JBoss Operations Network 3.0
Installation Guide

for installing servers and agents

Ella Deon Lackey

JBoss Operations Network 3.0 Installation Guide

for installing servers and agents

Ella Deon Lackey
dlackey@redhat.com

Legal Notice

Abstract

This manual covers the installation and setup of JBoss ON 3.0 servers and agents and basic tasks for configuring the inventory.

Table of Contents

Preface

JBoss Operations Network 3.0 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-3.0.0.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 nam and directory paths, and any text displayed in a p

Formatting Style	Purpose

`Monospace`
`with a`
`background`
	This type of formatting is used for anything entere returned in a command prompt.

Italicized text	Any text which is italicized is a variable, such as *instance_name* or *hostname*. Occasionally, this is used to emphasize a new term or other phrase.
Bolded text	Most phrases which are in bold are application na such as **Cygwin**, or are fields or options in a user interface, such as a **User Name Here:** field or **S** 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. Additional Reading

JBoss ON has several guides and documents available that help people plan, install, and deploy JBoss ON.

Guide	
Release Notes	Contains important information on new features, fixed bugs, known issues and workarounds, and other important deployment information for this specific version of JBoss Operations Network.
Planning and Features Guide	Contains information on all the features and concepts in JBoss ON, with tips and outlines to help plan how to deploy JBoss ON.

Guide	
Installation Guide	Contains complete information on how to set up databases, install JBoss ON servers and agents, and migrate existing deployments. This also contains platform support information.
Basic Admin Guide	Covers all management tasks for JBoss ON, both for managing resources and configuring the JBoss ON server and agents.
Plug-in Writing Guide	Contains procedures, references, and examples to write custom plug-ins for the server and for the agent.
Manging JBoss ON from the Command Line	Describes how to use the JBoss ON Java CLI to manage a server remotely, without the UI.
Resource Monitoring and Operations Guide	Contains a complete reference of each resource type in JBoss ON for monitoring metrics, available operations, and other configuration options.

For the latest information about JBoss ON, including current release notes, complete product documentation, technical notes, and deployment information, see the JBoss Operations Network documentation site at http://www.redhat.com/docs/en-US/JBoss_ON/.

4. Document History

Revision 3.0.1-3	2013-10-31	Rüdiger Landmann
Rebuild with publican 4.0.0		

Revision 3.0.1-2	April 2, 2012	Ella Deon Lackey
Removing the procedure for editing the agent jar file.		

Revision 3.0-0	December 7, 2011	Ella Deon Lackey
Initial release of JBoss Operations Network 3.0.		

Chapter 1. JBoss Operations Network 3.0 Prerequisites

The list of supported platforms for JBoss Operations Network are listed at
http://www.redhat.com/resourcelibrary/articles/jboss-operations-network-supported-configurations.

1.1. Supported Platforms

The JBoss Operations Network server and agent require Java 6. Any operating system which supports the appropriate JVM could be run a JBoss ON server or agent. Check with Red Hat support for specific platform and environment information.

These platforms are fully supported with native JVM support:

- Red Hat Enterprise Linux 6 (32-bit and 64-bit)

- Red Hat Enterprise Linux 5 (32-bit and 64-bit)

- Red Hat Enterprise Linux 4 (32-bit and 64-bit)

- Microsoft Windows 2008 R2 (32-bit and 64-bit)

- Solaris 11 (64-bit)

- Solaris 10 (64-bit) (deprecated)

The JBoss ON server requires a Java 6 JDK, while the JBoss ON agent requires a Java 6 JRE.

1.2. Supported Databases

JBoss Operations Network 3.0 supports these databases for the JBoss ON server:

- Oracle 11g

- Oracle 10g (deprecated)

- PostgreSQL 9.0 (new)

- PostgreSQL 8.4.x

- PostgreSQL 8.3.x

- PostregSQL 8.2.4 and later releases (deprecated)

1.3. 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

Minimum
[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.

1.4. Setting up Java

Sun and OpenJDK Java are supported.

> **Note**
>
> The server requires the JDK, while the agent only requires the JRE.

1.4.1. Setting up the JDK for the JBoss ON Server

After downloading and installing the latest Java, configure the system to use that version of Java. For example, to configure the JDK for the server on Red Hat Enterprise Linux:

1. 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/
    ```

2. 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:
    ```

1.4.2. Setting up the JRE for the JBoss ON Agent

1. 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/
```

2. 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:
```

1.5. Preparing the Server 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:

» **On Linux, 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 on Linux systems, 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.

Chapter 2. Installing JBoss ON Components

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.

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 2.3, "Installing and Configuring 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 Postgres
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

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 then 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 * *number_of_agents*	1.5 * 100 agents = 150
Servers	60 * *number_of_servers*	60 * 2 servers = 120
with Oracle Enterprise Manager	*highest_number_of_processes* + 40	1.5 * 100 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 or determine an Oracle instance to be used for the JBoss ON database.

   Run the Oracle server for JBoss ON on its own hardware, and then create a database. The database can be named anything.

