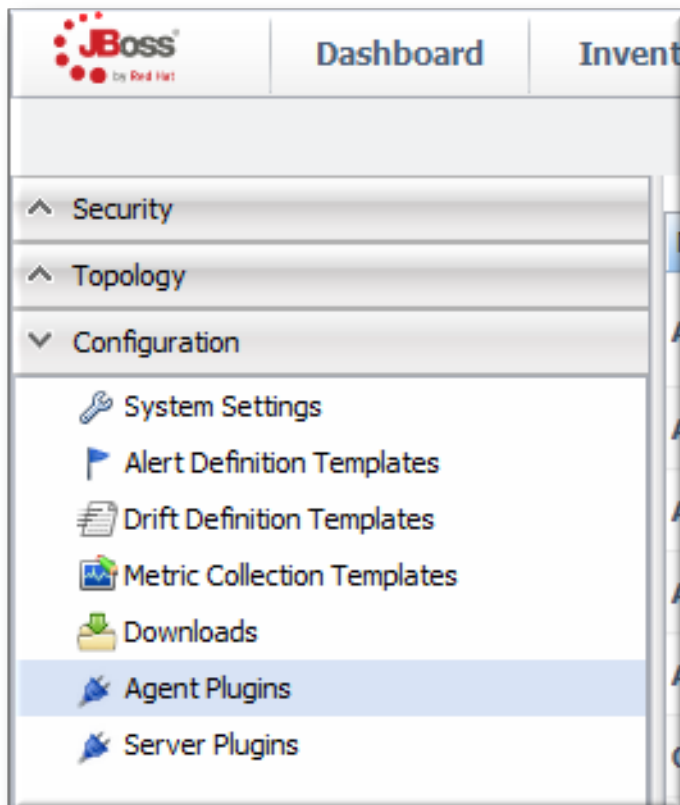
Any configuration, monitoring data and baselines, and resource histories will be lost after upgrading to the new EAP 6 plug-in.

To load the new EAP 6 plug-in:

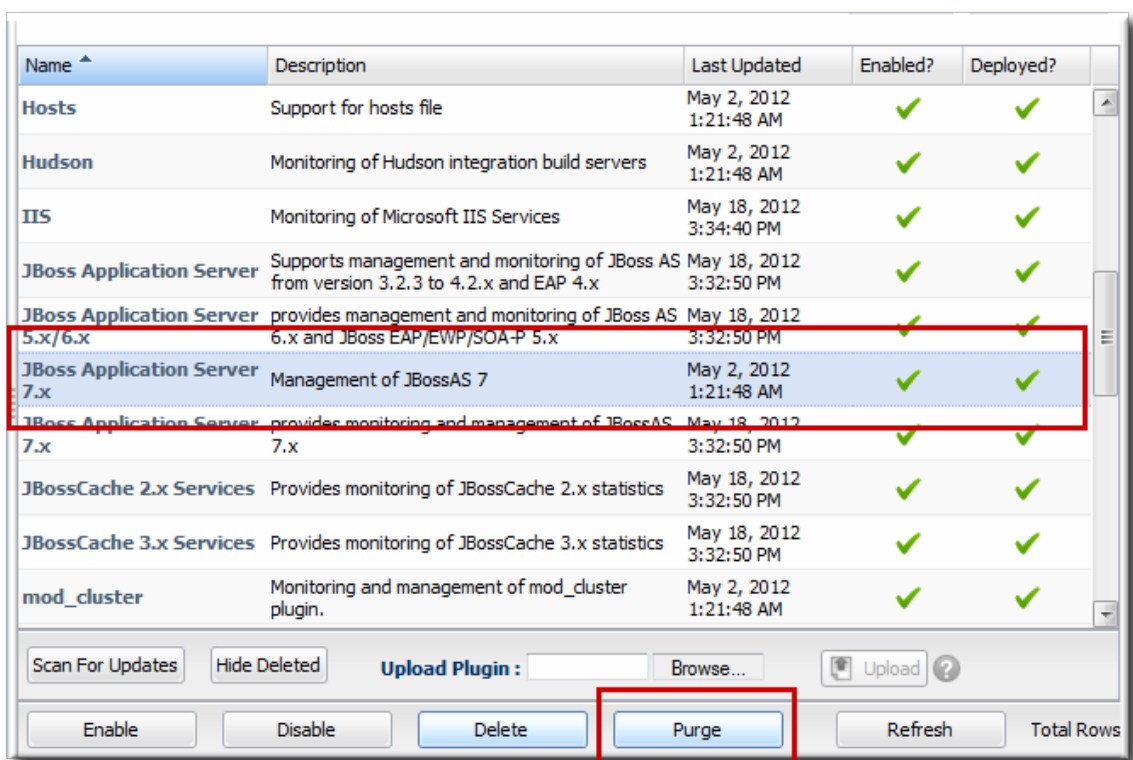
1. Delete the original tech preview plug-in and purge it from the JBoss ON database. Purging the plug-in allows the server to deploy the new plug-in in its place.
 - a. In the top menu, click the **Administration** tab.



- b. In the **Configuration** box on the left navigation bar, click the **Agent Plugins** link.



- c. Select the EAP 6 tech preview plug-in.
- d. Click the **DeLeTe** button.



- e. Click the **SHOW DELETED** button at the bottom of the plug-ins list.
- f. Select the EAP 6 plug-in, and then click the **PURGE** button. This removes the entry in the JBoss ON database that tells the servers to ignore that original tech preview plug-in and any updates to it.

Name ^	Description	Last Updated	Enabled?	Deployed?
Hosts	Support for hosts file	May 2, 2012 1:21:48 AM	✓	✓
Hudson	Monitoring of Hudson integration build servers	May 2, 2012 1:21:48 AM	✓	✓
IIS	Monitoring of Microsoft IIS Services	May 18, 2012 3:34:40 PM	✓	✓
JBoss Application Server	Supports management and monitoring of JBoss AS from version 3.2.3 to 4.2.x and EAP 4.x	May 18, 2012 3:32:50 PM	✓	✓
JBoss Application Server 5.x/6.x	provides management and monitoring of JBoss AS 6.x and JBoss EAP/EWP/SOA-P 5.x	May 18, 2012 3:32:50 PM	✓	✓
JBoss Application Server 7.x	Management of JBossAS 7	May 2, 2012 1:21:48 AM	!	!
JBoss Application Server 7.x	provides monitoring and management of JBossAS 7.x	May 18, 2012 3:32:50 PM	✓	✓
JBossCache 2.x Services	Provides monitoring of JBossCache 2.x statistics	May 18, 2012 3:32:50 PM	✓	✓
JBossCache 3.x Services	Provides monitoring of JBossCache 3.x statistics	May 18, 2012 3:32:50 PM	✓	✓
mod_cluster	Monitoring and management of mod_cluster plugin.	May 2, 2012 1:21:48 AM	✓	✓

Total Rows



Important

Wait for the purge operation to complete before continuing with the upgrade process.

2. If the server has not already been upgraded, upgrade the JBoss ON server, as described in [Section 3.8.1, "Upgrading the JBoss ON Server"](#).
3. Install the EAP 6 plug-in pack, as described in [Chapter 5, Installing JBoss Agent Plug-in Packs](#).
4. Import the EAP 6 server and its children.

Configuring and managing resources for EAP 6 domains and standalone servers is described in [How to Manage JBoss Servers](#).

Chapter 4. Installing and Upgrading the JBoss ON Server on Windows

4.1. Preparing for Installation

4.1.1. Setting up the JDK

The JBoss ON server requires Java 6 or Java 7 JDK.

1. Download and install the appropriate version of Java, if necessary.
2. Set the **JAVA_HOME** environment variable to the installation directory.

```
C:\>set JAVA_HOME=C:\Program Files (x86)\Java
```

The default Java service wrapper included with JBoss ON requires a 32-bit JVM, so the Java preference set for the server must be a 32-bit JDK. **The JBoss ON server must use a 32-bit JVM even on 64-bit systems.**

3. Set the system to use the **bin** directory of the correct version of the JDK.

```
C:>set path C:\Program Files\Java\jdk1.6.0_29\bin
```

4. Set the classpath to the of the correct version of the JRE distribution.

```
C:>set classpath C:\Program Files (x86)\Java\jre6\lib\ext\QTJava.zip
```

4.1.2. JVM for Running as a Service

JBoss ON includes Tanuki Software's Java service wrapper so that the JBoss ON server can be configured to run as a Windows service. That default Java service wrapper included with JBoss ON requires a 32-bit JVM, so the Java preference set for the server must be a 32-bit JDK.

```
RHQ_SERVER_JAVA_HOME=C:\Program Files\Java\jdk1.6.0_29
```

The JBoss ON server must use a 32-bit JVM even on 64-bit systems.

Running the server or agent with a 32-bit JVM does not in any way affect how JBoss ON manages other resources which may run with a 64-bit JVM. JBoss ON can still manage those resources and those resources can still use the 64-bit Java libraries for their own processes.

To run the server in a 64-bit JVM, an alternative Java wrapper service must be obtained independently from Tanuki Software or from another Java service vendor.

4.1.3. Preparing the Host Machine

For JBoss ON servers and agents to be able to communicate, they have to be able to connect to each other's host machines. Make sure that the machines are configured so that the server and agent machines are able to locate and connect with each other. There are three areas that commonly need to be configured:

- ✦ **Synchronize machine clocks.** All JBoss ON servers and agents must have synchronized clocks. Clock variations cause issues in availability reporting, metric measurements, graphing, and even identifying and importing resources into inventory. The Network Time Protocol project, <http://www.ntp.org/>, has information on installing and configuring NTP to ensure your clocks are synchronized.
- ✦ **Configure DNS.** Both forward and reverse DNS mapping entries must be present for all hosts involved in monitoring. This includes all JBoss ON servers and all machines running agents.
- ✦ **Configure the firewall to allow communication over the server and agent ports.** Ensure the necessary ports have been opened to prevent the firewall from blocking the JBoss ON server and agents from communicating. The JBoss ON server typically uses port 7080, and the JBoss ON agents typically use port 16163.

4.1.4. Selecting Path Names

Make sure that the *complete* path name for the server installation directory is relatively short. Path names longer than 19 characters can cause problems with executing some server tasks. Use a location such as **C:\jon** rather than **C:\Documents and Settings\myusername\jon-server**.

Windows' handling of file and path names is covered in [the "Naming Files, Paths, and Namespaces" in the Windows Data Access and Storage API](#).

4.1.5. Utilities to Use with JBoss ON

The only utilities used to manage the JBoss ON server are a ZIP utility to install the binaries and, possibly, a text editor to view and edit configuration files.

The recommended ZIP utility is WinZip. Examples in this guide usually use the Windows command prompt, so, optionally, install the WinZip CLI utility add-on. WinZip downloads are available at <http://www.winzip.com>.

4.2. Installing the Server JAR File

1. Log into the Windows machine as Administrator. The server and plug-in packages must be installed as the Administrator account.
2. Stop any currently running JBoss ON instances.

```
C:\rhq\jon-server-3.1.2.0.GA1\bin\rhq-server.bat stop
```



Warning

If the new JBoss ON server will use a database that existing JBoss ON instances are also using, then all of the existing JBoss ON instances have to be stopped. Otherwise, the installer will hang when it tries to contact the database and the database is unavailable because it is in use by another JBoss ON server.

3. Download the JBoss ON binaries from the [Customer Support Portal](#).
 - a. In the Customer Support Portal, click **Software**, and then select **JBoss Operations Network** in the product drop-down box.
 - b. Download the **JBoss Operations Network 3.1.2 Base Distribution** package by clicking the **Download** icon.

4. Create a directory for the server to be installed in.

Use a relatively short name. Path names longer than 19 characters can cause problems running the server or executing some tasks.

5. Unzip the server distribution to the desired installation directory.

```
C:> winzip32 -e jon-server-3.1.2.0.GA1.zip C:\jon\jon-server-3.1.2.0.GA1
```

6. Set the directory path to the JDK installation for a 32-bit JDK. For example:

```
set RHQ_SERVER_JAVA_HOME=C:\Program Files\Java\jdk1.6.0_29
```

The default Java service wrapper included with JBoss ON requires a 32-bit JVM, so the Java preference set for the server must be a 32-bit JDK. **The JBoss ON server must use a 32-bit JVM even on 64-bit systems.**

Running the server or agent with a 32-bit JVM does not in any way affect how JBoss ON manages other resources which may run with a 64-bit JVM. JBoss ON can still manage those resources and those resources can still use the 64-bit Java libraries for their own processes.

7. Install the JBoss ON server as a Windows service. This action must be "Run as Administrator."

```
C:\rhq\jon-server-3.1.2.0.GA1\bin\rhq-server.bat install
```

8. Start the JBoss ON server. This action must be "Run as Administrator."

```
C:\rhq\jon-server-3.1.2.0.GA1\bin\rhq-server.bat start
```

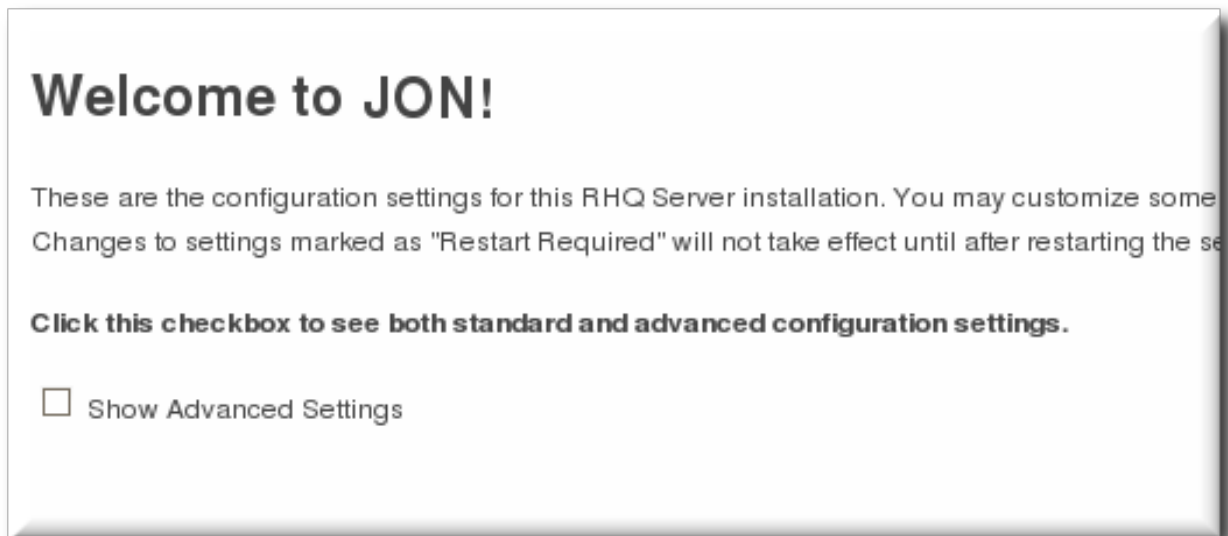
9. Set up the JBoss ON server using the web installer ([Section 4.3, "Going Through the Web Installer"](#)) or by editing the configuration file ([Section 4.4, "Silently Installing the Server"](#)).

