



JBoss Operations Network 3.1

Initial Setup for the Resource Inventory Groups and Users

for initially setting up resources and security in JBoss ON

Edition 3.1

JBoss Operations Network 3.1 Initial Setup for the Resource Inventory Groups and Users

for initially setting up resources and security in JBoss ON
Edition 3.1

Ella Deon Lackey
dlackey@redhat.com

Legal Notice

Copyright © 2012 Red Hat, Inc..

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

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

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

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

Abstract

Once JBoss Operations Network is installed, you are ready to set up your inventory. The inventory in JBoss ON lists all of the recognized platforms, servers, and services which are managed by JBoss ON – these are your resources. These resources can be organized into groups to help simplify management tasks. Security-related entries like users, roles, and assigned permissions also need to be created once JBoss ON is installed. This guide provides GUI-based procedures to set up all of your resource entries, groups, and security entries, as well as providing a basic overview of the JBoss ON GUI and tasks.

Table of Contents

1. USING THE JBOSS ON WEB INTERFACE	4
1.1. Supported Web Browsers	4
1.2. Logging into the JBoss ON Web UI	4
1.3. A High Level Walk-Through	5
1.3.1. The Top Menu	6
1.3.2. The Left Menu	7
1.3.3. Dashboard	9
1.3.4. Inventory Browsers and Summaries	10
1.3.5. Entry Details Pages	10
1.3.6. Shortcuts in the UI	13
1.4. Getting Notifications in the Message Center	14
1.5. Sorting and Changing Table Displays	14
1.6. Customizing the Dashboard	16
1.6.1. Editing Portlets	16
1.6.2. Adding and Editing Dashboards	16
1.7. Setting Favorites	18
1.8. Deleting Entries	19
2. DYNAMIC SEARCHES FOR RESOURCES AND GROUPS	19
2.1. About Search Suggestions	20
2.2. About the Dynamic Search Syntax	21
2.2.1. Basic String Searches	21
2.2.2. Property Searches	24
2.2.3. Complex AND and OR Searches	26
2.3. Saving, Reusing, and Deleting Dynamic Searches	27
3. VIEWING AND EXPORTING REPORTS	28
3.1. Types of Reports	28
3.2. Exporting Report Data to CSV	30
4. MANAGING THE RESOURCE INVENTORY	31
4.1. About the Inventory: Resources	32
4.1.1. Managed Resources: Platforms, Servers, and Services	32
4.1.2. Content-Backed Resources	33
4.1.3. Resources in the Inventory Used by JBoss ON	34
4.2. Interactions with System Users for Agents and Resources	34
4.2.1. The Agent User	35
4.2.2. Agent Users and Discovery	35
4.2.3. Users and Management Tasks	35
4.2.4. Using sudo with JBoss ON Operations	36
4.3. Discovering Resources	37
4.3.1. Finding New Resources: Discovery	37
4.3.2. Running Discovery Scans Manually	38
4.3.3. Importing Resources from the Discovery Queue	39
4.3.4. Ignoring Discovered Resources	39
4.4. Importing New Resources Manually	39
4.5. Setting up a JVM for Discovery	41
4.5.1. Required JVM Configuration for Discovery	41
4.5.2. Excluding Java Processes from Discovery	41
4.5.3. Manually Importing a JVM Resource	42
4.6. Configuring Tomcat/EWS Servers for Discovery (Windows)	44
4.7. Creating Child Resources	44