2. Log into Oracle as the system user.

```
CONNECT sys/your_sys_password AS sysdba;
```

3. 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;
```

4. 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;
```

5. 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 has 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;
```

6. 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 has 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;
```

2.3. Installing and Configuring the JBoss ON Server

The JBoss ON server binaries are installed simply by unzipping the packages. After that, then the specific server instance must be configured. This can be done through a simple, web-based installer or by editing the configuration file manually and configuring it silently.

2.3.1. Using the GUI Installer (Red Hat Enterprise Linux)

> 💬 **Note**
>
> See Chapter 5, *Troubleshooting Installation and Upgrade* if there are any problems during the install process.

1. Stop any currently running JBoss ON instances.

```
serverRoot/jon-server-3.0.0.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 3.0 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-3.0.0.GA1.zip
```

   This creates a version-specific installation directory, **/opt/jon-server-3.0.0.GA1**. A directory with this name should not exist prior to the unzip operation.

4. Install any additional agent plug-ins.

   Some JBoss applications (EWS, EDS, EAP, and SOA-P) have separate plug-ins, which are only available with special subscriptions. These agent plug-ins are part of JBoss ON can be be installed and enabled as part of the server installation process.

> **💬 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*.

   a. Unzip the additional plug-in packs that were downloaded from the Customer Support Portal in step 2.

```
unzip jon-plugin-pack-agent_plugin_name-3.0.0.GA1.zip -d
/opt/jon/jon-server-3.0.0.GA1/
```

   This creates a subdirectory with the name **jon-plugin-pack-*plugin_name*-3.0.0.GA1/**.

b. List the current contents of the JBoss ON server plug-in directory.

```
ls -l serverRoot/jon-server-
3.0.0.GA1/jbossas/server/default/deploy/rhq.ear.rej/rhq-
downloads/rhq-plugins
```

c. Copy the new plug-ins from the **jon-plugin-pack-*plugin_name*-3.0.0.GA1/** directory to the JBoss ON server plug-in directory.

```
cp /opt/jon/jon-server-3.0.0.GA1/jon-plugin-pack-plugin_name-
3.0.0.GA1/* serverRoot/jon-server-
3.0.0.GA1/jbossas/server/default/deploy/rhq.ear.rej/rhq-
downloads/rhq-plugins
```

5. Run the JBoss ON server:

```
serverRoot/jon-server-3.0.0.GA1/bin/rhq-server.sh start
```

6. Open the server UI at **http://localhost:7080/**.

7. Clicking the **Click here to continue the installation** link brings you to the main installer page.

8. 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]
> **Important**
>
> Do not unselect the advanced settings checkbox after setting advanced values, or the advanced settings will be reset to the default values.

9. Set the database connection properties.

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


> ![Important]
> **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.

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

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

12. 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.

13. When the server is configured, click the link to go to the UI.

14. 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.

2.3.2. Using the GUI Installer (Windows)

> **Note**
>
> See Chapter 5, *Troubleshooting Installation and Upgrade* if there are any problems during the install process.

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-3.0.0.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 click the **Product** drop-down box arrow to open the **JBoss Operations Network** software download.

   b. Download the **JBoss Operations Network 3.0 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.

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

   ```
   mkdir C:\rhq
   ```

   > **Important**
   >
   > Do not install the server in a directory with a path longer than 19 characters. Long path names can cause problems with executing some server tasks. Use a location such as **C:\rhq** rather than **C:\Documents and Settings\myusername**.

5. Unzip the server distribution in the directory where will be executed from. For example, using 7-Zip:

   ```
   cd C:\rhq

   7z e jon-server-3.0.0.GA1.zip
   ```

This creates a version-specific installation directory, such as **C:\rhq\jon-server-3.0.0.GA1**. A directory with this name should not exist prior to the unzip operation.

6. Install any additional agent plug-ins.

   Some JBoss applications (EWS, EDS, EAP, and SOA-P) have separate plug-ins, which are only available with special subscriptions. These agent plug-ins are part of JBoss ON can be be installed and enabled as part of the server installation process.

   > **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*.

   a. Unzip the additional plug-in packs that were downloaded from the Customer Support Portal in step 3. This creates a subdirectory with the name **jon-plugin-pack-*plugin_name*-3.0.0.GA1**.

   ```
   cd C:\rhq

   7z e  jon-plugin-pack-agent_plugin_name-3.0.0.GA1.zip
   ```

   b. Copy the new plug-ins from the **jon-plugin-pack-*plugin_name*-3.0.0.GA1** directory to the JBoss ON server plug-in directory.

7. Set the directory path to the JRE or JDK installation. For example:

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

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

   ```
   C:\rhq\jon-server-3.0.0.GA1\bin\rhq-server.bat install
   ```

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

   ```
   C:\rhq\jon-server-3.0.0.GA1\bin\rhq-server.bat start
   ```

10. Open the server UI at **http://localhost:7080/**.

11. Complete steps 6 to 14 in Section 2.3.1, "Using the GUI Installer (Red Hat Enterprise Linux)".

2.3.3. Silently Installing the JBoss ON Server

The initial setup of the server is the same as in Section 2.3.1, "Using the GUI Installer (Red Hat Enterprise Linux)". 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.