4.3. Going Through the Web Installer

1. Open the server UI at <http://localhost:7080/>.



2. Clicking the **Click here to continue the installation** link brings you to the main installer page.
3. By default, the installer displays only the typical settings required for installation. For more custom environments, click the **Show Advanced Settings** check box to display the advanced settings in the next two sections.



Important

Do not unselect the advanced settings checkbox after setting advanced values, or the advanced settings will be reset to the default values.

4. Set the database connection properties. This should include the database type, the database hostname and port, and the name of the database created for JBoss ON (in this example, **rhq**).

The installation options are slightly different depending on the database configuration.

Database Settings define the database configured for this installation. All are required. Use the "Test Connection" button to validate the settings.

Database Type	PostgreSQL
Database Connection URL	jdbc:postgresql://127.0.0.1:5432/rhq
Database JDBC Driver Class	org.postgresql.Driver
Database XA DataSource Class	org.postgresql.xa.PGXADDataSource
Database User Name	rhqadmin
Database Password	

Confirm database settings

Test Connection



Important

With Oracle, selecting the overwrite tables option when there is nothing to overwrite causes an error message with **ErrorCode=[2289]**. This can be ignored.

- Set the basic connection information for the new JBoss ON server, such as its HTTP and HTTP ports.

Installation Settings define the required server endpoint for this installation.

	Requires Restart?
Server Name	localhost.localdomain No
Server Public Address	localhost.localdomain No
HTTP Port	7080 Yes
Secure HTTPS Port	7443 Yes

- Set preliminary notification information for the JBoss ON server for alerts to use for SNMP and email notifications.

Server Settings configure the server for this installation. All server settings are required.

	Requires Restart?
Server Bind Address	0.0.0.0 Yes
Embedded Agent Enabled	<input type="radio"/> Yes <input checked="" type="radio"/> No No
Email SMTP Hostname	localhost No
Email From Address	rhqadmin@localhost No

Install Server!

- Click the **INSTALL** button to begin configuring the server. This can take several minutes, and a loading screen will be up until the server is configured and a redirect is available.



8. When the server is configured, click the link to go to the UI.
9. Log into the GUI using the user account with the default superuser and password, **rhqadmin/rhqadmin**. The default username and password are predefined for the superuser; the password can be reset later.



Note

Change the password for the superuser account, **rhqadmin**, when you first log in. Passwords are changed in the **Administration > Users** area.

10. *Optional.* Install any additional agent plug-ins, as in [Chapter 5, Installing JBoss Agent Plug-in Packs](#).
11. *Optional.* Install the JBoss ON CLI, as described in [Chapter 6, Installing the JBoss ON CLI](#).
12. Begin configuring JBoss ON agent and resources, as in [Section 3.5, “Additional Post-Setup Checklist”](#).

4.4. Silently Installing the Server

The initial setup of the server is the same as in [Section 4.2, “Installing the Server JAR File”](#). Instead of using the visual, web-based installer, however, a silent installation loads a pre-configured properties file and starts the server with all of its configuration in place.

1. Stop any currently running JBoss ON instances.

```
C:\rhq\jon-server-3.1.2.0.GA1\bin\rhq-server.bat stop
```



Warning

If the new JBoss ON server will use a database that existing JBoss ON instances are also using, then all of the existing JBoss ON instances have to be stopped. Otherwise, the installer will hang when it tries to contact the database and the database is unavailable because it is in use by another JBoss ON server.

2. Install the server JAR file, as in [Section 4.2, “Installing the Server JAR File”](#).
3. Install the agent plug-ins.

Unzip the plug-in pack archive, and copy the files into the **C:\rhq\plugins** directory.

4. Open the JBoss ON server **rhq-server.properties** file for editing.
5. Set the **rhq.autoinstall.enabled** parameter to true to instruct the server to load the configuration automatically from file.

```
rhq.autoinstall.enabled=true
```

6. Set the parameter to tell the server whether preserve the any existing data in the database. For a new installation, this can be set to **auto** so the installer will supply the database schema.

```
rhq.autoinstall.database=auto
```



Important

The autoinstaller options are only evaluated once, when the JBoss ON server is first installed. After that initial configuration, the autoinstaller is disabled. These properties are ignored once the server is set up and cannot be used to initiate a re-install of an existing instance.

7. Optionally, set the IP address or hostname for the server. If this is not set, then the server will detect it when it starts.

```
rhq.autoinstall.public-endpoint-address=server1.example.com
```

8. Set the port numbers for the instance.

```
rhq.server.startup.web.http.port=7080
rhq.server.startup.web.https.port=7443
```

9. Set the username and password to connect to the backend database. The password is stored in the properties file in an encoded form. Use the **generate-db-password.sh** script to hash the password.

```
serverRoot/jon-server-3.1.2.0.GA1/bin/generate-db-password.sh
mypassword
Encoded password: 1d31b70b3650168f79edee9e04977e34
```

Then, set the hashed password as the **rhq.server.database.password** value.

```
rhq.server.database.user-name=rhqadmin
rhq.server.database.password=1d31b70b3650168f79edee9e04977e34
```

10. Scan the rest of the database configuration to make sure that it corresponds to your specific database type and instance. The parameters, and some possible values for PostgreSQL and Oracle databases, are described in [Table 4.1, “rhq-server.properties Parameters for Database Configuration”](#).

```
# Database
rhq.server.database.connection-
url=jdbc:postgresql://127.0.0.1:5432/rhq
```

```
rhq.server.database.driver-class=org.postgresql.Driver
rhq.server.database.xa-datasource-
class=org.postgresql.xa.PGXADatasource
rhq.server.database.type-mapping=PostgreSQL
rhq.server.database.server-name=127.0.0.1
rhq.server.database.port=5432
rhq.server.database.db-name=rhq
hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect

# Quartz
rhq.server.quartz.driverDelegateClass=org.quartz.impl.jdbcjobstore.Pos
tgreSQLDelegate
rhq.server.quartz.selectWithLockSQL=SELECT * FROM {0}LOCKS ROWLOCK
WHERE LOCK_NAME = ? FOR UPDATE
rhq.server.quartz.lockHandlerClass=org.quartz.impl.jdbcjobstore.StdRow
LockSemaphore
```

11. Set the directory path to the JDK installation for a 32-bit Java library. For example:

```
set RHQ_SERVER_JAVA_HOME=C:\Program Files\Java\jdk1.6.0_29
```

The default Java service wrapper included with JBoss ON requires a 32-bit JVM, so the Java preference set for the server must be a 32-bit JDK. **The JBoss ON server must use a 32-bit JVM even on 64-bit systems.**

Running the server or agent with a 32-bit JVM does not in any way affect how JBoss ON manages other resources which may run with a 64-bit JVM. JBoss ON can still manage those resources and those resources can still use the 64-bit Java libraries for their own processes.

12. Install the JBoss ON server as a Windows service. This action must be "Run as Administrator."

```
C:\rhq\jon-server-3.1.2.0.GA1\bin\rhq-server.bat install
```

13. Start the JBoss ON server. This action must be "Run as Administrator."

```
C:\rhq\jon-server-3.1.2.0.GA1\bin\rhq-server.bat start
```

When the server starts, it loads the edited **rhq-server.properties** file and is fully configured.

Table 4.1. rhq-server.properties Parameters for Database Configuration

Parameter	Description
rhq.server.database.type-mapping	Gives the type or vendor of the database that is used by the JBoss ON server. This is either PostgreSQL or Oracle.
rhq.server.database.connection-url	The JDBC URL that the JBoss ON server uses when connecting to the database. An example is <i>jdbc:postgresql://localhost:5432/rhq</i> or <i>jdbc:oracle:oci:@localhost:1521:orcl</i> .
rhq.server.database.driver-class	The fully qualified class name of the JDBC driver that the JBoss ON server uses to communicate with the database. An example is <i>oracle.jdbc.driver.OracleDriver</i> .

Parameter	Description
rhq.server.database.xa-datasource-class	The fully qualified class name of the JDBC driver that the JBoss ON server uses to communicate with the database. Examples of this are <i>org.postgresql.xa.PGXADatasource</i> or <i>oracle.jdbc.xa.client.OracleXADatasource</i> .
rhq.server.database.user-name	The name of the user that the JBoss ON server uses when logging into the database
rhq.server.database.password	The password of the database user that is used by the JBoss ON server when logging into the database. This password is stored in a hash.
rhq.server.database.server-name	The server name where the database is found. This must match the server in the connection URL. This is currently only used when connecting to PostgreSQL.
rhq.server.database.port	The port on which the database is listening. This must match the port in the connection URL. This is currently only used when connecting to PostgreSQL.
rhq.server.database.db-name	The name of the database. This must match the name found in the connection URL. This is currently only used when connecting to PostgreSQL.
rhq.server.quartz.driverDelegateClass	The Quartz driver used for connections between the server and the database. The value of this is set by the installer and depends on the type of database used to store the JBoss ON information. For PostgreSQL, this is org.quartz.impl.jdbcjobstore.PostgreSQLDelegate , and for Oracle, this is org.quartz.impl.jdbcjobstore.oracle.OracleDelegate .

4.5. Additional Post-Setup Checklist

Once the JBoss ON server is configured, there are a number of different areas that administrators can configure to set up and manage resource in JBoss ON and to configure JBoss ON itself.

Most of the initial setup — building out the JBoss ON inventory with managed resources, setting up roles and access control, and creating resource groups — is covered in [Initial Setup: the Resource Inventory, Groups, and Users](#). JBoss ON has a number of other features like drift monitoring, alerting, application and content deployment, server high availability, and configuration management. The way that these are configured depends on the demands of your infrastructure. [Table 4.2, “Configuration and Features to Set Up”](#) has links to procedures for features which are commonly set up soon after JBoss ON is set up; browse the JBoss ON GUI and documentation for a more comprehensive view of JBoss ON's capabilities.

Table 4.2. Configuration and Features to Set Up

Feature	Description	Doc Link
---------	-------------	----------

Feature	Description	Doc Link
Creating resource groups	For simplicity and effectiveness with managing resources, JBoss ON has resource groups. There are a number of different types of groups, including compatible groups (groups of the same type of resource) and dynamically-created groups. This makes it possible to apply configuration changes, alert definitions, drift definitions, and other settings all at once and to create useful groups for better monitoring and inventory tracking.	"Managing Groups"
Configuring roles and access control	JBoss ON defines access to resources based on <i>roles</i> . Both users and resources are assigned to roles, and then access control rights are assigned to the roles.	"About Security in JBoss ON: Roles and Access Control"
LDAP user authentication	By default, users are created in JBoss ON and then stored in the JBoss ON database. However, JBoss ON can be configured to check an LDAP server first for user accounts, so existing LDAP users can be authenticated in JBoss ON — without having to create users in JBoss ON first.	"How JBoss ON Uses LDAP for Authentication"
LDAP group authorization	LDAP groups can be associated with JBoss ON roles. This means that the members in the LDAP group are automatically granted the rights and can manage the resources defined in the JBoss ON role.	"How JBoss ON Roles Work with LDAP User Groups"
Creating alerts	An alert is a way of informing administrators that some event or condition has occurred on a resource. JBoss ON provides a number of different ways to set and respond to alerts, both by sending notifications and by taking a specified action.	"Brief Introduction to Alerts and Notifications"
Configuring drift monitoring	When configuration settings change, the configuration <i>drifts</i> from its designated state. Drift monitoring provides a mechanism for administrators to define and manage that designated configuration state, to apply it to multiple resources, and to be informed if any resource drifts from that state.	"Understanding Drift"

Feature	Description	Doc Link
Setting up bundles for provisioning	Provisioning is a way of deploying content through JBoss ON. Provisioning takes a <i>bundle</i> , an archive file, and deploys it on a platform or a JBoss resource. This bundle can be a configuration file, a set of EARs or WARs, or even a full application server. JBoss ON supports multiple versions of the same bundle and can deploy these bundles to different resources, all managed through a single point in JBoss ON.	"Creating Bundles"
Setting affinity and high availability	When multiple JBoss ON servers are added, they naturally establish a high availability and failover topology. Agents can be managed by any server in the cloud, but it is possible to set a preference, or <i>affinity</i> , for agents to be managed by selected servers. This can improve performance or be used to reflect the infrastructure topology.	"Configuring High Availability"

4.6. Configuring the Server as a Windows Service

The `rhq-server.bat` script has an installation option that installs the script as a Windows service.

1. Set the environment variable for the user to run the Windows service as.

Every Windows service has to run as some system user. There are two environment variables in the `rhq-server.bat` script that set the user to use:

- ✦ `RHQ_SERVER_RUN_AS` sets any Windows user to be the JBoss ON server user. The username given here must be in the standard Windows format, `DOMAIN\user`, such as `EXAMPLEDOMAIN\jsmith`.
- ✦ `RHQ_SERVER_RUN_AS_ME` sets the server to run as whoever the current user is. This overrides the `RHQ_SERVER_RUN_AS`, if both are set.

If neither environment variable is set, then the JBoss ON server runs as the system account.

2. Run the `rhq-server.bat` script with the `install` option to set up the service. This prompts for the password of whatever user account is used for the JBoss ON service.

```
serverRoot\bin\rhq-server.bat install
```

4.7. Re-Installing the Server

Whether it is configured through the web UI or silently, the JBoss ON server uses an autoinstaller to configure itself initially. Once that initial configuration is complete, the autoinstaller is disabled (so even setting

the *rhq.autoinstaller.** properties in `rhq-server.properties` does not re-initiate the server configuration).

To re-install the server, remove the entire home directory for the server, and then unzip the original JBoss ON server archive and configure the server as if it were all new, as in [Section 4.3, “Going Through the Web Installer”](#) or [Section 4.4, “Silently Installing the Server”](#).

4.8. Upgrading JBoss ON

An upgrade procedure for JBoss Operations Network essentially overlays the new JBoss ON packages and libraries over the existing configuration and databases. The upgrade procedure, then, is very similar to the installation process. The new packages need to be installed, and then the server is set up in the same setup wizard. The difference is that the server reuses its existing databases and data so that the configuration from the previous installation is preserved.

Upgrade to JBoss Operations Network 3.1.2 is only supported from JBoss ON 2.x versions and later.



Note

The JBoss ON servers must be upgraded before the JBoss ON agents can be upgraded.



Warning

There will be a minimal loss monitoring data because of the downtime required when the server and agents are being upgraded. Additionally, any monitoring data for the JBoss ON server will be lost, if the server is included in the inventory.

4.8.1. Upgrading the JBoss ON Server

Not every step in this upgrade procedure applies to every JBoss Operations Network installation. Just run through the steps in order, and perform the ones necessary for your deployment.