4.8. Viewing and Editing Resource Information	46
4.9. Managing Connection Settings	48
4.10. Uninventorying and Deleting Resources	49
4.10.1. A Comparison of Uninventorying and Deleting Resources	49
4.10.2. Use Caution When Removing Resources	50
4.10.3. Uninventorying through the Inventory Tab	50
4.10.4. Uninventorying through the Parent Inventory	51
4.10.5. Uninventorying through a Group Inventory	52
4.10.6. Deleting a Resource	54
4.11. Viewing Inventory Summary Reports	55
5. MANAGING GROUPS	56
5.1. About Groups	56
5.1.1. Dynamic and Static Groups	57
5.1.2. About Autogroups	57
5.1.3. Comparing Compatible and Mixed Groups	58
5.1.4. Leveraging Recursive Groups	60
5.2. Creating Groups	60
5.3. Changing Group Membership	62
5.4. Editing Compatible Group Connection Properties	63
6. USING DYNAMIC GROUPS	65
6.1. About Dynamic Groups Syntax	65
6.1.1. General Expression Syntax	65
6.1.2. Simple Expressions	68
6.1.3. Pivot Expressions	68
6.1.4. Compound Expressions	69
6.1.5. Unsupported Expressions	70
6.1.6. Dynagroup Expression Examples	71
6.2. Creating Dynamic Groups	72
6.3. Recalculating Group Members	74
7. CREATING USER ACCOUNTS	75
7.1. Managing the rhqadmin Account	75
7.2. Creating a New User	76
7.3. Editing User Entries	77
7.4. Disabling User Accounts	78
7.5. Changing Role Assignments for Users	79
8. MANAGING ROLES AND ACCESS CONTROL	80
8.1. Security in JBoss ON	80
8.1.1. Access Control and Permissions	80
8.1.2. Access and Roles	83
8.2. Creating a New Role	84
8.3. Extended Example: Read-Only Access for Business Users	87
9. INTEGRATING LDAP SERVICES FOR AUTHENTICATION AND AUTHORIZATION	88
9.1. Supported Directory Services	88
9.2. LDAP for User Authentication	88
9.2.1. About LDAP Authentication and Account Creation	88
9.2.2. Issues Related to Using LDAP for a User Store	89
9.2.3. Configuring LDAP User Authentication	91
9.3. Roles and LDAP User Groups	93
9.3.1. About Group Authorization	93

9.3.2. Associating LDAP User Groups to Roles	95
9.4. Extended Example: memberOf and LDAP Configuration	96
10. DOCUMENT INFORMATION	98
10.1. Giving Feedback	98
10.2. Document History	98
INDEX	100

1. USING THE JBOSS ON WEB INTERFACE

JBoss Operations Network has a rich, layered UI which covers a broad range of functionality. This chapter gives a brief summary of the major sections of the UI so that users can more effectively perform management tasks.

1.1. Supported Web Browsers

JBoss ON supports these browser releases for accessing the server GUI:

- Firefox 10
- Internet Explorer 8

1.2. Logging into the JBoss ON Web UI

Aside from some minor configuration in its `rhq-server.properties` file, JBoss ON is completely administered through its web interface.

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`.

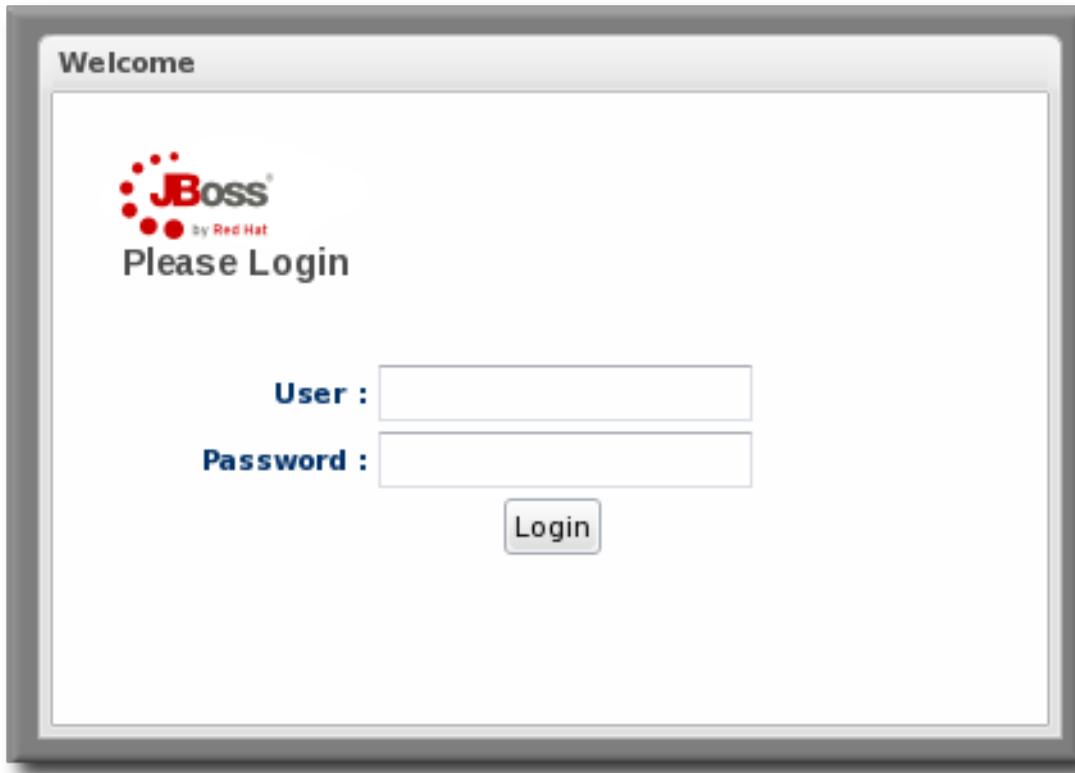


Figure 1. Logging into JBoss ON



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.