> **Note**
>
> This procedure uses command examples for Red Hat Enterprise Linux. If you are installing the server on a Windows server, then adapt the commands accordingly.

1. Stop any currently running JBoss ON instances.

   ```
   serverRoot/jon-server-3.0.0.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 3.0 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. For example, on Red Hat Enterprise Linux:

```
cd /opt

unzip jon-server-3.0.0.GA1.zip
```

On Windows, using 7-Zip:

```
cd C:\rhq

7z e jon-server-3.0.0.GA1.zip
```

On Windows systems, create or select a directory with a relatively short name. Path names longer than 19 characters can cause problems with executing some server tasks. Use a location such as **C:\rhq** rather than **C:\Documents and Settings\myusername**.

4. Install the agent plug-ins. On Red Hat Enterprise Linux:

```
cd /opt/jon/jon-server-3.0.0.GA1

unzip jon-plugin-pack-agent_plugin_name-3.0.0.GA1.zip
```

On Windows:

```
cd C:\rhq

7z e  jon-plugin-pack-agent_plugin_name-3.0.0.GA1.zip
```

5. Open the JBoss ON server configuration properties file:

```
vim serverRoot/jon-server-3.0.0.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.

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

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
```

10. 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-3.0.0.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
```

11. 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 2.2, "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
```

12. Start the JBoss ON server:

```
serverRoot/jon-server-3.0.0.GA1/bin/rhq-server.sh start
```

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

Table 2.2. 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.

Parameter	Description
rhq.server.database.connection-url	The JDBC URL that the JBoss ON server uses when connecting to the database. An example is *jdbc:postgresql://localhost:5432/rhq* or *jdbc:oracle:oci:@localhost:1521:orcl*.
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 *oracle.jdbc.driver.OracleDriver*.
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 *org.postgresql.xa.PGXADataSource* or *oracle.jdbc.xa.client.OracleXADatasource*.
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`.

2.4. Installing Additional JBoss Agent Plug-ins

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.

The installation procedure at Section 2.3, "Installing and Configuring the JBoss ON Server" includes the steps to install these additional plug-in packages as part of installing a new JBoss ON server. These resource plug-ins can also be installed after the server is installed.

> **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, click **Software**, and then select the **JBoss ON for** *Plug-in* drop-down box.

2. Download the plug-in packs.

3. Unzip the additional plug-in packs. This creates a subdirectory with the name **jon-plugin-pack-*plugin_name*-3.0.0.GA1**.

4. List the current contents of the JBoss ON server plug-in directory. For example:

   ```
   [root@server rhq-agent]# ls -l  serverRoot/jon-server-
   3.0.0.GA1/jbossas/server/default/deploy/rhq.ear/rhq-downloads/rhq-
   plugins
   ```

5. Stop the JBoss ON server.

   ```
   serverRoot/jon-server-3.0.0.GA1/bin/rhq-server.sh stop
   ```

6. Copy the new plug-ins from the **jon-plugin-pack-*plugin_name*-3.0.0.GA1/** directory to the JBoss ON server plug-in directory.

   ```
   [root@server rhq-agent]# cp /opt/jon/jon-server-3.0.0.GA1/jon-plugin-
   pack-plugin_name-3.0.0.GA1/* serverRoot/jon-server-
   3.0.0.GA1/jbossas/server/default/deploy/rhq.ear/rhq-downloads/rhq-
   plugins
   ```

7. Start the JBoss ON server again.

   ```
   serverRoot/jon-server-3.0.0.GA1/bin/rhq-server.sh start
   ```

8. Have the agents reload their plug-ins to load the new plug-ins. This can be done from the command line using the agent's **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 or agents.

Chapter 3. Installing the JBoss ON Agent

In JBoss ON 3.0, the JBoss ON agent is bundled with the JBoss ON server and is not available as a separate download. The same packages are used to install a new agent and to upgrade an existing agent.

3.1. Installing the JBoss ON Agent

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 3.2, "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.

5. 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 3.1, "All Options Available During Advanced Setup".

6. Configure the agent as a background service, as in Section 4.4, "Running the JBoss ON Agent as a Service".

> **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 property 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 3.1. 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

Setup Option	Description	Normal or Advanced Setup
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
RHQ Server Hostname or IP Address	Gives the IP address or hostname of the primary server that the agent communicates with. This information *must* 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 *must* 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 *must* 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 *must* 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

Setup Option	Description	Normal or Advanced Setup
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
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 *cannot connect* 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

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.

3.2. 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 3.2, "Configuration File Keys for Agent Setup" have to be set in the file.

Table 3.2. 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.

Installer Prompt Text	Key Name	Description
[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.

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
```

Chapter 4. Quick Set up for JBoss ON

After installing the JON Server, you will be directed to the JON GUI Console after you click the **Done! Click here to get started!** message on the installer page. Alternatively, you can point your browser to http://localhost:7080.