Warning

Upgrading the JBoss ON server essentially creates a new server instance that replaces the old instance. If the JBoss ON server was added to the inventory, then the old JBoss ON server resource must be deleted from the inventory because it will not be a usable resource after upgrade. Once the upgrade process is complete, then the JBoss ON server must be added to the inventory again and all of the previous configuration for that resource (like alerts, scheduled operations, and group membership) must be redone.



Note

See [Chapter 7, Troubleshooting Installation and Upgrade](#) if there are any problems during the upgrade process.

1. First, do some prep work on the JBoss ON configuration. It is easier to clean up the configuration before migration than it is after.
 - a. Remove any unused or out of service platforms from the inventory.
 - b. Remove any alert definitions which use conditions for obsolete metrics.

For migrating to JBoss ON 3.1.2, there are four alert conditions — all for PostgreSQL databases — which should be removed:

- ✦ User Time
 - ✦ Kernel Time
 - ✦ Physical Memory
 - ✦ Virtual Memory
2. Prepare the JBoss ON agents for upgrade. Agents will auto-upgrade, meaning that when they detect that the server has a new version, the agent will request an update. Follow the instructions at [Section 8.7, “About Agent Automatic Updates”](#) to prepare the agent, and then just leave it running. The agent should be running in the background to upgrade properly, as in [Section 8.4, “Running the JBoss ON Agent as a Service”](#).

In some rare cases, the agent will be upgraded manually instead of upgrading itself. In that case, stop the agent before upgrading the server, and follow the instructions at [Section 8.8, “Manually Upgrading the JBoss ON Agent”](#).

3. Stop the JBoss ON server which is being upgraded *as well as* any currently running JBoss ON instances. For example:

```
serverRoot/jon-server-3.1.2.0.GA1/bin/rhq-server.sh stop
```



Warning

If the upgraded JBoss ON server will use a database that existing JBoss ON instances are also using, then all of the existing JBoss ON instances have to be stopped. Otherwise, the installer will hang when it tries to contact the database and the database is unavailable because it is in use by another JBoss ON server.

4. Open the server root directory. For example:

```
cd /opt/jon
```

5. Unzip the server packages.

```
unzip jon-server-3.1.2.0.GA1.zip
```



Important

Do not copy the new server installation on top of a previous server installation.

The directory structure within the server package gives the new server installation directory a

version-specific name, such as `/opt/jon/jon-server-3.1.2.0.GA1`.

- Copy over any changes in your original `rhq-server.properties` file to the new file in `serverRoot/jon-server-3.1.2.0.GA1/bin`. Changes to this file include things like setting up SSL and enabling SMTP for email notifications.



Note

In JBoss ON 2.3.1 and older versions, the password to access the database is stored in plaintext. In JBoss ON 3.1.2, this password is hashed for security.



Note

If you don't want to edit the `rhq-server.properties` file manually, you can change the server settings to the proper configuration in the **Advanced Settings** form during the server setup.

- If the server was installed as a Windows service, then uninstall the Windows service for the original server:

```
cd c:\old-serverRoot\bin
./rhq-server.bat remove
```

Then install the new server as a Windows service:

```
cd c:\new-serverRoot\bin
./rhq-server.bat install
```

- Additional plug-in packs for specific needs (such as supporting management tasks for EWS, EAP, and SOA-P) are available to be installed separate from the core JBoss ON agent packages. Each plug-in pack as at least one (and sometimes more than one) agent plug-in. Each zip file for the plug-ins has a README.txt file with specific setup instructions.

The plug-in files can be unzipped anywhere. For example:

```
cd /opt/jon/jon-server-3.1.2.0.GA1
unzip jon-plugin-pack-agent_plugin_name-3.1.2.0.GA1.zip
```



Note

If there are multiple JBoss ON servers in a high availability setup, the agent plug-in pack only has to be installed once. The other servers will pick up the plug-ins as part of the high availability polls.

- Start the JBoss ON server. For example:

```
serverRoot/jon-server-3.1.2.0.GA1/bin/rhq-server.sh start
```

- Back up your server database before going through the setup wizard. In case there is a problem with the upgrade process, the backup allows you to restore to its previous state.
- Open the web UI.

```
http://hostname:7080
```

As with a new installation, the installer opens after you log in.

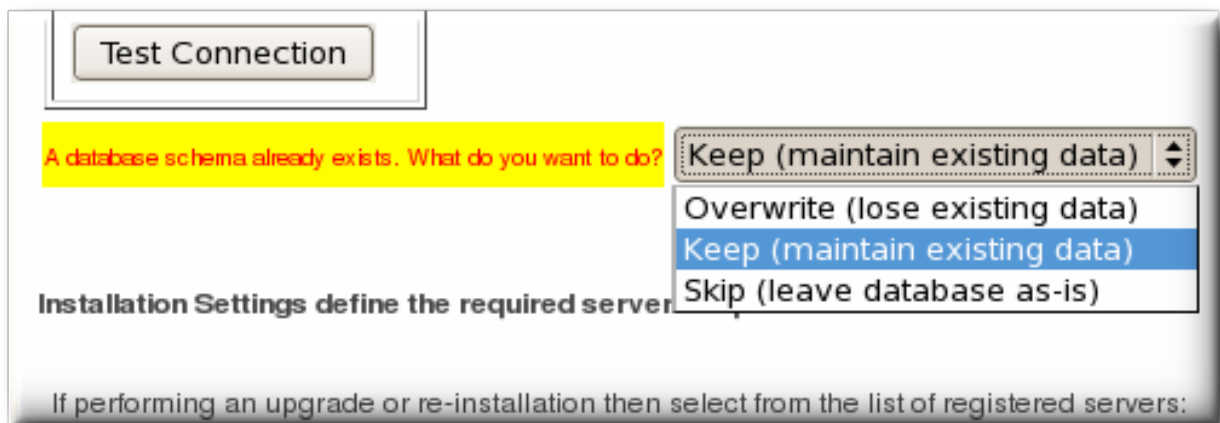
- The setup process is the same as the initial setup procedure in [Section 4.3, “Going Through the Web Installer”](#).



Warning

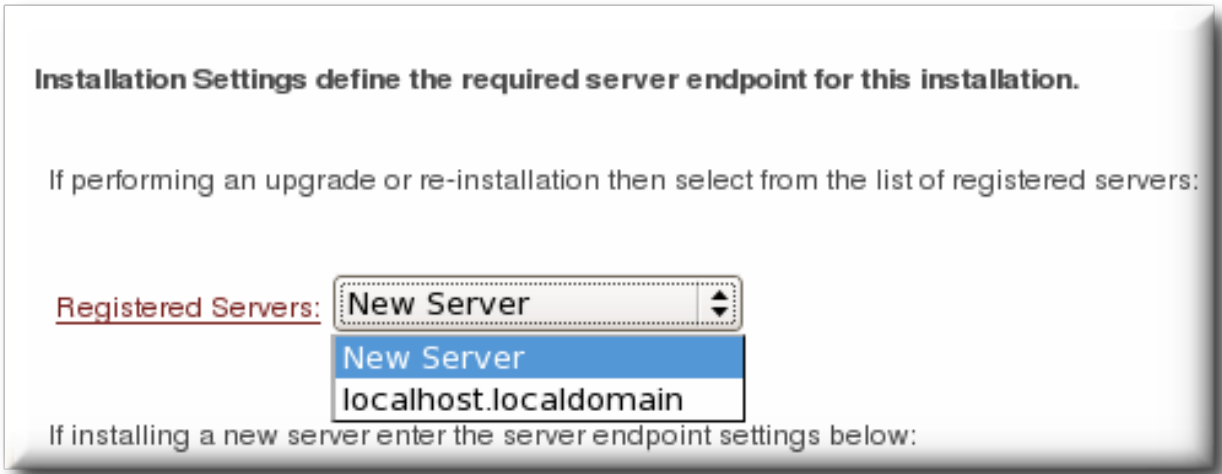
Do not change any of the settings for the server, especially identifying information such as the **Server Name** field. This can cause errors during the upgrade process.

When the database connection information is entered, the JBoss ON installer detects the existing JBoss ON database. This introduces a new field to the installer, prompting you for what to do with the existing database.



Choose the default, **Keep (maintain existing data)**. Do *not* choose **Overwrite (lose existing data)**, or the installer will delete all of your JBoss ON data, including your inventory, monitoring history, alerts, and metrics.

- The **Registered Servers** lists every server in the server cloud. For upgrades and re-installs, this gives you the option to keep the existing server configuration (such as ports and notification settings) or to set new values. To preserve the settings, select the server from the registered servers list; otherwise, select **New Server**.



14. Restart the browser after upgrade completes or force the browser to refresh, using **Ctrl+F5**.



Note

The browser caches an old session ID for the previous JBoss ON server installation. Attempting to open the GUI without refreshing the cache causes the login to fail and throws an error in the JBoss ON server logs:

```
Exception:
2012-05-23 16:37:08,046 ERROR
[org.apache.catalina.connector.CoyoteAdapter] An exception or
error occurred in the container during the request processing
java.lang.NullPointerException
    at
org.apache.catalina.connector.CoyoteAdapter.parseSessionCookiesId
(CoyoteAdapter.java:507)
    at
org.apache.catalina.connector.CoyoteAdapter.postParseRequest(Coyo
teAdapter.java:449)
    at
org.apache.catalina.connector.CoyoteAdapter.service(CoyoteAdapter
.java:239)
... 8< ...
```

15. Start any JBoss ON agents that were stopped for the upgrade process.
16. If the older JBoss ON server was added to the JBoss ON inventory, then remove it. The old JBoss ON server must be removed from the inventory because it is no longer a usable resource.
17. *Optional.* Add the new JBoss ON server as a resource in the inventory.

4.8.2. Migrating SNMP Settings

The SNMP settings for the server (unlike other server settings) are not migrated with the other server settings. These settings have to be restored manually after migration, and there are two ways to do this:

- ✦ Run a SQL command to copy the settings from the database table for the old server instance into the table for the new instance.

- ✦ Set the SNMP settings in the JBoss ON GUI.

4.8.2.1. Running SQL Commands to Migrate the SNMP Settings

1. Open the administrative page, with the location **admin/test/sql.jsp**. For example:

```
http://server.example.com:7080/admin/test/sql.jsp
```

2. Run the command to migrate the SNMP settings:

```
select property_key,property_value from RHQ_SYSTEM_CONFIG where  
property_key like 'SNMP%';
```

3. Click the **Execute SQL** button.
4. Reconfigure all of the SNMP alert notification senders.

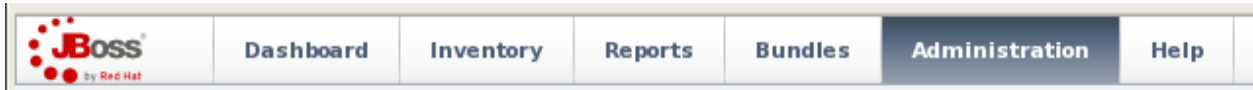
4.8.2.2. Configuring SNMP Settings in the GUI



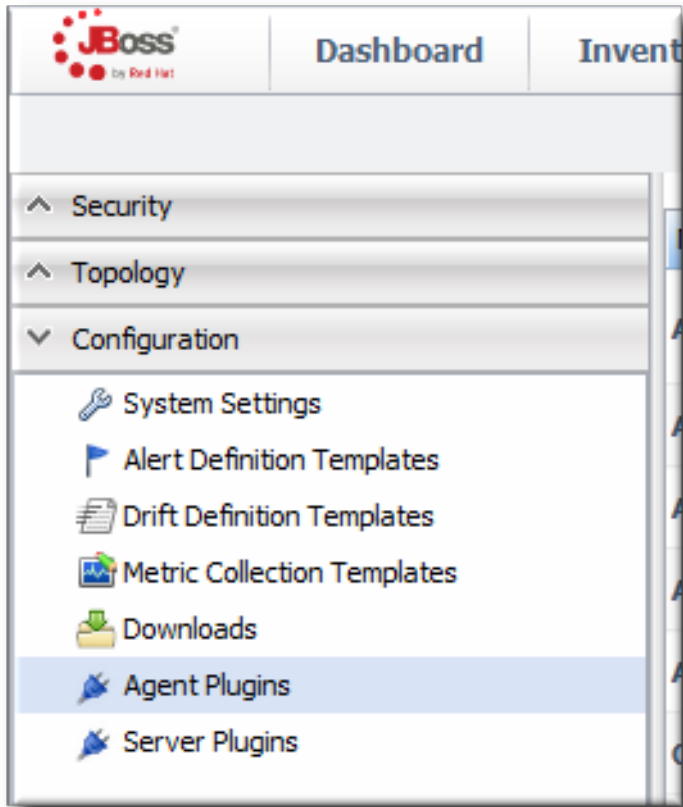
Note

Take a screenshot of the SNMP settings *before* beginning the upgrade procedure, so that the image is available as a reference to re-do the SNMP settings.

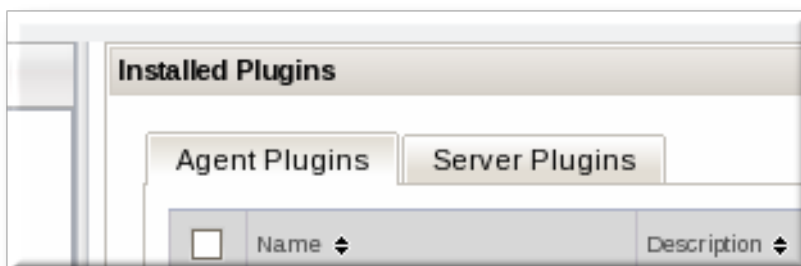
1. In the top menu, click the **Administration** tab.



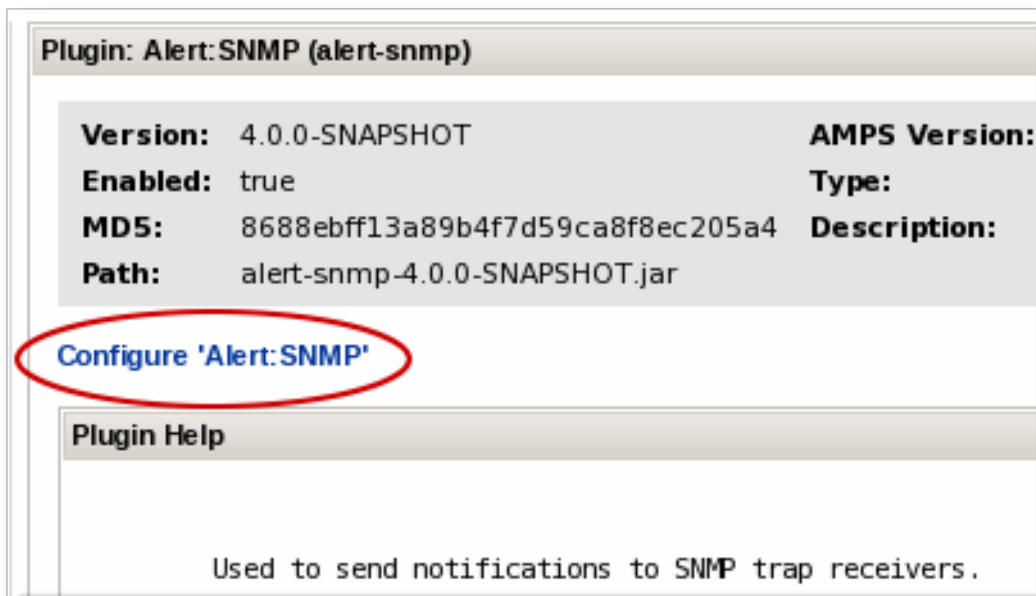
2. In the **Configuration** box on the left navigation bar, click the **Plugins** link.