1.3. A High Level Walk-Through

The JBoss ON UI is very rich – there are a lot of small elements that are all layered together to provide a very detailed and flexible interface for interacting with the JBoss ON servers and resources. To maximize its use of space, JBoss ON uses top navigation menus, tabbed browsing with subtabs, active links, and navigation trees to establish relationships between JBoss ON resources and JBoss ON functionality. In a very general view, several types of visual elements that work together to comprise the UI:

- The top menu
- The left menu tables
- The dashboard
- Resource-based tables, which can be for the resource inventory, a summary report, or the results of a search
- Configuration pages which both provide details for and access to elements in JBoss ON, including resources, groups, plug-ins, and JBoss ON server settings

All of these elements fit together in a repeated and reliable pattern.

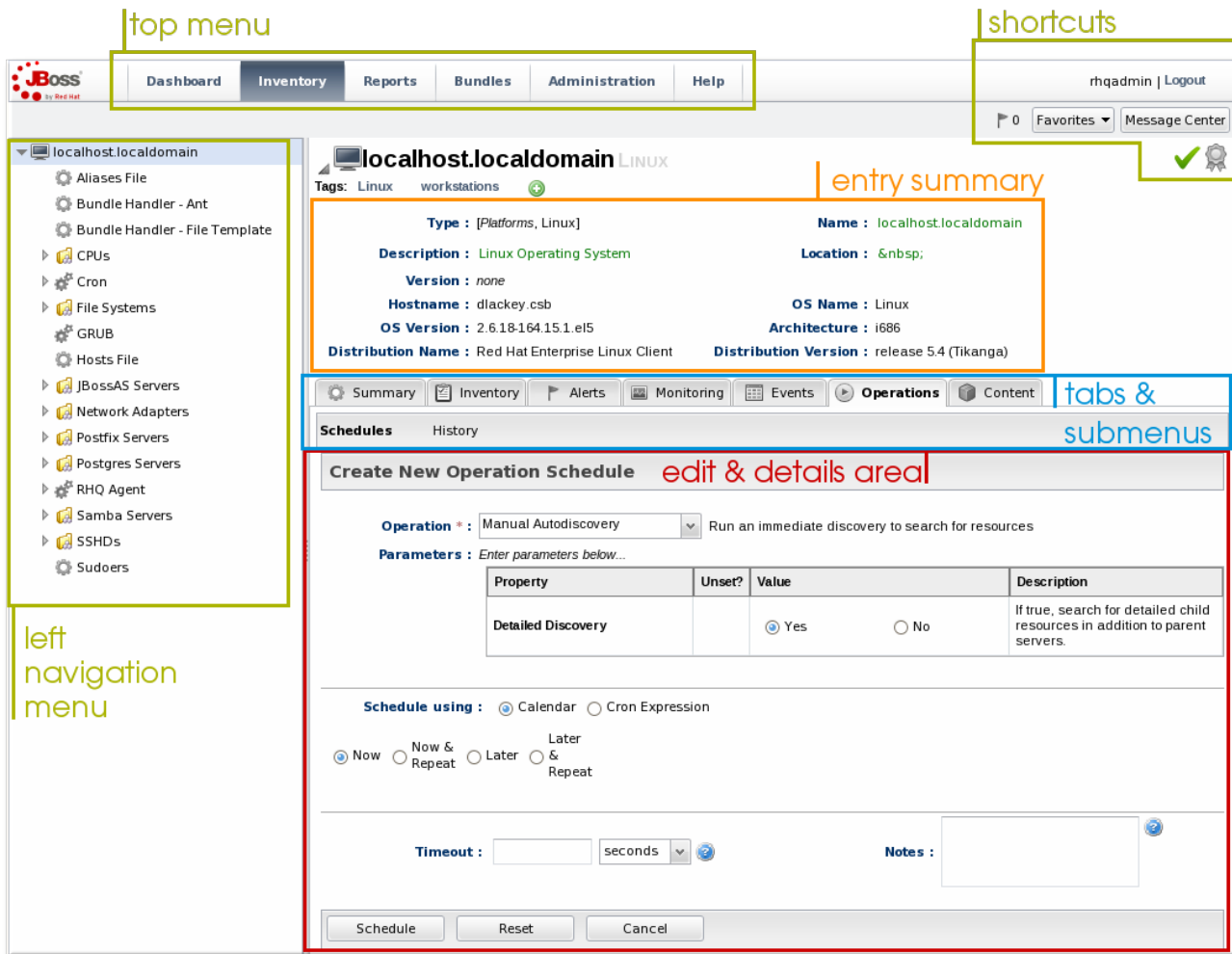


Figure 2. UI Elements All Together

Understanding the type of page that you're on can make it easier to navigate through the JBoss ON UI and can help you more completely understand what you can accomplish in JBoss ON.

1.3.1. The Top Menu

At the very top of the JBoss ON UI is a menu bar with, with five tabs that go to the major configuration areas of JBoss ON.

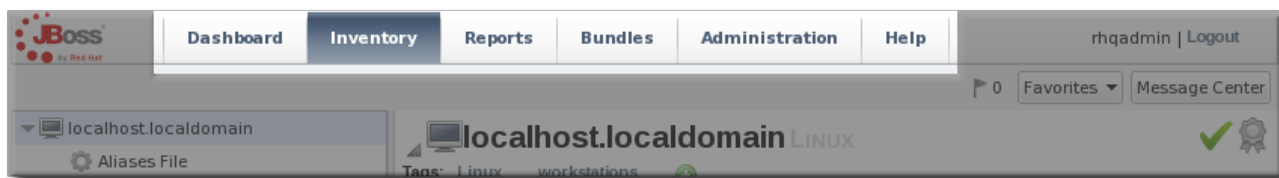


Figure 3. The Top Menu

Each menu item relates to a different functional aspect of JBoss ON.

- The Dashboard contains a global overview of JBoss ON and its resources. Different, configurable snapshot summaries (called *portlets*) show different aspects of the resources and server, such as the discovery queue, recent alerts, recent operations, and resource counts.
- The Inventory tab shows both resources and groups.
- The Reports tab shows pre-defined reports. These are slightly different than the Dashboard,

which focuses exclusively on resource information: the reports look at the current actions of the different subsystem (or major functional areas) of JBoss ON, such as alerts, operations, metric collection, and configuration history.

- The Bundles tab opens the provisioning and content functional area. This is for uploading and deploying content bundles that are used to provision new applications.
- Administration goes to all areas related to configuring the JBoss ON server itself. This includes server settings, plug-ins, users and security, and agent settings.

1.3.2. The Left Menu

Rather than using drop-down or tabbed options, much of the configuration for JBoss ON is accessed through the left menu. There are individual tables that contain related areas of configuration, like users and groups, server configuration, server/agent connections, and content for the Administration area in [Figure 4, “The Left Menu”](#).