4.1. Logging into the JBoss ON Web UI

JBoss ON is primarily administered through its web interface. This UI can be accessed from any web browser by connection to the server using the URL **http://**server-hostname:port.

By default, the JBoss ON server listens over port 7080. (A different port can be configured when the server is installed, and the port number can be changed in the server configuration.) To connect to the server, then, simply open a standard HTTP page with a URL in the format *hostname:port*. For example:

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

Then, log in using any valid username/password combination. The default administrative user has the name and password **rhqadmin/rhqadmin**. (The default username and password are defined in the JBoss ON server.) After you log in, reset the password for the **rhqadmin** superuser.
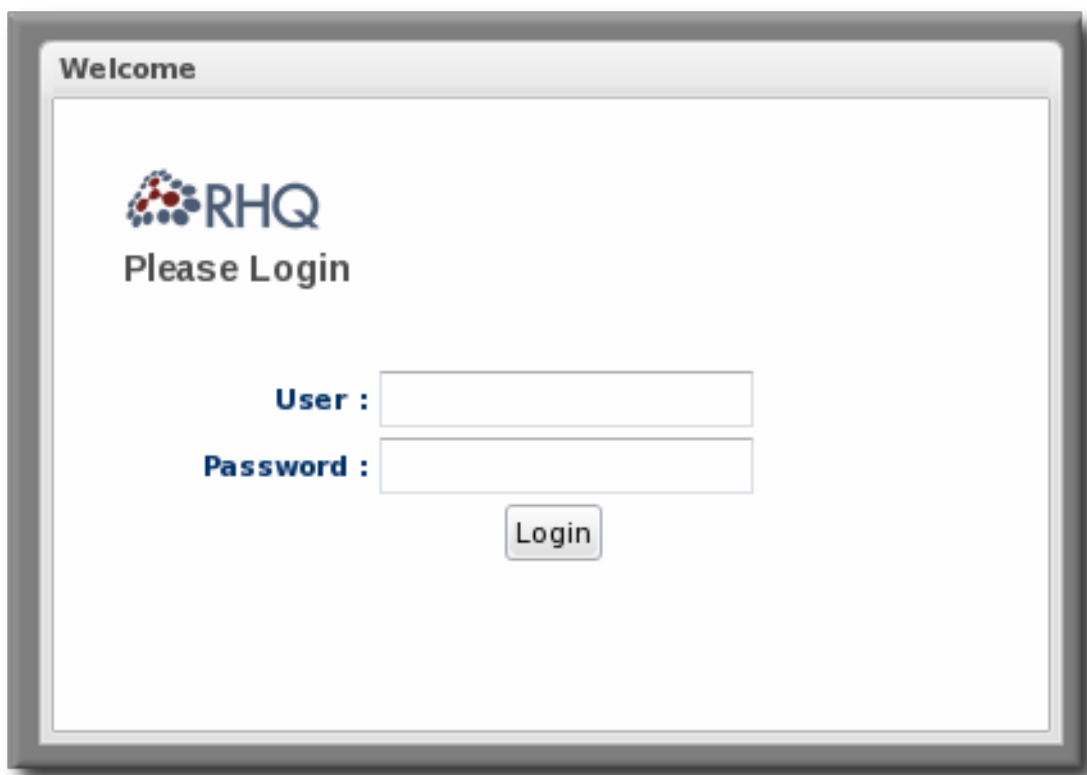
Figure 4.1. Logging into JBoss ON

> **Note**
>
> By default, JBoss ON only runs over standard HTTP. SSL must be specially configured to open a secure connection over HTTPS. For HTTPS, JBoss ON uses the connection port 7443.

4.2. Initially Discovering and Importing Resources

Once the agent is installed and configured, it runs discovery scans for resources on the machine and report them back to the JBoss ON server. Before resources can be managed from the JBoss ON server, those resources must be imported into inventory, which essentially tags those resources as something that should be recognized and managed by JBoss ON. (Not all services or servers may need to be managed by JBoss ON, and those can be ignored by JBoss ON.)

Resources are imported through the **Discovery Queue**.

1. In the top navigation menu, click the **Inventory** tab.

2. Select the **Discovery Queue** item in the left navigation area.

3. Any newly-discovered resources are listed in the **Discovery** table. The resources are listed in a hierarchy under the parent server (even if the server has already been imported).