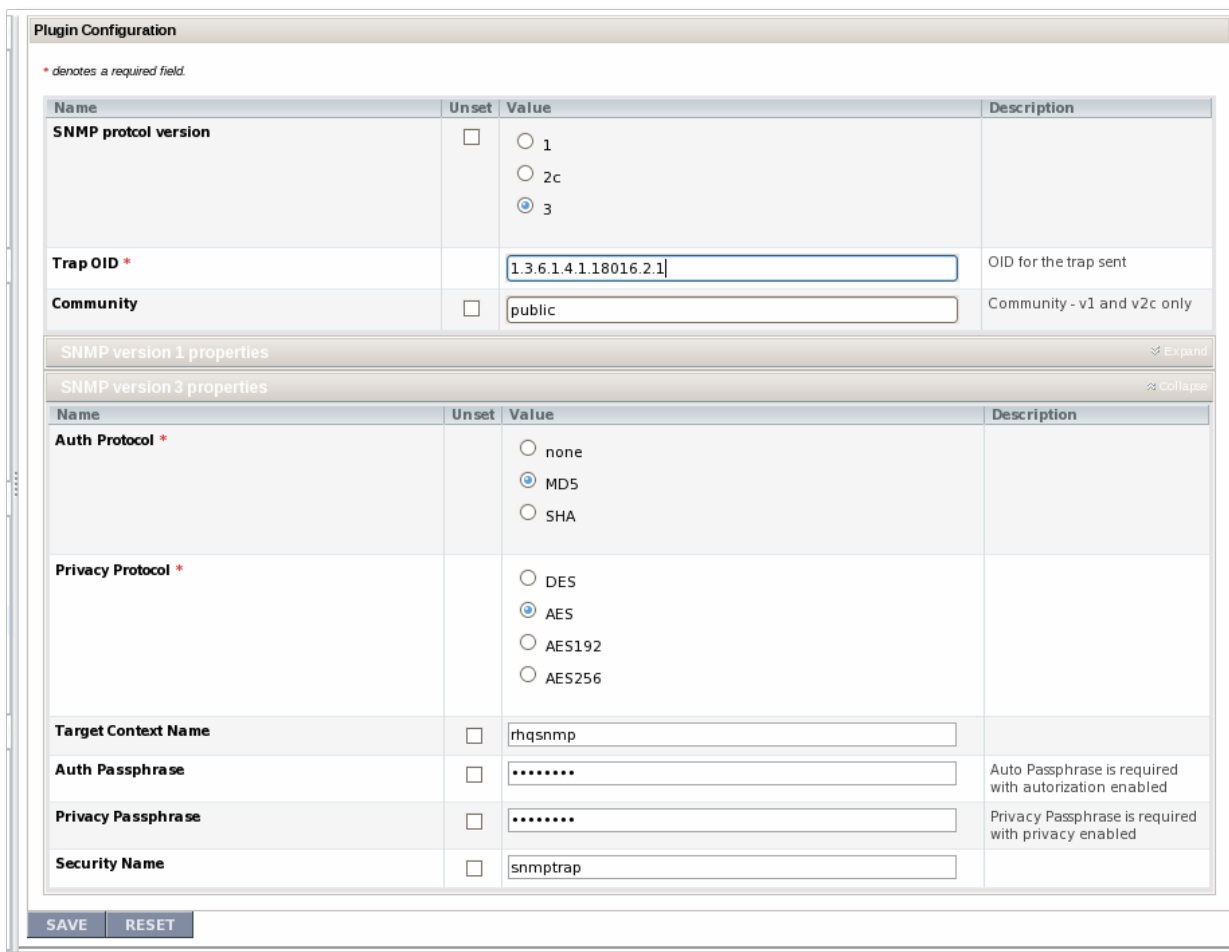
3. Click the **Server Plugins** tab.



4. Click the name of the SNMP plug-in in the list.
5. In the plug-in details page, click the **Configure 'Alert:SNMP'** link to open the configuration page for the plug-in.



6. Click the **EDIT** button at the bottom of the configuration screen to make the fields active.
7. All SNMP versions require the JBoss ON MIB OID (**1.3.6.1.4.1.18016.2.1**) and selected version, plus optional information to access the trap. Expand the version-specific configuration section and fill in the information about the SNMP agent.



Plugin Configuration

* denotes a required field.

Name	Unset	Value	Description
SNMP protocol version	<input type="checkbox"/>	<input type="radio"/> 1 <input type="radio"/> 2c <input checked="" type="radio"/> 3	
Trap OID *		<input type="text" value="1.3.6.1.4.1.18016.2.1"/>	OID for the trap sent
Community	<input type="checkbox"/>	<input type="text" value="public"/>	Community - v1 and v2c only
SNMP version 1 properties Expand			
SNMP version 3 properties Collapse			
Name	Unset	Value	Description
Auth Protocol *		<input type="radio"/> none <input checked="" type="radio"/> MD5 <input type="radio"/> SHA	
Privacy Protocol *		<input type="radio"/> DES <input checked="" type="radio"/> AES <input type="radio"/> AES192 <input type="radio"/> AES256	
Target Context Name	<input type="checkbox"/>	<input type="text" value="rhqsnmp"/>	
Auth Passphrase	<input type="checkbox"/>	<input type="text" value="....."/>	Auto Passphrase is required with authorization enabled
Privacy Passphrase	<input type="checkbox"/>	<input type="text" value="....."/>	Privacy Passphrase is required with privacy enabled
Security Name	<input type="checkbox"/>	<input type="text" value="snmptrap"/>	

SAVE **RESET**

8. Once SNMP is set up for the server, reconfigure all of the SNMP alert notification senders.

4.8.3. Upgrading the JBoss EAP 6 Resource Plug-in

JBoss Operations Network 3.0 and 3.0.1 had a tech preview version of the JBoss Enterprise Application Platform (EAP) 6 agent plug-in. With normal upgrade processes, agent plug-ins are updated when the JBoss ON server is updated. **However, tech preview plug-ins are not upgraded. The tech preview version of the plug-in must be deleted, and then the new plug-in installed.**

The EAP 6 tech preview plug-in and the EAP 6 GA plug-in are treated as two separate plug-ins by JBoss ON. If the tech preview version is not deleted, then any EAP 6 resource will be discovered and listed twice in the inventory, once discovered by the tech preview plug-in and discovered again by the GA plug-in.



Note

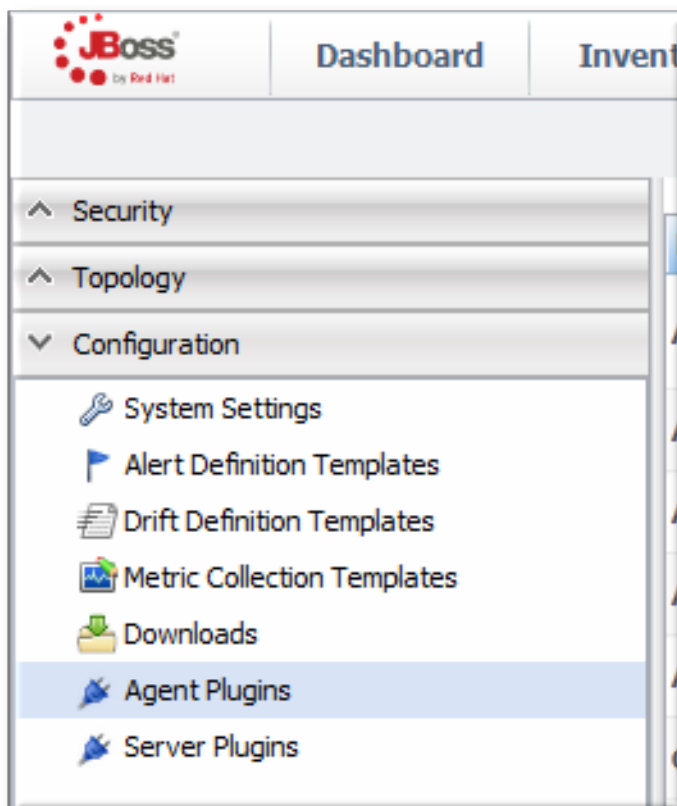
Any configuration, monitoring data and baselines, and resource histories will be lost after upgrading to the new EAP 6 plug-in.

To load the new EAP 6 plug-in:

1. Delete the original tech preview plug-in and purge it from the JBoss ON database. Purging the plug-in allows the server to deploy the new plug-in in its place.
 - a. In the top menu, click the **Administration** tab.



- b. In the **Configuration** box on the left navigation bar, click the **Agent Plugins** link.



- c. Select the EAP 6 tech preview plug-in.
 - d. Click the **Delete** button.

Name ^	Description	Last Updated	Enabled?	Deployed?
Hosts	Support for hosts file	May 2, 2012 1:21:48 AM	✓	✓
Hudson	Monitoring of Hudson integration build servers	May 2, 2012 1:21:48 AM	✓	✓
IIS	Monitoring of Microsoft IIS Services	May 18, 2012 3:34:40 PM	✓	✓
JBoss Application Server	Supports management and monitoring of JBoss AS from version 3.2.3 to 4.2.x and EAP 4.x	May 18, 2012 3:32:50 PM	✓	✓
JBoss Application Server 5.x/6.x	provides management and monitoring of JBoss AS 6.x and JBoss EAP/EWP/SOA-P 5.x	May 18, 2012 3:32:50 PM	✓	✓
JBoss Application Server 7.x	Management of JBossAS 7	May 2, 2012 1:21:48 AM	✓	✓
JBoss Application Server 7.x	provides monitoring and management of JBossAS 7.x	May 18, 2012 3:32:50 PM	✓	✓
JBossCache 2.x Services	Provides monitoring of JBossCache 2.x statistics	May 18, 2012 3:32:50 PM	✓	✓
JBossCache 3.x Services	Provides monitoring of JBossCache 3.x statistics	May 18, 2012 3:32:50 PM	✓	✓
mod_cluster	Monitoring and management of mod_cluster plugin.	May 2, 2012 1:21:48 AM	✓	✓

Total Rows

- e. Click the **SHOW DELETED** button at the bottom of the plug-ins list.
- f. Select the EAP 6 plug-in, and then click the **PURGE** button. This removes the entry in the JBoss ON database that tells the servers to ignore that original tech preview plug-in and any updates to it.

Name ^	Description	Last Updated	Enabled?	Deployed?
Hosts	Support for hosts file	May 2, 2012 1:21:48 AM	✓	✓
Hudson	Monitoring of Hudson integration build servers	May 2, 2012 1:21:48 AM	✓	✓
IIS	Monitoring of Microsoft IIS Services	May 18, 2012 3:34:40 PM	✓	✓
JBoss Application Server	Supports management and monitoring of JBoss AS from version 3.2.3 to 4.2.x and EAP 4.x	May 18, 2012 3:32:50 PM	✓	✓
JBoss Application Server 5.x/6.x	provides management and monitoring of JBoss AS 6.x and JBoss EAP/EWP/SOA-P 5.x	May 18, 2012 3:32:50 PM	✓	✓
JBoss Application Server 7.x	Management of JBossAS 7	May 2, 2012 1:21:48 AM	!	!
JBoss Application Server 7.x	provides monitoring and management of JBossAS 7.x	May 18, 2012 3:32:50 PM	✓	✓
JBossCache 2.x Services	Provides monitoring of JBossCache 2.x statistics	May 18, 2012 3:32:50 PM	✓	✓
JBossCache 3.x Services	Provides monitoring of JBossCache 3.x statistics	May 18, 2012 3:32:50 PM	✓	✓
mod_cluster	Monitoring and management of mod_cluster plugin.	May 2, 2012 1:21:48 AM	✓	✓

Total Rows



Important

Wait for the purge operation to complete before continuing with the upgrade process.

2. If the server has not already been upgraded, upgrade the JBoss ON server, as described in [Section 3.8.1, "Upgrading the JBoss ON Server"](#).
3. Install the EAP 6 plug-in pack, as described in [Chapter 5, *Installing JBoss Agent Plug-in Packs*](#).
4. Import the EAP 6 server and its children.

Configuring and managing resources for EAP 6 domains and standalone servers is described in [How to Manage JBoss Servers](#).

Chapter 5. Installing JBoss Agent Plug-in Packs

JBoss Operations Network has additional agent plug-ins to handle specific JBoss resources — EWS, EDS, EAP, or SOA-P. Although these are JBoss ON resource plug-ins, they are included in separate packages and require a separate subscription to download them.



Note

Installing the EWS, EDS, EAP, or SOA-P plug-ins *is not* the same as deploying a custom agent plug-in or a server-side plug-in. Deploying custom plug-ins is done after JBoss ON server installation and is described in the *Basic Admin Guide*.

1. Download the plug-in JAR files from the [Customer Support Portal](#).

In the Customer Support Portal, open the **Downloads** tab and select the **Downloads** item in the **JBoss ENTERPRISE MIDDLEWARE** area.

2. Select the **JBoss ON for Plug-in** product in the drop-down box.
3. Download the plug-in packs.
4. Extract the additional plug-in pack archives to a temporary location. This creates a subdirectory with the name **jon-plugin-pack-plugin_name-version**. For example:

```
[root@server rhq-agent]# unzip jon-plugin-pack-eap-3.1.2.0.GA1.zip -d /tmp
```

5. Copy the extracted plug-in JAR files from the **jon-plugin-pack-plugin_name-3.1.2.0.GA1/** directory to the JBoss ON server plug-in directory. For example:

```
[root@server rhq-agent]# cp /tmp/jon-plugin-pack-plugin_name-3.1.2.0.GA1/*.jar /opt/jon/jon-server-3.1.2.0.GA1/plugins
```

6. Have the JBoss ON server update its plug-ins. This can be done through the JBoss ON GUI or by restarting the server.

To load the plug-ins through the GUI:

- a. Open the **Administration** tab.
 - b. In the **Configuration** area on the left, select the **Agent Plug-ins** link.
 - c. At the bottom of the list of loaded agent plug-ins, click the **SCAN FOR UPDATES** button.
7. Have the agents reload their plug-ins to load the new plug-ins. This can be done from the agent's command prompt using the **plugins** command:

```
> plugins update
```

This can also be done in the JBoss ON GUI by scheduling an *update plugins* operation for an agent or a group of agents.