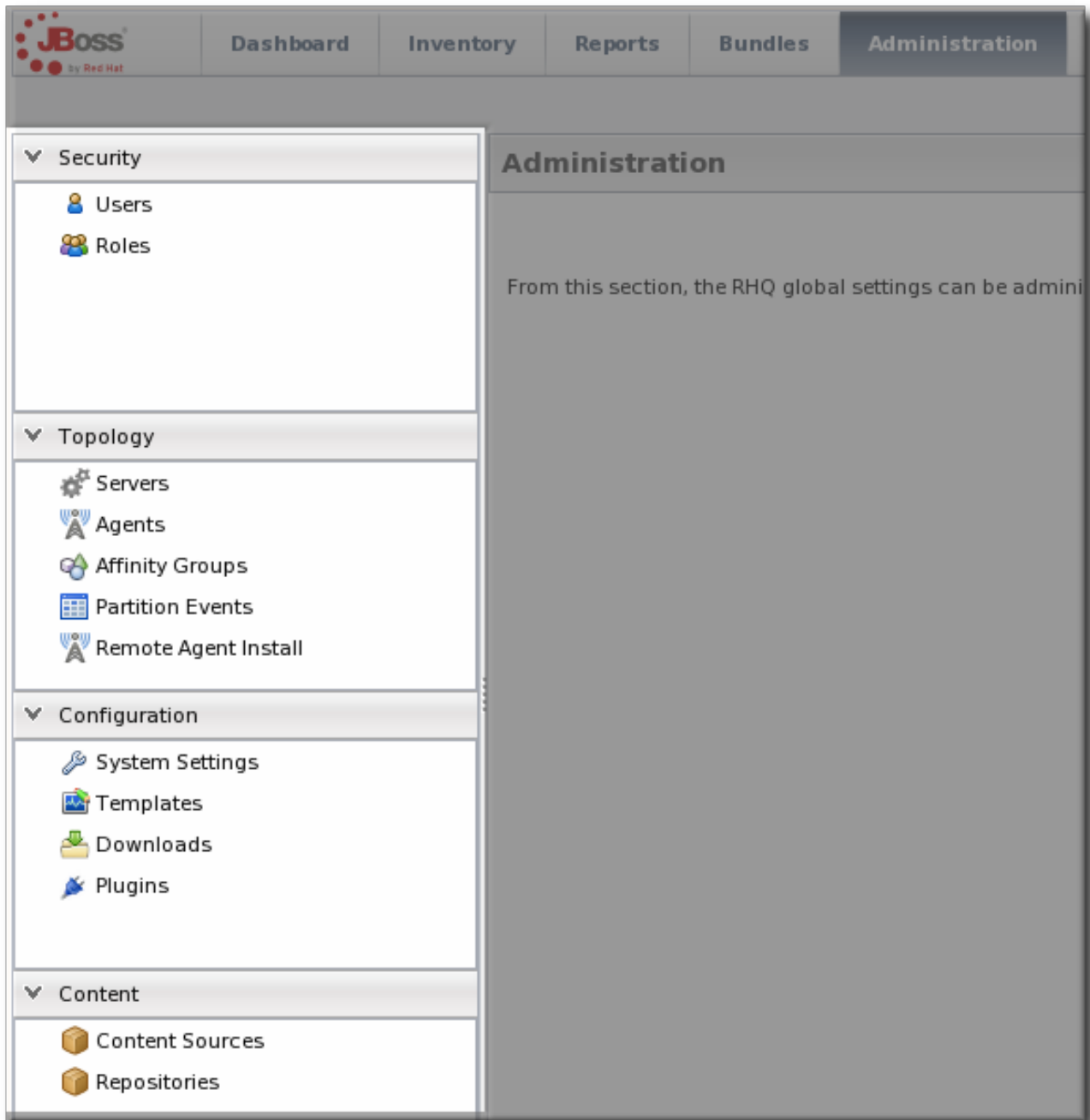


Figure 4. The Left Menu

Clicking the up or down arrows at the left of the menu tables collapses and expands the tables. This can make it easier to navigate the left menu.

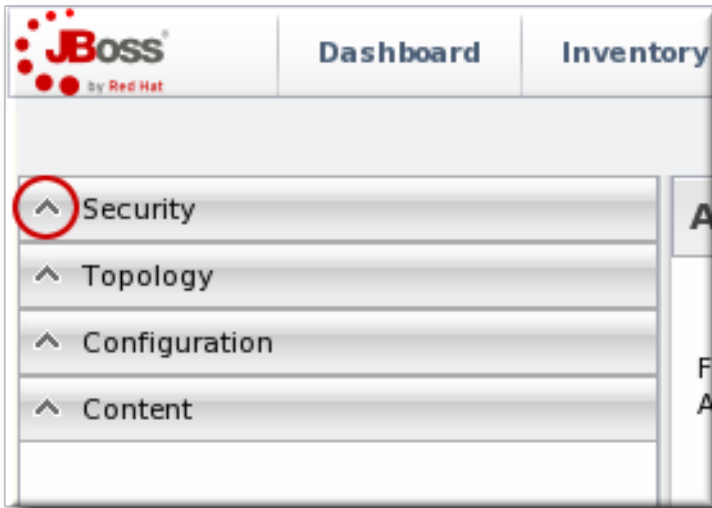


Figure 5. Collapsing the Left Menu

1.3.3. Dashboard

The Dashboard is an overview of everything in JBoss ON, from recent actions (fired alerts and operations) to availability reports to newly discovered and imported resources. This page, unlike any other area of JBoss ON, is customizable so it can be used to display only the collection of information that you want to see. Each table of information is called a *portlet*, a mini-portal into a view of the JBoss ON server or resources. There can be multiple Dashboard views configured, with different portlets or different layouts; these are accessed by tabs at the top of the Dashboard page.