4. Select the checkbox by the newly-discovered platform and agree to select all of its children.

5. Click the **Import** button.
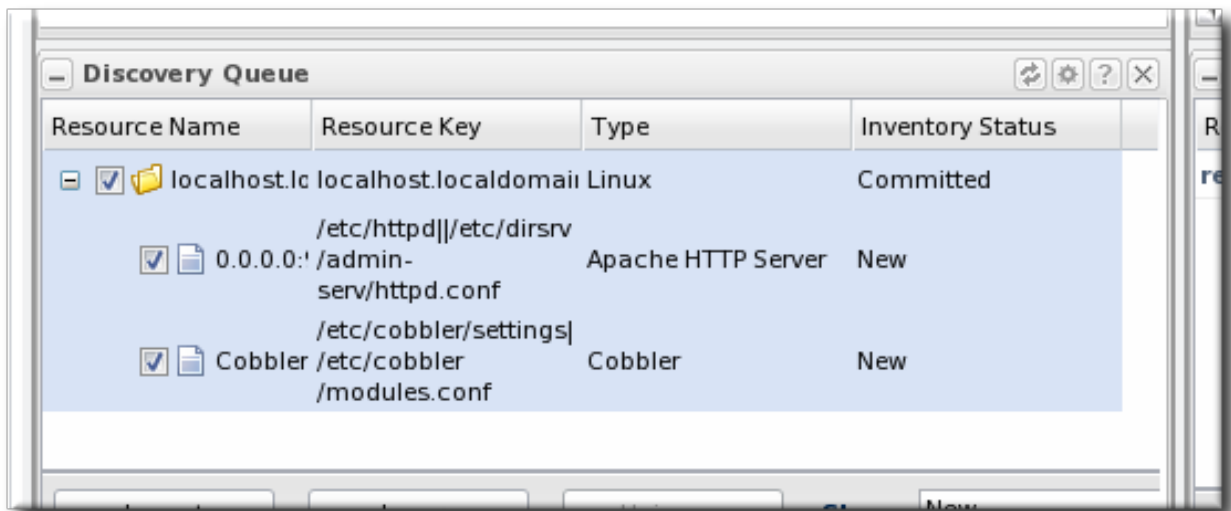
Even after the platform has been imported, the JBoss ON agent continues to run more detailed discovery scans. Discovery checks for child servers and services of the imported resources. All of these children resources are imported automatically.

The **Inventory** tab can display all resources in the JBoss ON inventory.
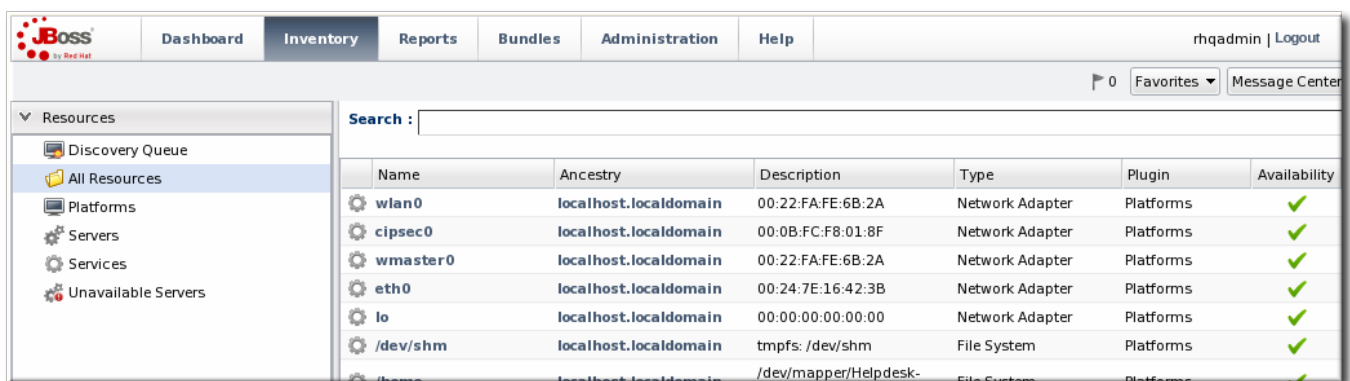
Figure 4.2. Inventory

The agent continues to report data about the resources in inventory to the JBoss ON server, including availability and metric data.

4.3. Running the JBoss ON Server as a Service

4.3.1. Configuring the JBoss ON Server as a Service on Red Hat Enterprise Linux

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-3.0.0.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
   ```

4.3.2. Configuring JBoss ON 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 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 as 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
```

4.4. 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.

4.4.1. Running the Agent as a Windows 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. 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.

4.4.2. 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
```

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 4.1, "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 4.1. 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"

Feature	Description	Doc Link
Configuring roles and access control	JBoss ON defines access to resources based on *roles*. 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 *drifts* 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 *bundle*, 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"

Feature	Description	Doc Link
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 *affinity*, for agents to be managed by selected servers. This can improve performance or be used to reflect the infrastructure topology.	"Configuring High Availability"

Chapter 5. 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.

5.1. Exceptions and Error Logs

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 3.0 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 3.0, 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.

5.2. 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.

5.3. 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 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-***3.0***.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. 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 3.0 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.

7.1. 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 5, 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 3.0, 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 7.3.1, "Preparing the Agent for 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 4.4, "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 7.3.3, "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-3.0.0.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-3.0.0.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-3.0.0.GA1**.

6. Copy over any changes in your original **rhq-server.properties** file to the new file in **serverRoot/jon-server-3.0.0.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 3.0, 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. *Windows only.* 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
```

8. 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-3.0.0.GA1

unzip jon-plugin-pack-agent_plugin_name-3.0.0.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.