Alternatively, restart the agents.

Chapter 6. Installing the JBoss ON CLI

The JBoss Operations Network CLI is a standalone Java application that uses the [Java Scripting API](#) (this requires Java 6 or later). The CLI allows JBoss ON to be better integrated into the network environment by allowing administrators to interact with JBoss ON programmatically.



Note

Java 6 ships with the [Rhino](#) JavaScript engine, so JavaScript is the supported scripting language in the CLI.

A large subset of JBoss ON functionality is exposed in the CLI.

The JBoss ON server contains packages called the Remote Client, which contain the JBoss ON CLI packages, **rhq-client-3.1.2.zip**.



Note

Only the corresponding version of the CLI can be used to manage the JBoss ON server. Other versions are not compatible.

To install the CLI:

1. Open the JBoss ON GUI.

```
http://server.example.com:7080
```

2. Click the **Administration** link in the main menu.
3. Select the **Downloads** menu item.
4. Scroll to the **Command Line Client Download** section, and click **Download Client Installer**.
5. Save the **.zip** file into the directory where the CLI should be installed.
6. Unzip the packages.

```
unzip rhq-client-version.zip
```

Chapter 7. Troubleshooting Installation and Upgrade

This chapter looks at some of the most common issues that may be encountered after installing or upgrading the JBoss Operations Network server. Other issues are also covered in the JBoss ON frequently-asked-questions.

7.1. Exceptions and Error Logs

Q: During upgrade, I see the error *X is not a valid jarfile - it is either corrupted or file has not been fully written yet* and none of my plug-ins are updated.

A: When upgrading the server and installed JBoss plug-ins, there may be an error in the logs that one of the plug-in JAR files was corrupted:

```
2012-08-28 10:31:36,473 ERROR
[org.rhq.enterprise.server.core.plugin.PluginDeploymentScanner] Scan
failed. Cause: java.lang.Exception:File [/home/jon/jon-server-
3.1.0.GA/jbossas/server/default/deploy/rhq.ear/rhq-downloads/rhq-
plugins/hornetq-jopr-plugin-2.0.0.Final.jar] is not a valid jarfile -
it is either corrupted or file has not been fully written yet.
```

This appears to happen if there is an EAP instance in the inventory and several different JBoss plug-ins are installed, though it may occur in other situations.

That error prevents the other updated plug-ins deployed in the JBoss ON server's **plugins/** directory from being picked up.

Solution. If that error occurs, simply restart the JBoss ON server again. All of the plug-ins will be successfully deployed after restarting another time.

Q: I'm seeing null pointer exceptions for the **org.apache.catalina.connector.CoyoteAdapter** service. What do these mean?

A: Null pointer exceptions for the **org.apache.catalina.connector.CoyoteAdapter** service are returned when the JBoss ON 3.1.2 server is first installed. These errors are harmless and can be ignored. Installation will complete successfully, and both the server and the GUI will start and run properly.

Q: I upgraded to 3.1.2, and there are null pointer exceptions (**javax.management.InstanceNotFoundException**) in my error logs about the transport service not being registered.

A: When starting a server while agents are running, the server may log transport servlet errors in the logs:

```
[org.rhq.enterprise.server.resource.metadata.ResourceMetadataManagerBean
]
Persisting new ResourceType [ModeShapePlugin:Sequencing
Service(id=0)]...
2011-01-10 16:45:38,571 ERROR [org.apache.catalina.core.ContainerBase]
Servlet.service() for servlet ServerInvokerServlet threw exception
java.lang.reflect.UndeclaredThrowableException
  at $Proxy424.processRequest(Unknown Source)
  at
org.jboss.remoting.transport.servlet.web.ServerInvokerServlet.processReq
```



```

uest(ServerInvokerServlet.java:128)
  at
org.jboss.remoting.transport.servlet.web.ServerInvokerServlet.doPost(Ser
verInvokerServlet.java:157)
  at javax.servlet.http.HttpServlet.service(HttpServlet.java:710)
  at javax.servlet.http.HttpServlet.service(HttpServlet.java:803)
  ....

```

This is because the remoting (communications or transport) classes are loaded early in the startup sequence, before the server is completely started. This causes some communications interruptions until the server is completely started. These errors can be ignored.

Q: I'm seeing error messages when I install (or upgrade) my server. What do they mean?

A: During the upgrade, you may see error messages in the console similar to the following:

```

ERROR [ClientCommandSenderTask] {ClientCommandSenderTask.send-
failed}Failed to send
command [Command: type=[remotepojo]; cmd-in-response=[false]; config=
[{rhq.timeout=1000,
rhq.send-throttle=true}]; params=
[{targetInterfaceName=org.rhq.enterprise.communications.Ping,
invocation=NameBasedInvocation[ping]}]]. Cause:
org.jboss.remoting.CannotConnectException:[.....]

```

These can be ignored.

Q: I upgraded to JBoss ON 3.0.1. However, I see null pointer exceptions in my server logs and the plug-ins still show version 3.0.0. (The 'Server Name' field was changed during upgrade.)

A: During upgrade, the process to register the upgraded server plug-ins fails with a null pointer exception. For example:

```

2012-03-08 20:33:34,523 ERROR
[org.rhq.enterprise.server.core.plugin.ServerPluginScanner] Failed to
register server plugin file [/home/hudson/jon-server-
3.0.1.GA/jbossas/server/default/deploy/rhq.ear/rhq-serverplugins/rhq-
serverplugin-ant-bundle-4.2.0.JON.3.0.1.GA.jar]
java.lang.NullPointerException
  at
org.rhq.enterprise.server.core.plugin.ServerPluginScanner.registerServer
Plugin(ServerPluginScanner.java:212)
  ...

```

This error only occurs if a different server name was entered in the configuration page when the server was upgraded. Changing the **Server Name** field is **not supported** for upgrades.

Q: The error log is showing ErrorCode=[2289]. Why?

A: With Oracle, selecting the overwrite tables option when there is nothing to overwrite causes an error message with **ErrorCode=[2289]**. This can be ignored.

7.2. Connection Issues

Q: I can't connect to my server after installation.

A: If the installer is not bound to 0.0.0.0 when setting up a server, then it does not set all of the required connection properties. Specifically, the installer does not set the `java.rmi.server.hostname` parameter to the real value, and it uses the default of 0.0.0.0. This parameter must be set to the real IP address of the server by manually editing the `rhq-server.properties` file. Restart the server after editing the properties file to load the changes.

7.3. UI Problems

Q: I upgraded my server, but when I try to connect to the installer page to configure it, it keeps trying to redirect me to the (old) `coregui/` module. How do I get to the installer?

A: Some browsers cache the previous `coregui/` module and attempt to redirect you there automatically after upgrading, even though the upgraded `coregui/` module has not yet been loaded.

Simply navigate directly to the installer page:

```
http://server.example.com:7080/installer/start.jsf
```

Chapter 8. Installing and Upgrading the Agent from the JAR File

JAR files to install the JBoss Operations Network agent on Red Hat Enterprise Linux, Windows, Solaris, AIX, and other *nix distributions are available as a download from the JBoss ON server.

8.1. Before Installing the Agent

8.1.1. Setting up the JRE for the JBoss ON Agent

The JBoss ON agent requires either Java 6 or Java 7 JRE.

1. Download and install the appropriate version of the JRE, if necessary.
2. Set the **JAVA_HOME** environment variable to the installation directory.
 - a. Open the **.bashrc** for the system user that will run JBoss ON. For example:

```
vim /home/jon/.bashrc
```

- b. Add a line to set the **JAVA_HOME** environment variable to the specific JRE directory. For example:

```
export JAVA_HOME=/usr/lib/jvm/jre-1.6.0-openjdk/bin/java/
```

3. Set the system to use the correct version of the JRE using the system **alternatives** command. The selected version has the ***+** symbols by it.

```
/usr/sbin/alternatives --config java
```

```
There are 5 programs which provide 'java'.
```

Selection	Command
1	/usr/lib/jvm/jre-1.5.0-sun/bin/java
2	/usr/lib/jvm/jre-1.4.2-gcj/bin/java
3	/usr/lib/jvm/jre-1.6.0-sun/bin/java
*+ 4	/usr/lib/jvm/jre-1.6.0-openjdk/bin/java
5	/usr/lib/jvm/jre-1.6.0-bea/bin/java

```
Enter to keep the current selection[+], or type selection number:
```

8.1.2. Picking the Agent System User

Before installing the agent, plan what system user and group to use to run the agent. The given user can have an impact on how resources are discovered and how they should be configured for management.

The common types of servers which JBoss ON manages are:

- ✦ JBoss EAP servers
- ✦ PostgreSQL databases
- ✦ Tomcat servers

- » Apache servers
- » Generic JVMs

For the agent to be able to discover a resource requires, at a minimum, that the agent have read access to that resource's configuration. Some resource types may require more than just read access. For JBoss EAP resources, for example, the agent must have read permissions to the **run.jar** file, plus execute and search permissions for every directory in the path to the **run.jar** file.

Read access or even root access may not be sufficient for some resource types. Tomcat servers can only be discovered if the JBoss ON agent and the Tomcat server are running as the same user. The same is true for JVMs and JMX servers with the attach API.

The system user which the agent runs as impacts several common agent tasks:

- » Discovery
- » Deploying applications
- » Executing scripts
- » Running start, stop, and restart operations
- » Creating child resources through the JBoss ON UI
- » Viewing and editing resource configuration

There is a general assumption that the agent runs as the same user as the managed resources, and this is the easiest option to manage resources effectively.



Important

While it is possible to run the JBoss ON agent as the root user, and in some limited contexts that may be the simple choice, consider the security implications of running a service as root before setting up the agent.

Generally, services should be run with the *least amount of access required to perform their operations*. This is because if a service is ever compromised, its access permissions can be exploited by an attacker.

The Red Hat Enterprise Linux *Security Guide* contains a section on [security guidelines and links to security planning documents](#). There are similar recommendations in the Windows documentation.

When the JBoss ON agent is installed from the agent installer JAR file, the system user and group who own the agent installation files is the same user who installs the JAR. So, a special system user can be created or selected, and then the agent can be installed by that user.

If the agent and the resource are run as different users and the agent needs to perform some actions as the resource user, there are a few configuration options, depending on what needs to be done:

- » Configure scripts or operations to run using **sudo**. For long-running operations, such as starting a service or a process, the user which executes the script should be the same as the resource user because that user will have the proper authorization and permissions.
- » Set start script environment variables to use the resource's principal and credentials, if available.

- ✦ *For JVM or JMX servers.* Select the connection configuration based on the user settings. For different users, use JMX remoting. For the same user, use either JMX remoting or the attach API.

Table 8.1. Cheat Sheet for Agent and Resource Users

Resource	User Information
PostgreSQL	No effect for monitoring and discovery. The agent user must have read/write permissions to the PostgreSQL configuration file for configuration viewing and editing.
Apache	No effect for monitoring and discovery. The agent user must have read/write permissions to the Apache configuration file for configuration viewing and editing.
Tomcat JMX server or JVM	Must use the same user or can't be discovered Different users are fine when using JMX remoting; cannot be discovered with different users and the attach API
JBoss AS/EAP	Different users are all right, but requires read permissions on run.jar and execute and search permission on all ancestor directories for run.jar

8.2. Installing the Agent from JAR File

1. Point your browser to the download URL on the server. For example:

```
http://server.example.com:7080/agentupdate/download
```

Save the agent binary update **.jar** in a directory where you want to install the agent. The file you save should have a **.jar** extension.

2. Copy the agent update binary **.jar** you downloaded from the JBoss ON server to the directory.
3. Install the JAR:

```
java -jar downloaded_agent_jar_file.jar --install
```

This will tell the agent update binary to extract the JBoss ON agent distribution and install a fresh copy of it in the **rhq-agent** subdirectory.



Important

Do not install the agent in a directory with spaces in the name, such as **C:\Program Files**.

Installing the agent in a directory with spaces in the pathname can cause problems for the agent establishing a connection with certain types of resources, including some JBoss services.

4. Start the agent to launch the setup process.

```
agentRoot/rhq-agent/bin/rhq-agent.sh
```



Note

It is possible to skip the setup wizard by submitting the configuration all at once. [Section 8.3, “Using an Answer File for the Agent Installation”](#) has the details for setting up an file that can pass the configuration directly to the agent installer.

- As prompted, supply the information to configure the agent and the server connection.

```
[Agent Name] agentdomain.example.com
[Agent Hostname or IP Address] agentdomain.example.com
[Agent Port] 16163
[JON Server Hostname or IP Address] server.example.com
[JON Server Port] 7080
native enable
```

- ✦ The agent name must be unique among all agents in the JBoss ON deployment. By default, the name is the fully-qualified domain name of the host machine.
- ✦ The port is the one that the agent uses to listen for incoming messages from the server. This is ***rhq.agent.server.bind-port*** in the configuration file, if the default value isn't used.
- ✦ The server hostname and port are used by the agent to connect to a server to register itself with the JBoss ON system. This is not necessarily the primary server that the agent will use after registration. In the configuration file, these are ***rhq.agent.server.bind-address*** and ***rhq.agent.server.bind-port***

The full list of parameters, including advanced setup options, are listed in [Table 8.2, “All Options Available During Advanced Setup”](#).

- Configure the agent as a background service, as in [Section 8.4, “Running the JBoss ON Agent as a Service”](#).

Once the agent is configured, it persists its configuration in the Java Preferences backing store. Once this happens, **agent-configuration.xml** is no longer needed or used. Editing **agent-configuration.xml** will no longer have any effect on the agent, even after restarting the agent. To pick up changes to the **agent-configuration.xml** file, the agent must be restarted with the **--cleanconfig** command line option or the configuration must be reloaded with the **config --import** agent prompt command.



Important

If the agent fails to register with the server and seems to hang after outputting the message *The agent does not have plugins - it will now wait for them to be downloaded...* or otherwise does not work properly after configuring it, please check the agent log file for error messages (`agentInstallDir/logs/agent.log`).

Typically, problems occur when the agent binds to an IP address or hostname that is not resolvable or accessible by the JBoss ON servers.

Similarly, make sure all of the JBoss ON servers' public endpoint addresses are resolvable by the JBoss ON agent. The JBoss ON server that is entered for an agent to register with may not be the same one that the agent uses as its primary server; it depends on the high availability configuration. If the agent cannot contact its server, then it fails to start properly.