9. Start the JBoss ON server. For example:

```
serverRoot/jon-server-3.0.0.GA1/bin/rhq-server.sh start
```

10. 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.

11. Open the web UI.

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

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

12. The setup process is the same as the initial setup procedure in Chapter 2, *Installing JBoss ON Components*.

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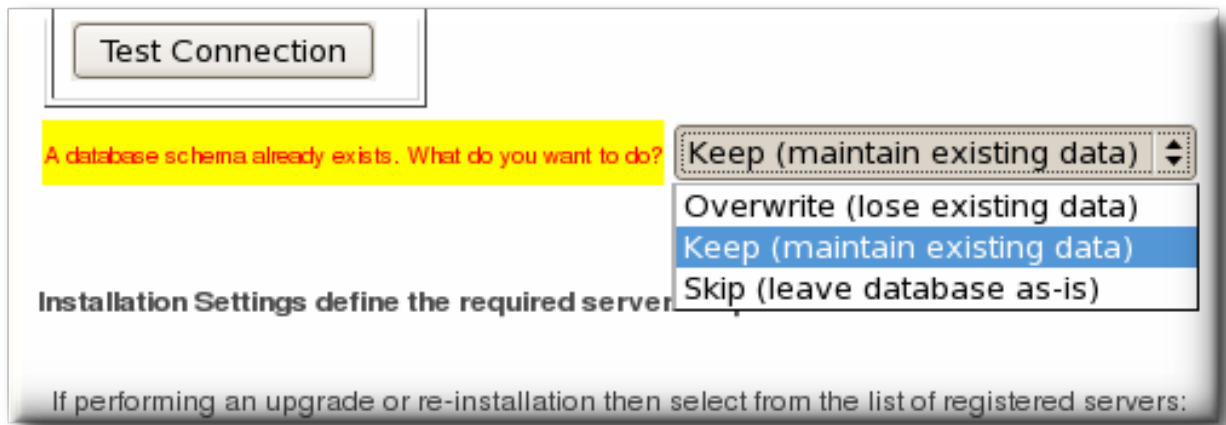
Choose the default, **Keep (maintain existing data)**. Do *not* choose **Overwrite (lose existing data)**, or the installer will delete all of your JBoss ON data, including your inventory, monitoring history, alerts, and metrics.

13. 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**.
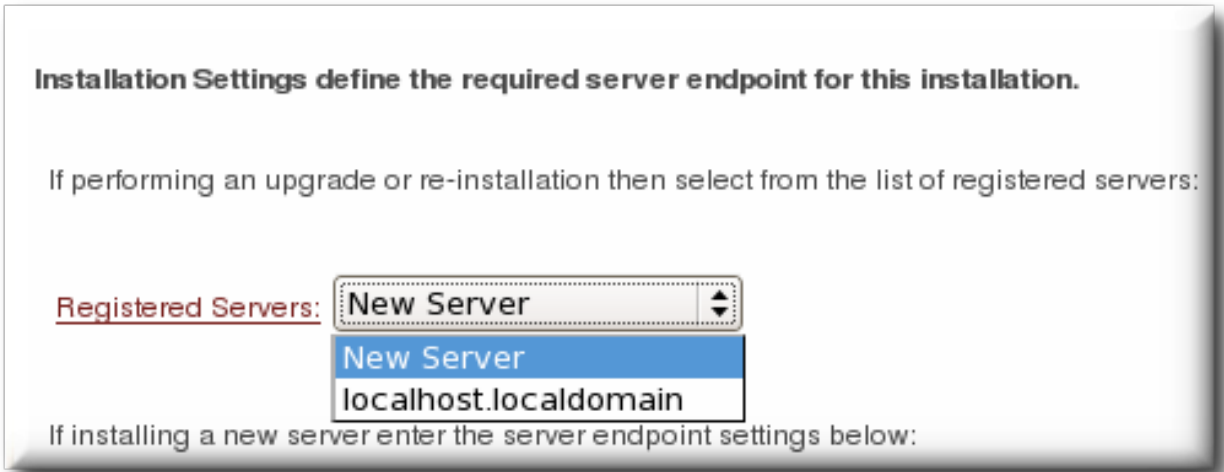
14. Start any JBoss ON agents that were stopped for the upgrade process.

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.