Table 8.2. All Options Available During Advanced Setup

Setup Option	Description	Normal or Advanced Setup
Agent Hostname or IP Address	The address that the binds to to listen for messages from the server. This is usually the same as the address that the JBoss ON server uses to connect to the agent; if the addresses are different because of the network environment, then transport parameters must be set to resolve the address.	Normal
Agent Port	The port number that the agent listens on. As with the IP address, this is usually the same as the port configured for the servers to use to connect to agents, but if these ports are different because of the network environment, then transport parameters must be set to resolve the port.	Normal
Agent Transport Protocol	Sets the protocol that the agent expects to use to receive incoming messages from the server. This is usually socket or sslsocket.	Advanced
Agent Transport Parameters	Sets transport parameters to append to the end of the locator (URL-style address) used by the remoting framework for agent-server connections.	Advanced

Setup Option	Description	Normal or Advanced Setup
RHQ Server Hostname or IP Address	Gives the IP address or hostname of the primary server that the agent communicates with. This information <i>must</i> be the same as the hostname or IP address that is configured in the JBoss ON server configuration.	Normal
RHQ Server Port	Gives the port number of the primary server that the agent communicates with. This information <i>must</i> be the same as the port number that is configured in the JBoss ON server configuration.	Normal
RHQ Server Transport Protocol	Sets the transport protocol that the agent uses for outgoing messages to the JBoss ON server. This information <i>must</i> be the same as the transport method that the server is configured to expect in its configuration preferences.	Advanced
RHQ Server Transport Parameters	Gives additional transport parameters that are to be used when the agent connects to the primary JBoss ON server. Since this is used to connect to the server, these parameters <i>must</i> be the same as the transport parameters set in the JBoss ON server configuration. These settings are especially important if the JBoss ON agent needs to connect to a different host or port than what the JBoss ON server actually binds to.	Advanced
Command Send Timeout	Sets the timeout period, in milliseconds, that the agent waits before aborting an attempt to send a command. This is essentially the amount of time that the JBoss ON server has in order to process commands and return its results. Both this timeout and any timeout period set in the transport parameters are enforced, so these should be the same (if server transport parameters are given). A setting of zero (0) means that there is no timeout period.	Advanced

Setup Option	Description	Normal or Advanced Setup
Command Send Retry Interval	Sets the minimum amount of time, in milliseconds, the agent waits before trying to resend a guaranteed command that previously failed.	Advanced
Command Send Max Retries	Sets the maximum number of times that an agent attempts to resend a guaranteed delivery command that failed. Any command that fails with the message <i>cannot connect</i> will be retried infinitely, until the send succeeds. For any other error, the agent retries the send until it hits this limit and drops the command.	Advanced
Maximum Commands To Concurrently Send	Sets the maximum number of commands the agent can send to the server at any one time. If the clientMaxPoolSize setting is passed in the JBoss ON server URI transport parameters, its value must be the same as this value.	Advanced

8.3. Using an Answer File for the Agent Installation

It is possible to skip the setup wizard by submitting the configuration all at once. There are two ways to do this: either the agent's configuration file (**agent-configuration.xml**) can be edited directly or a simple text file can be passed to the agent installer.

The simplest method is to create a text file that is passed with the start script when the agent is first started that has all of the prompts answered. This mirrors the prompt questions, in order:

```
agent_name
agent_hostname
agent_port
server_hostname
server_port
native_api_support
```

The final file would then resemble this:

```
agent.example.com
agent.example.com
16163
server.example.com
7080
native --enable
```

To pass the text file, use the **--input** option with the **rhq-agent.sh|bat** script:

```
agentRoot/rhq-agent/bin/rhq-agent.sh --input myAnswers.txt --nonative
```

The alternative is to edit the **agent-configuration.xml** directly. This offers some more flexibility because other settings (like SSL configuration) can be configured and passed with the initial setup. At a minimum, the entry keys listed in [Table 8.3, “Configuration File Keys for Agent Setup”](#) have to be set in the file.

After the configuration file is ready, then start the agent and pass the configuration file:

```
agentRoot/rhq-agent/bin/rhq-agent.sh --config agentRoot/rhq-agent/conf/agent-configuration.xml
```

Table 8.3. Configuration File Keys for Agent Setup

Installer Prompt Text	Key Name	Description
	rhq.agent.configuration-setup-flag	Tells the installer that the agent configuration is already in the configuration file. This must be set to true for the installer to load the configuration file.
[Agent Name]	rhq.agent.name	Gives a unique name to identify the agent to the server.
[Agent Hostname or IP Address]	rhq.communications.connector.bind-address	Gives the hostname or IP address that the server will use to connect to the agent. This <entry> line may need to be uncommented before it is set.
[Agent Port]	rhq.communications.connector.bind-port	Gives the port for the server to use to communicate with the agent. The default (16163) can be used in most cases.
[JON Server Hostname or IP Address]	rhq.agent.server.bind-address	Gives the hostname or IP address that the agent will use to connect to the server to register itself. If this is a hostname, it must be resolvable by the agent.
[JON Server Port]	rhq.agent.server.bind-port	Gives the port for the agent to use to communicate with the server. The default (7080) can be used, assuming the server was configured with the default values.
native	rhq.agent.disable-native-system	Enables the JNI libraries used by the agent. This enables the agent to discover and manage some types of resources using the system native libraries.

8.4. Running the JBoss ON Agent as a Service

In almost all production environments, the agent should be started as a background daemon process. On Windows, this runs as a service. On Linux and Unix systems, the agent starts at boot time from **init.d**.

More detail on editing the agent server wrapper script and managing the agent daemon is covered in the [Configuring JBoss ON Servers and Agents](#) guide.

8.4.1. Running the Agent as a Windows Service

The default Java service wrapper included with JBoss ON requires a 32-bit JVM, so the Java preference set for the agent must be a 32-bit JRE. **The JBoss ON agent must use a 32-bit JVM even on 64-bit systems.**

Running the agent with a 32-bit JVM does not in any way affect how JBoss ON manages other resources which may run with a 64-bit JVM. JBoss ON can still manage those resources and those resources can still use the 64-bit Java libraries for their own processes.

1. Make sure the agent is fully set up. The agent does not prompt for the configuration when it is started as a service.
2. Edit the **rhq-agent-wrapper.bat** script and set the environment variable to define the system user as whom the init script will run. There are two options:
 - ✦ **RHQ_AGENT_RUN_AS** explicitly sets the user account name. This must match the format of a Windows user account name, *DOMAIN\username*.
 - ✦ **RHQ_AGENT_RUN_AS_ME** forces the agent to run as whoever the current user is; this uses the format *.\%USERNAME%*. If both environment variables are defined, this variable overrides **RHQ_AGENT_RUN_AS**.



Note

Before setting **RHQ_AGENT_RUN_AS_ME** or **RHQ_AGENT_RUN_AS**, make sure that the given user actually has permission to start services. If necessary, assign the user the appropriate rights. Assigning rights is covered in the Windows documentation.

If neither variable is set, the agent init script runs as the System user.

Other available environment variables are listed and defined in the comments in the **rhq-agent-wrapper.bat** script.

3. Run the **rhq-agent-wrapper.bat** script to install the init script as a service. Use the **install** command to install the init script.
4. When prompted, fill in the password for the system user as whom the service will run.

8.4.2. Running the Agent as a Daemon or init.d Service

1. Make sure the agent is fully set up. The agent does not prompt for the configuration when it is started as a service.
2. Open the **rhq-agent-env.sh** file.
3. Uncomment and configure the required environment variables for the agent's **bin** directory, the JDK directory, and the PID directory (which must be writable by the agent user).

```
RHQ_AGENT_HOME=agentRoot/rhq-agent/
export RHQ_AGENT_JAVA_HOME=/usr
PIDFILEDIR=/var/run
```

**Note**

When setting the **PIDFILEDIR** on Red Hat Enterprise Linux, edit the *pidfile* setting in the **rhq-agent-wrapper.sh** script file. The wrapper script value is used by **chkconfig**.

4. Set any of the optional environment variables.
 - ✦ **RHQ_AGENT_DEBUG** enables debug logging.
 - ✦ **RHQ_AGENT_JAVA_EXE_FILE_PATH** specifies a Java executable.
 - ✦ **RHQ_AGENT_JAVA_OPTS** passes settings to the agent JVM.
 - ✦ **RHQ_AGENT_ADDITIONAL_JAVA_OPTS** passes additional Java options to the JVM.
5. Log into the system as root.

**Important**

The rest of this procedure describes how to configure the agent init script as a service on Red Hat Enterprise Linux. For other Unix systems, follow a similar procedure that corresponds to the specific platform.

6. Make sure the wrapper script is executable.

```
[root@server rhq-agent]# chmod a+x agentRoot/rhq-agent/bin/rhq-agent-wrapper.sh
```

7. Symlink the **rhq-agent-wrapper.sh** file to **/etc/init.d/**. For example:

```
[root@server rhq-agent]# ln -s agentRoot/rhq-agent/bin/rhq-agent-wrapper.sh /etc/init.d/rhq-agent-wrapper.sh
```

**Important**

On Solaris, symlinking the agent script file requires invoking **readlink** in **rhq-agent-wrapper.sh**. **readlink** is not supplied by default in some Solaris installations. Solaris users must download **readlink** from a source such as Sunfreeware.

8. Register **rhq-agent-wrapper.sh** with **chkconfig**.

```
[root@server rhq-agent] # /sbin/chkconfig --add rhq-agent-wrapper.sh
```

9. Enable the agent service to run at boot time and have it stop gracefully at when the system shuts down.

```
[root@server rhq-agent] # /sbin/chkconfig rhq-agent-wrapper.sh on
```

If the agent service should not be started when the system boots, turn the script off in **chkconfig**:

```
[root@server rhq-agent] # /sbin/chkconfig rhq-agent-wrapper.sh off
```

8.5. Changing Agent Connection Configuration

There are two parts to the agent configuration:

- ✦ The agent connection properties, which define the agent instance and how it communicates to the server
- ✦ The agent JVM properties, which manage agent performance and options

The agent connection properties are defined when the agent is installed. This includes information like the server for it to connect to, its port number, and whether to use SSL connections.

That agent connection configuration is initially read from **agent-configuration.xml** and overlaid with the values entered at the setup prompts at start up. After the agent is initially configured, the agent persists that configuration and never refers to the **agent-configuration.xml** again. For that information to be changed, the agent connection information has to be wiped out and reset.

To change the connection configuration, one option is to use the **--cleanconfig** option and run through the setup wizard again.

```
agentRoot/rhq-agent/bin/rhq-agent.sh --cleanconfig
```

Most JVM and optional settings (the persisted configuration) are made in the **rhq-agent-env.sh** file, which is loaded every time the agent starts, or using agent prompt commands like **setconfig**. This is described in [Configuring JBoss ON Servers and Agents](#).

8.6. Installing Multiple Agents with a Shared Directory or Account

Multiple agents, running on multiple systems, can share the same system user accounts. If the same user is used for a JBoss ON agent on different systems *and* those system users all use the same shared home directory, then they all share the same agent configuration location and preference node by default. Because of the way the agent uses Java preferences, this requires special agent configuration to prevent the agents from overwriting each other's preferences.

A similar situation can occur on Windows systems if the same domain user is used for the JBoss ON agent. In that case, the Java preferences are stored in a registry key which is used by the domain user and is loaded into the local user's profile. If there are multiple agents using the same domain user, then they will overwrite each other's registry keys.

All of the agent configuration, after setup, is stored in a Java preferences node. With the default configuration, the node name is **default**, and the node location is **agentUserHomeDir/.java/.userPrefs/rhq-agent/default**.

If multiple agents are installed using the same file share, then all of them attempt to use the same default node and location.

When multiple agents attempt to use the same Java preferences node, each new agent overwrites the previous agent's configuration as it is set up. This means that only the newest agent's configuration is saved, so only the newest agent can be started. Starting any of the previous agents fails because they cannot find their own configuration.

The preferences node is uniquely identified by two settings:

- ✦ Its name, which is defined as an agent configuration setting
- ✦ Its location, which is itself a Java option

To run multiple agents with the same home directory, the preferences node has to be uniquely identified for each agent. There are a couple of different ways to do that:

- ✦ Editing the agent configuration files directly
- ✦ Setting an explicit Java option

8.6.1. Editing the Configuration Files

When the agent is first set up, the name of the agent preferences node is set in the **agent-configuration.xml** file and is loaded from there. The node location is derived from the node name setting.

1. Edit the **agent-configuration.xml** file to use the new node name:

```
[rhquser@server ~]$ vim agentRoot/rhq-agent/conf/agent-configuration.xml

<node name="agent01-node">
```

2. Then, start the agent with the **--config** option to load the edited configuration file and the **--prefs** option to point to the specific node location:

```
[rhquser@server ~]$ agentRoot/rhq-agent/bin/rhq-agent.sh --prefs=agent01-node --config=agent-configuration.xml
```



Important

If the custom Java preferences node is specified by editing the **agent-configuration.xml** file, then every time the agent restarts, the node location has to be passed to the agent using the **--prefs** option.

8.6.2. Setting a Java Option

Editing the **agent-configuration.xml** file only sets the node name; the node location still has to be passed every time the agent is started.

By setting a Java option in the **rhq-agent-env.sh** file, the Java preferences node information is set once and then persisted, so you can restart the agent as a service, without having to pass **--prefs** options or edit and reload the configuration.

1. Open the agent prompt. For example, if the agent process is already running, the prompt can be opened by re-running the **rhq-agent.sh** script with the **-n** option.

```
[rhquser@server ~]$ agentRoot/rhq-agent/bin/rhq-agent.sh -n
```

2. Use the **setconfig** command to set the **RHQ_AGENT_ADDITIONAL_JAVA_OPTS** value with the preference node. For example:

```
> setconfig RHQ_AGENT_ADDITIONAL_JAVA_OPTS="-
Djava.util.prefs.userRoot=agentUserHomeDir/.java/.userPrefs/rhq-
agent/agent01-node"
```

The preference node can be in the user preferences directory with a different name, such as **agent01-node**, or it can be in an entirely different location, such as **/etc/agent-preferences**, which is not a shared or filesystem-mounted location.

- Restart the agent process to load the new configuration. For example, if the agent is running as a service:

```
[rhquser@server ~]$ service rhq-agent-wrapper.sh stop

[rhquser@server ~]$ service rhq-agent-wrapper.sh start
```



Note

It is also possible to stop the agent, edit the **rhq-agent-env.sh** file directly, and then restart the agent.



Important

Do not set the Java option within the rhq-agent runtime directory because this file is overwritten during JBoss ON agent updates. The default location for the rhq-agent runtime is **\$USERHOME/.java/.userPrefs/rhq-agent/default**.

8.7. About Agent Automatic Updates

Even for automatic upgrades for the agent, certain preparation has to be made to the JBoss ON agent to make sure that the upgrade process goes smoothly and the agent restarts successfully.

8.7.1. The Process When an Agent Autoupgrades

By default, both JBoss ON servers and agents are configured to upgrade agents automatically. As soon as the JBoss ON server is upgraded, then the agents will be upgraded.



Note

The agent should be running in the background to upgrade properly, as in [Section 8.4, "Running the JBoss ON Agent as a Service"](#).

The automatic upgrade process is part of the normal agent tasks of checking the server for updates:

- The updated server puts the updated agent packages in a directory accessible to the agent.
- The server notifies the agent that the agent needs to update as soon as the server detects that the agent is running an older version.

3. As the agent prepares to update, it begins shutting down its other process. This can take several minutes, as it gracefully shuts down each thread.
4. The agent downloads the new binaries from the server.
5. The agent spawns a new Java process.
6. The Java process backs up the old agent configuration and applies the update.
7. The Java process then restarts the agent and kills itself.

It is possible to instruct an agent to initiate an update or to check if updates are available using the **update** through the agent command line:

```
agentRoot/rhq-agent/bin/rhq-agent . sh
> update
```

8.7.2. Configuring Agent Preferences

Agent preferences for settings like the Java home directory can be set in environment variables for normal use. However, the environment variables set in the shell are lost when the upgrade process stops and restarts the agent, and that means that the agent may not automatically restart after upgrade. Custom settings for the agent, such as **RHQ_AGENT_HOME** or **RHQ_AGENT_ADDITIONAL_JAVA_OPTS**, should be added to the **rhq-agent-env.sh** file. This file is preserved during upgrade so all of the settings are carried over.



Warning

Do not edit any of the scripts used to launch the JBoss ON agent. There are four files which should never be modified:

- ❖ rhq-agent.sh
- ❖ rhq-agent-wrapper.sh
- ❖ rhq-agent.bat
- ❖ rhq-agent-wrapper.bat

Any changes to the launcher scripts are overwritten during the automatic update. Changes to the environment files (such as **rhq-agent-env.sh**) are preserved during the update.

8.7.3. Upgrading Custom log4j Settings

Any edits to the Java settings in the **rhq-agent-env.sh** file are preserved between upgrades. However, any changes to the log settings (**log4j.xml**) and other files are lost between upgrades.

8.7.4. Configuring Keystores and Truststores

If the original JBoss ON agent was configured to run over SSL using custom keystores and truststores, then make sure that those stores are configured so that the upgraded agent can still access them:

1. The keystore files must have the word *keystore* in their filenames. For example, **my-agent-keystore.dat**.

2. The truststore files must have the word *truststore* in their filenames. For example, **my-agent-truststore.dat**.
3. Both the keystore and truststore files *must* be located in the agent's **agentRoot/rhq-agent/data** directory.

As long as the SSL files are properly set up, then they will be copied over into the new agent configuration, and the new agent will automatically run in SSL.

8.7.5. Setting Writer Permissions on the Agent Home Directory

The upgrade process will write new files to the agent's current installation directory, so the agent's system user must have write permissions to that directory.

If the agent's home directory is **/opt/rhq-agent**, then the agent has to be able to write to the **/opt/rhq-agent** directory. If necessary, reset the permissions on the agent home directory. For example:

```
chown agent_user /opt/rhq-agent
```

8.7.6. Starting the Agent as a Background Service

[Section 8.4, "Running the JBoss ON Agent as a Service"](#) describes how to configure the agent to run as a background service. On Windows, this runs as a service. On Linux and Unix systems, the agent starts at boot time from **init.d**.

The auto-upgrade process assumes that the agent is running in the background. If the agent is not running as a background daemon, when the agent auto-updates itself, the old agent process running in the console is shutdown, and the new agent is restarted as a background service if possible. On Windows, if the agent is not installed as a service, the agent is restarted in a console window.

8.8. Manually Upgrading the JBoss ON Agent

To ensure compatibility with the JBoss ON server, each agent must be upgraded to the same version of JBoss ON as the server.

Agents have the ability to auto-update themselves. So, under most conditions, it isn't necessary to manually upgrade agents. However, if the auto-update fails for some reason or you disabled agent auto-update, then the agent can be upgraded manually.



Note

All agents must be upgraded at the same time. Having agents of different versions is not supported.

1. Shut down the JBoss ON agent.
2. *Windows only*.. If the agent is running as a Windows service, uninstall the Windows service:

```
cd old-agent-install-dir/bin
./rhq-agent-wrapper.bat remove
```

3. Upgrade the JBoss ON server, as in [Section 3.8.1, "Upgrading the JBoss ON Server"](#). The JBoss ON server must be upgraded before any agents are upgraded.

4. Restart the upgraded JBoss ON servers if they are not yet started.
5. Download the agent update binary from the server.
6. Copy the agent update binary JAR file into the parent directory where the agent is installed. For example:

```
cp agent-update-binary.jar /opt/rhq-agent
```

7. Extract the new JBoss ON agent from the agent update binary by running the following command:

```
java -jar agent-update-binary.jar --
update=agent_installation_directory
```

This will tell the agent update binary to extract the JBoss ON agent distribution and update the current agent that is found in **rhq-agent** subdirectory. At this point, the upgraded JBoss ON agent is located in the original **rhq-agent** directory. The old agent has been backed up to the **rhq-agent-old** directory. Any upgrade errors are written to the agent's log files.

8. Finally, start the JBoss ON agent.

8.9. Reinstalling the Agent (JAR)

An agent can be re-installed, with completely fresh configuration. There are three points of configuration for the agent: the agent's (local) persisted configuration, the agent inventory (and associated resource data), and the platform entry in the server inventory. Both the configuration on the local machine and the agent and resource configuration on the JBoss ON server need to be cleared for the agent to be reinstalled successfully:

- ✦ The agent's persisted Java configuration should be purged.
- ✦ The agent's inventory should be purged, along with any resource history and configuration.
- ✦ The agent (via the platform entry) must be removed from the JON inventory.



Important

If the agent was configured, but the platform was never imported into the inventory, then *you must import the platform from the discovery queue first*, and then delete the platform. The discovery queue is a halfway point in the inventory. Even if the agent is removed, the platform could still linger in the discovery queue as a ghost entry.

To reinstall the agent:

1. Make sure that the original agent instance is properly removed.
 - a. Stop the agent process.
 - b. Remove the platform entry from the JBoss ON server inventory.
2. Restart the agent with the **--fullcleanconfig** option. This registers the agent with a new security token and fresh configuration settings.

```
agentRoot/rhq-agent/bin/rhq-agent.sh --fullcleanconfig
```

Chapter 9. Installing the Agent from RPM

An RPM package is available for Red Hat Enterprise Linux systems to install the agent and, optionally, the agent service.

9.1. About Agent RPMs

Installing an agent through an RPM offers a simpler and standardized installation process, which makes it possible to install agents on resources in cloud and PaaS environments or when kickstarting machines in data centers.

The JBoss ON agent RPMs can also make it easier to manage the agent more like other Linux system applications because the agent is automatically configured to function as a system service:

- ✦ System user and group settings with appropriate permissions already set
- ✦ System services to start, stop, and restart the agent
- ✦ System service to change the agent's configuration
- ✦ Upgrades using system tools

When installing the agent from the JAR file, there are several factors in the install environment that define the agent configuration, such as the installation directory and the location of the Java preferences store. Meaning, the installation directory is determined by where the JAR is unpacked. The agent user and the Java preferences location are both defined by what system user performs the installation.

Many of these settings are defined as part of the RPM setup, so the environment has minimal impact on the resulting agent configuration. This section describes some of the differences between RPM and JAR installations and some of the default configuration set by the RPM process.

9.1.1. Differences Between JAR and RPM Installations

The most notable difference is that the RPM defines the home directory and locations of important files, regardless of the location where the `rpm` is run or the user account (root) who initiated it.

Table 9.1. Some Differences Between JAR and RPM Installations

Configuration Area	JAR Value	RPM Value
Agent user	Set to the system user who installs it	jboss-on-agent user, jboss-on-group
Agent service	Not set	jon-agent
Environment variables	<code>installDir/bin/rhq-agent-env.sh</code>	<ul style="list-style-type: none"> ✦ <code>/etc/init.d/jon-agent</code> (init script) for <code>ADDITIONAL_JAVA_OPTS</code> ✦ <code>/usr/share/jboss-on-3.1.2.0.GA1/agent/bin/rhq-agent-env.sh</code> for <code>JAVA_OPTS</code>
Home directory location	Wherever the JAR is installed	<code>/usr/share/jboss-on-3.1.2.0.GA/agent/</code>
agent-configuration.xml location	In the <code>conf/</code> directory where the JAR is installed	<code>/etc/jboss-on/agent/ [a]</code>

Configuration Area	JAR Value	RPM Value
Java preferences location	~/java/default (system user Java preferences)	/var/lib/jboss-on/agent/prefs/.java/.userPrefs/rhq-agent/default/
Data directory location	<i>agentInstallDir</i> /data	/var/lib/jboss-on/agent/data/
Log directory location	In the logs/ directory where the JAR is installed	/var/log/jboss-on/agent/ [b]
Autoupgrade	Enabled	Disabled
	[a] symlinked to /usr/share/jboss-on-3.1.2.0.GA/agent/conf	
	[b] symlinked to /usr/share/jboss-on-3.1.2.0.GA/agent/logs	

9.1.2. The JBoss ON User

Before installing an agent JAR, one of the most critical decisions is selecting the system user as which the JBoss ON agent will run ([Section 8.1.2, “Picking the Agent System User”](#)). This has security implications for the agent process on the system, and it also affects how the user interacts with local server and application resources — which each have their own system users and permissions.

The agent RPM automatically creates a new system user with the appropriate system configuration to address security issues like directory access.



Important

The agent user still has to interact with resources. The appropriate group permissions, SELinux contexts, and other resource configuration can still affect how the agent can discover and manage a resource. [Section 8.1.2, “Picking the Agent System User”](#) outlines these considerations; if necessary, alter the system configuration to allow the agent the appropriate level of access to the resource.



Note

The agent RPM creates the **jon-agent** user and the **jbosson** group when it is installed. **The user and group are not removed if the RPM is uninstalled.**

Table 9.2. Agent User Configuration

Property	Value
Username	jbosson-agent
Group name	jbosson
User ID (UID)	400
Group ID (GID)	400
User properties	NOSHELL
Init script owner	root
Init script user	jon-agent [a]
	[a] This can be edited to be any system user.

9.1.3. Service Tools and Init Script

Part of the RPM setup includes configuring the JBoss ON agent as a system service. An init script is installed for the agent at `/etc/init.d/jon-agent`. `chkconfig` is configured so that the agent starts when the system starts and runs as a daemon.

The init script includes all of the normal service management commands, as well as specific commands to manage the agent itself:

- **start**
- **stop**
- **restart**
- **status**
- **kill**, which forces the agent process to stop
- **config**, which runs through the agent configuration wizard again and refreshes the agent configuration with new settings

The **start**, **stop**, **restart**, and **status** commands are available when the agent is manually configured to run as a service, as described in [Section 8.4, “Running the JBoss ON Agent as a Service”](#). However, the **kill** and **config** commands are only available with the init script provided with the agent RPM.

The agent init script, `/etc/init.d/jon-agent`, sets the environment variables that are set in the `rhq-agent-env.sh` file with a JAR installation. This init script defines the agent system user and group, the log and data directory locations, and Java options. Editing the init script can, for example, allow the agent to run as a different user or to start with different JVM settings.



Important

When the agent is installed from the RPM, the only supported way to edit the agent configuration is by running the **config** command or by editing the init script. Editing the `rhq-agent-env.sh` file or other configuration files directly is not supported.

9.1.4. Update Differences

When an agent is installed through a JAR, there is a key set in the `agent-configuration.xml` file that tells the agent to check for upgrades. The agent then polls the server, and if the JBoss ON server version is newer than the agent version, the agent requests updated binaries from the server.

The agent RPMs use an entirely different installation path than the agent JAR files, and an agent installed as an RPM relies on the local system tools to manage its packages. The upgrade flag, then, in the `agent-configuration.xml` file is turned off, to disable attempts at an autoupgrade and to allow the local system to manage the agent packages.

```
<entry key="rhq.agent.agent-update.enabled" value="false" />
```

With autoupdates disabled, the agent must be upgraded manually whenever the JBoss ON server is upgraded, to ensure that its version remains in sync with the JBoss ON server version.

9.1.5. Available Channels

Table 9.3. Available Channels for the Agent RPM

Product Name	Product Version	Channel Name
JBoss Enterprise Application Server (EAP)	5, x86	jbappplatform-5-i386-server-6-rpm
JBoss Enterprise Application Server (EAP)	5, x86_64	jbappplatform-5-x86_64-server-6-rpm
JBoss Enterprise Application Server (EAP)	6, x86_64	jbappplatform-6-i386-server-6-rpm
JBoss Enterprise Application Server (EAP)	6, x86_64	jbappplatform-6-x86_64-server-6-rpm

9.2. Installing the Agent from RPM

The JBoss ON agent is installed through the **jboss-on-agent** package. This package installs all of the agent files, creates a specific JBoss ON agent system user, and configures the JBoss ON agent as a system service.

1. Configure the **yum** repos to include the JBoss ON channel (as listed in [Table 9.3, “Available Channels for the Agent RPM”](#)).
2. Use **yum** to install the package.

```
[root@server ~]# yum install jboss-on-agent
```

Alternatively, download packages from <http://access.redhat.com> and install using **rpm**.

```
[root@server ~]# rpm -ivh jboss-on-agent-3.1.2.0.GA1.el6.noarch.rpm
```



Note

Installing the RPM requires specific entitlements for the RHN user account for the 3.1.2 release.

This installs the agent in **/usr/share/jboss-on-3.1.2.0.GA1/agent**.

3. Configure the agent by running the **service jon-agent config** command. This runs through the advanced installer to configure the agent.

```
[jsmith@server ~]$ sudo service jon-agent config
RHQ 4.4.0.JON311GA [6910991] (Wed Aug 01 18:43:03 EDT 2012)
** Advanced Setup **
... 8< ...
Agent Name [agent.example.com] : agent1
Agent Hostname or IP Address [!*] :
Agent Port [16163] :
Agent Transport Protocol [socket] :
Agent Transport Parameters
[numAcceptThreads=1&maxPoolSize=303&clientMaxPoolSize=304&socketTimeou
t=60000&enableTcpNoDelay=true&backlog=200] :
RHQ Server Hostname or IP Address [255.255.255.255] :
RHQ Server Port [7080] :
RHQ Server Transport Protocol [servlet] :
```

```
RHQ Server Transport Parameters [/jboss-remoting-servlet-
invoker/ServerInvokerServlet] :
RHQ Server Alias [rhqserver] :
The setup has been completed for the preferences at node [/rhq-
agent/default].
```

The **config** command runs through all of the advanced setup options, which configures three areas of the agent:

- ✦ *The agent connection information*, used to register the agent to the server
 - The agent name
 - The agent port
 - The agent host (by hostname or IP address)
- ✦ *The agent-server communication settings*, which include configuring SSL or secure connections and rules on how frequently the agent communicates with the agent
 - The agent protocol, either socket (regular) or sslsocket (secure)

For sslsocket, the JBoss ON server needs to be configured to accept SSL connections, as in [the SSL chapter of Configuring JBoss ON Servers and Agents](#).