Once the server is upgraded, one visible change is that the upgraded GUI uses new icons and style sheets.
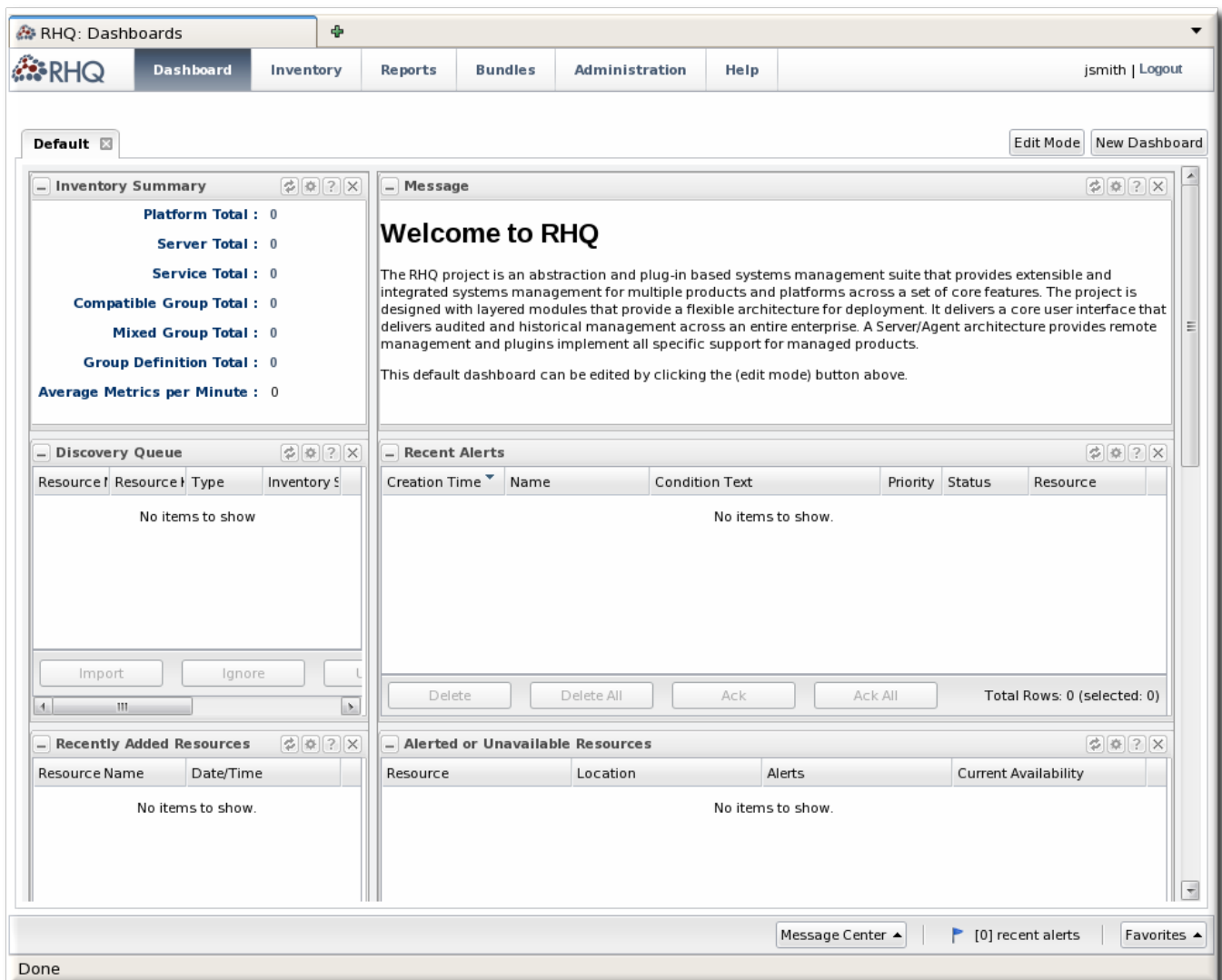
Figure 7.1. Updated UI

> **Note**
>
> Because there are CSS changes in JBoss ON 3.0, the GUI has a different layout and defined classes in JBoss ON 3.0 than in previous releases. After an upgrade, if the GUI is already open or if the style data are cached, then the JBoss ON UI will be poorly formatted. Manually reload the CSS and image files using `Ctrl+F5`.

7.2. 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.

7.2.1. 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.