- Any client transport parameters to use to connect to the server

Both the server and the agent use JBoss Remoting for communication. JBoss Remoting allows servers and clients to pass connection settings using a URL-style address. Transport parameters include things like pool sizes, timeout periods, and buffer settings. For the complete list, see the [JBoss Remoting client parameters documentation](#).

- The server protocol which the agent uses to send messages to the server, either servlet (regular) or sslservlet (secure)

The server connection settings configured on the agent *must* match the configuration in the server itself.

- Any server transport parameters to use to receive messages from the agent

Both the server and the agent use JBoss Remoting for communication. JBoss Remoting allows servers and clients to pass connection settings using a URL-style address. Transport parameters for the server relate to the servlet used to receive agent messages.

- ✦ *The JBoss ON server to register with*
 - The server host (by hostname or IP address)
 - The server port
 - The server alias, a short nickname to identify the server instance

4. Start the agent.

```
[jsmith@server ~]$ sudo service jon-agent start
```

9.3. Changing the Agent Configuration After an RPM Install

There are two parts to the agent configuration:

- ✦ The agent connection properties, which define the agent instance and how it communicates to the server
- ✦ The agent JVM properties, which manage agent performance and options

9.3.1. Changing Agent Connection Configuration

The agent connection properties are defined when the agent is installed. This includes information like the server for it to connect to, its port number, and whether to use SSL connections.

That agent connection configuration is initially read from **agent-configuration.xml** and overlaid with the values entered at the setup prompts at start up. After the agent is initially configured, the agent persists that configuration in its Java preferences (**/var/lib/jboss-on/agent/prefs/.java/.userPrefs/rhq-agent/default/**) and never refers to the **agent-configuration.xml** again.

For that information to be changed, the agent connection information has to be wiped out and reset.

To change the agent connection (registration) configuration, use the **config** service command to run through the setup options again. This cleans out the preferences store, re-reads the **agent-configuration.xml** file, and runs through the configuration setup again.



Important

When the JBoss ON agent is installed from an RPM, it is automatically configured as a system service. For data consistency and agent performance, always manage the agent connection configuration using the service tools, rather than attempting to edit the configuration files or JVM startup properties directly.

For example:

```
[jsmith@server ~]$ sudo service jon-agent config
RHQ 4.4.0.JON311GA [6910991] (Wed Aug 01 18:43:03 EDT 2012)
** Advanced Setup **
Agent Name [agent.example.com] : agent1
Agent Hostname or IP Address [!*] :
Agent Port [16163] :
Agent Transport Protocol [socket] :
... 8< ...
```

The **config** service command opens the agent start script and automatically passes a series of options which edit the agent connection configuration:

- ✦ **--cleanconfig**, to wipe the previous configuration settings
- ✦ **--setup** and **--advanced**, which force the agent to run its configuration setup again
- ✦ **--daemon** and **--nostart**, which runs the agent command prompt without starting the agent process and then exits (so that the agent can be started as a service)

So, the **service jon-agent config** command is equivalent to starting the agent with all those options:

```
rhq-agent.sh --cleanconfig --setup --advanced --daemon --nostart
```


9.3.2. Changing Agent JVM and Other Init Configuration

After the agent is configured, optional JVM settings (the persisted configuration) are set in the init script, `/etc/init.d/jon-agent` file, or in the environment script, `rhq-agent-env.sh`. Both files are loaded every time the agent starts; it is recommended to edit the init script, which sets the additional `JAVA_OPTS` values.

For example:

```
RHQ_AGENT_ADDITIONAL_JAVA_OPTS="-Drhq.agent.data-
directory=$RHQ_AGENT_DATA_DIR -Djava.util.prefs.userRoot=$RHQ_AGENT_PREFS_DIR
-Xms64m -Xmx128m -Djava.net.preferIPv4Stack=true"
export RHQ_AGENT_ADDITIONAL_JAVA_OPTS
```

The Java settings can also be edited using agent prompt commands like `setconfig`. This is described in [Configuring JBoss ON Servers and Agents](#).

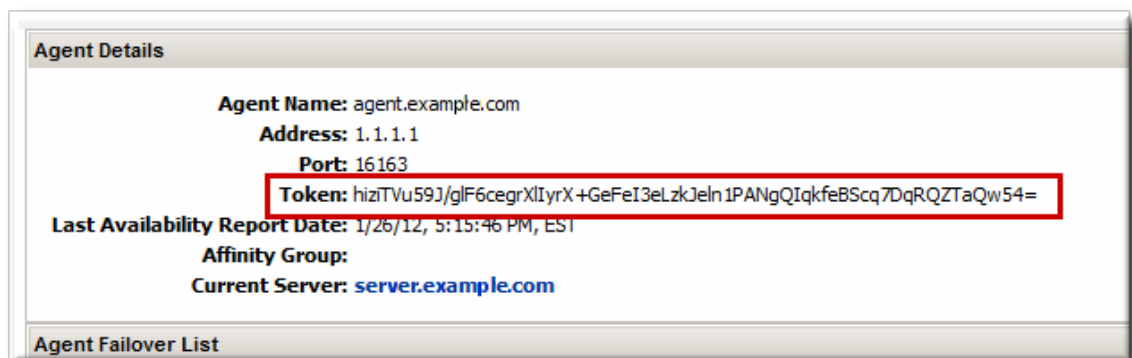
9.4. Migrating from a JAR Installation to an RPM Installation

When an agent is initially installed from the JAR file available with the JBoss ON server downloads ([Chapter 8, Installing and Upgrading the Agent from the JAR File](#)), it is possible to switch the agent to an RPM installation. There are two potential paths. Either the old agent can be scrapped and a new agent installed (which loses all of the original agent configuration and data) or the agent data can be migrated from the original JAR installation to the new RPM installation.

9.4.1. Converting an Agent (Losing Configuration Data)

This method loses all of the original data for the agent, including the persistent store, data directory, and logs. However, by re-using the original security token, the agent can re-register with the server and retrieve its previous inventory and resource histories and configuration, so none of the management information for the platform is lost.

1. Retrieve the security token for the agent.
 - a. Click the **Administration** tab and select the **Agents** link under the **Topology** section on the left.
 - b. Select the agent from the list, and click its name to open its details page.



- c. Copy the security token.
2. Shut down the agent.
3. Remove the JAR installation directory.

4. Install the agent RPM.
5. Edit the **agent-configuration.xml** file and add a line for the original security token for the agent.

```
vim /etc/jboss-on/agent/agent-configuration.xml  
  
<entry key="rhq.agent.security-token" value="abcd1234" />
```

6. Run through the agent configuration installer.

```
[jsmith@server ~]$ sudo service jon-agent config
```

7. Start the agent.

```
[jsmith@server ~]$ sudo service jon-agent start
```

9.4.2. Migrating an Agent to an RPM (Preserving Configuration Data)

It is possible to preserve the JVM and persisted data for the agent, which maintains all of the original configuration data. However, this requires accessing the Java store through a Java preferences browser in the original agent and copying it into the Java store for the new agent, without affecting any other data in the store. There is always a risk when editing Java stores.

1. Shut down the original agent.
2. Install the agent RPM.
3. Copy over the previous configuration directories for the agent. This includes the data directory (which contains operational information like the changesets for drift detection or truststores used for SSL) and the log directory. For example:

```
[root@server ~]# cp -r agentRoot/rhq-agent/data/ /var/lib/jboss-on/agent/data/  
[root@server ~]# cp -r agentRoot/rhq-agent/logs/ /var/log/jboss-on/agent/
```

4. Using a Java preferences editor, export the Java preferences specific to the agent from the original preferences store in **~/ .java/default** (by default).



Note

Be sure to retrieve the security token. The token allows the agent to re-register with the server successfully.

5. Using a Java preferences editor, import the Java preferences for the agent into the new preferences store in **/var/lib/jboss-on/agent/prefs/default** (by default).
6. Run through the agent configuration installer.

```
[jsmith@server ~]$ sudo service jon-agent config
```

7. Start the agent.

```
[jsmith@server ~]$ sudo service jon-agent start
```

9.5. Troubleshooting RPM Installs

Q: I installed my agent successfully, but now it won't connect to the server. Why?

A: The agent has to be started in the foreground to run through its installer. The installer is where the agent is told what JBoss ON server to access. If the agent service is started *without* going through the agent installer, the agent loads its configuration without ever identifying the server to connect to.

The agent configuration has to be edited, as described in [Section 9.3, “Changing the Agent Configuration After an RPM Install”](#), to set the server connection information.

```
[jsmith@server ~]$ sudo service jon-agent config
```

Chapter 10. Uninstalling JBoss ON

Because both the JBoss Operations Network server and agent are extracted JAR files, removing a server or agent ultimately consists of simply deleting those files.

10.1. Uninstalling an Agent

10.1.1. Removing an Agent on Linux (JAR)

1. Remove the platform entry from the JBoss ON server inventory.



Important

If the agent was configured, but the platform was never imported into the inventory, then *you must import the platform from the discovery queue first*, and then delete the platform. The discovery queue is a halfway point in the inventory. Even if the agent is removed, the platform could still linger in the discovery queue as a ghost entry.

2. Stop the agent.
3. Delete the agent's installation directory.

10.1.2. Removing an Agent RPM

1. Remove the platform entry from the JBoss ON server inventory.



Important

If the agent was configured, but the platform was never imported into the inventory, then *you must import the platform from the discovery queue first*, and then delete the platform. The discovery queue is a halfway point in the inventory. Even if the agent is removed, the platform could still linger in the discovery queue as a ghost entry.

2. Stop the agent service.

```
[root@server ~]# service jon-agent stop
```

3. If the package was installed using **yum**, then use the **yum** to remove the package:

```
[root@server ~]# yum remove jboss-on-agent jboss-on-agent-init
```

If the RPM package was installed using **rpm**, then uninstall it using **rpm**:

```
[root@server ~]# rpm -e jboss-on-agent-3.1.2.0.GA1 jboss-on-agent-init-3.1.2.0.GA1
```

10.1.3. Removing an Agent on Windows

1. Stop the agent.
2. If the agent is configured as a Windows service, then remove it as a service.

```
> rhq-agent.bat remove
```

3. Delete the agent's installation directory.

10.2. Uninstalling the Server

Removing a server still leaves the database and its information intact, so historic data remain available directly from the database itself.

10.2.1. Removing a Server on Red Hat Enterprise Linux

1. If this is the only JBoss ON server, then stop all agents. If there will be other JBoss ON servers in the topology, then agents managed by this server will naturally migrate over to the other servers in the high availability topology.
2. Stop the server.

```
> rhq-server.sh stop
```

3. Delete the server's installation directory.

10.2.2. Removing a Server on Windows

1. If this is the only JBoss ON server, then stop all agents. If there will be other JBoss ON servers in the topology, then agents managed by this server will naturally migrate over to the other servers in the high availability topology.
2. Stop the server.

```
> rhq-server.bat stop
```

3. If the server is configured as a Windows service, then remove it as a service.

```
> rhq-server.bat remove
```

4. Delete the server's installation directory.

Index

A

admin account, [Configuring the Server with the Web Installer](#), [Going Through the Web Installer](#)

agent

- automatic updates, [The Process When an Agent Autoupgrades](#)
- installation, [Installing and Upgrading the Agent from the JAR File](#)
- installation in a writable directory, [Setting Writer Permissions on the Agent Home Directory](#)
- manually upgrading, [Manually Upgrading the JBoss ON Agent](#)
- setting up the JRE, [Setting up the JRE for the JBoss ON Agent](#)

- starting as a service, [Running the JBoss ON Agent as a Service](#)
- starting as background service
 - for upgrade, [Starting the Agent as a Background Service](#)
- upgrade
 - preserving keystores and truststores, [Configuring Keystores and Truststores](#)

C

CLI

- installing, [Installing the JBoss ON CLI](#)

D

databases

- advanced Oracle configuration, [Configuring Oracle \(Advanced\)](#)
- configuring, [Configuring Oracle](#)
- configuring PostgreSQL, [Configuring PostgreSQL](#)
- editing the postgresql.conf file, [Editing the postgresql.conf File](#)
- oracle
 - setting the number of processes and sessions, [Setting the Number of Processes and Sessions](#)
 - SGA and PGA sizes, [Setting SGA and PGA Sizes](#)
 - tuning open cursors, [Tuning Open Cursors](#)
- oracle settings, [Prepping Oracle Settings](#)
- parameters
 - editing the pg_hba.conf file, [Editing pg_hba.conf](#)
- postgresql
 - Fixes for "Relation RHQ_Principal does not exist" Error, [Fixes for "Relation RHQ_Principal does not exist" Error](#)
 - setting kernel parameters, [Setting Kernel Parameters](#)
- setting PostgreSQL parameters, [Setting PostgreSQL Parameters](#)
- setting up Oracle, [Setting up Oracle](#)

default user account, [Configuring the Server with the Web Installer, Going Through the Web Installer](#)

F

firewall

- configuration, [Preparing the Host Machine, Preparing the Host Machine](#)

G

gent

- starting with init.d, [Running the Agent as a Daemon or init.d Service](#)

I

installation

- overview, [Installing and Upgrading the JBoss ON Server on Linux](#)
- troubleshooting, [Troubleshooting Installation and Upgrade](#)

J

JBoss ON server

- configuring as Red Hat Enterprise Linux service, [Configuring the Server as a Service](#)

O

oracle

- advance configuration, [Configuring Oracle \(Advanced\)](#)
- configuration, [Configuring Oracle](#)

P

prerequisites, [JBoss Operations Network 3.1.2 Prerequisites](#)

R

Red Hat Enterprise Linux

- JBoss ON running as a service, [Configuring the Server as a Service](#)

S

server

- configuring as a Windows service, [Configuring the Server as a Windows Service](#)
- configuring DNS, [Preparing the Host Machine](#), [Preparing the Host Machine](#)
- downloading server packages, [Installing and Upgrading the JBoss ON Server on Linux](#)
- host machine preparation, [Preparing the Host Machine](#), [Preparing the Host Machine](#)
- running on Red Hat Enterprise Linux, [Installing and Upgrading the JBoss ON Server on Linux](#)
- setting up the JDK, [Setting up the JDK for the JBoss ON Server](#), [Setting up the JDK](#)
- troubleshooting install and upgrade, [Troubleshooting Installation and Upgrade](#)
- upgrading, [Upgrading the JBoss ON Server](#), [Upgrading the JBoss ON Server](#)

T

troubleshooting, [Troubleshooting Installation and Upgrade](#)

U

upgrade

- agent
 - starting as background service, [Starting the Agent as a Background Service](#)
- troubleshooting, [Troubleshooting Installation and Upgrade](#)

upgrading, [Upgrading JBoss ON](#), [Upgrading JBoss ON](#)**users**

- default admin, [Configuring the Server with the Web Installer](#), [Going Through the Web Installer](#)

W

Windows

- running as a service, [Configuring the Server as a Windows Service](#)