7.2.2. 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.
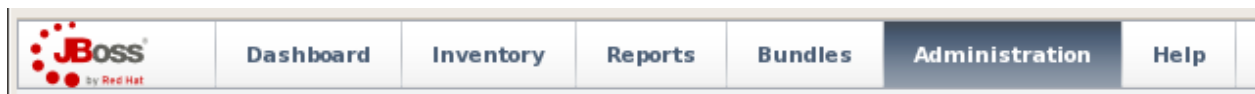
2. In the `Configuration` box on the left navigation bar, click the `Plugins` link.
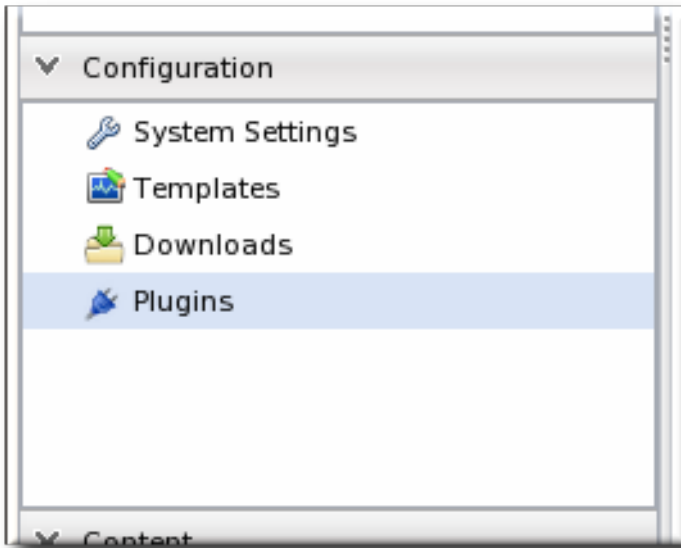
3. Click the **Server Plugins** tab.
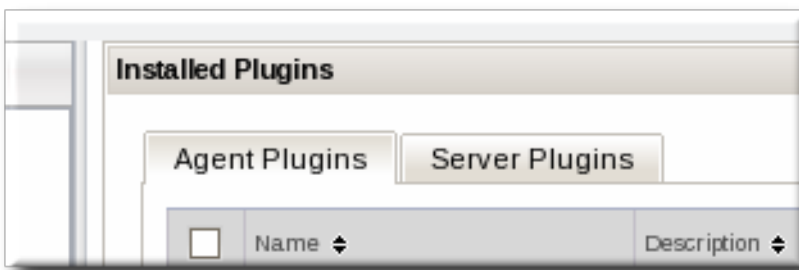
4. Click the name of the SNMP plug-in in the list.

5. In the plug-in details page, click the **Configure 'Alert:SNMP'** link to open the configuration page for the plug-in.
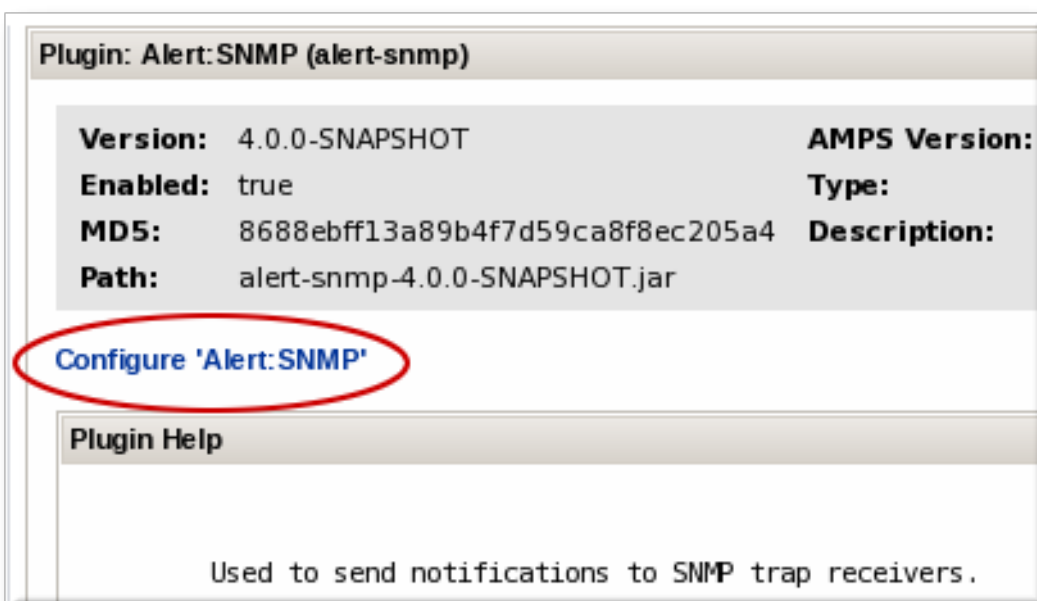
6. Click the **EDIT** button at the bottom of the configuration screen to make the fields active.

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.

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

7.3. Upgrading the Agent

7.3.1. Preparing the Agent for 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.

7.3.1.1. 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 five files which should never be modified:
>
> » **rhq-agent.sh**
> » **rhq-agent-wrapper.sh**
> » **rhq-agent.bat**
> » **rhq-agent-wrapper.bat**
> » the rhq-agent runtime
>
> 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.
>
> The default location for the rhq-agent runtime is **$USERHOME/.java/.userPrefs/rhq-agent/default**.

7.3.1.2. 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.

7.3.1.3. 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
```

7.3.1.4. Starting the Agent as a Background Service

Section 4.4, "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.

7.3.2. Automatically Updating the Agent

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 4.4, "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:

1. The updated server puts the updated agent packages in a directory accessible to the agent.

2. 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
```

7.3.3. 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 7.1, "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 --upgrade
   ```

   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.

Chapter 8. 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.

8.1. Uninstalling an Agent

8.1.1. Removing an Agent on Red Hat Enterprise Linux

1. Stop the agent.

2. Delete the agent's installation directory.

8.1.2. 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.

8.2. Uninstalling the Server

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

8.2.1. 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.

8.2.2. 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