The dashboard view shows the following data:

Inventory Summary

- Platform Total : 1
- Server Total : 12
- Service Total : 509
- Compatible Group Total : 0
- Mixed Group Total : 0
- Group Definition Total : 0
- Average Metrics per Minute : 194

Discovery Queue

Resource Name	Resource Key	Type	Inventory Status
localhost	localhost.localdon	Linux	Committed
/etc/httpd	/etc/httpd	Apache HTTP Server	New
0.0.0	/etc/ dirsrv/admin- serv/httpd.conf	Apache HTTP Server	New
/etc/cobbler	/etc/cobbler	Cobbler	New
Cobt	/etc/cobbler /modules.conf	Cobbler	New

Alerted or Unavailable Resources

Resource	Ancestry	Alerts	Availability
redhat0	localhost.localdomain	0	⚠

Total: 1

Recent Alerts

No results found using specified criteria.

Recent Operations

No items to show.

Figure 6. Dashboard View

The Dashboard is the landing page for JBoss ON, the first page that comes up after login.

1.3.4. Inventory Browsers and Summaries

Some pages are essentially long tables of information, presented in basically the same way:

- Tabs for different areas, with subtabs that further break down information
- A table of results
- Icons that open a configuration or task option for that specific entity
- Buttons that perform actions (create, delete, or some other specific action) on the entries; some of these buttons aren't active unless an entry is selected

The inventory interface in [Figure 7, “Inventory Browser”](#) is rich with functionality. The search bar for resources and groups uses a specialized syntax and flexible dynamic search. Hovering over any resource name gives a small popup message with more information about that resource. Clicking the name of the entry itself opens its default entry configuration page, while clicking the name of its parent opens up that parent resource's configuration page.

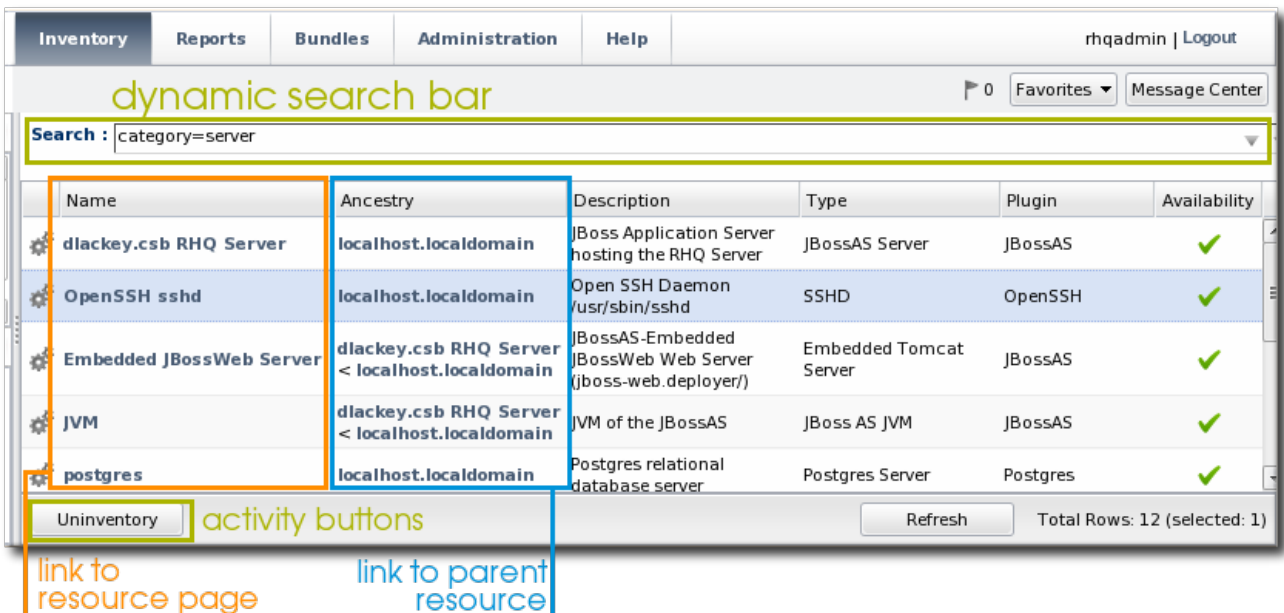


Figure 7. Inventory Browser

In [Figure 7, “Inventory Browser”](#), the **UNINVENTORY SELECTED** button is active because a resource is selected. If no entries are actively selected, activity buttons are grayed out.

1.3.5. Entry Details Pages

Possibly the most functionality-saturated area in JBoss ON is an entry's details page.

The left navigation area shows the hierarchy, both parents and children, of the selected resource. This makes it very easy to navigate among all of the different services and servers that affect a resource.

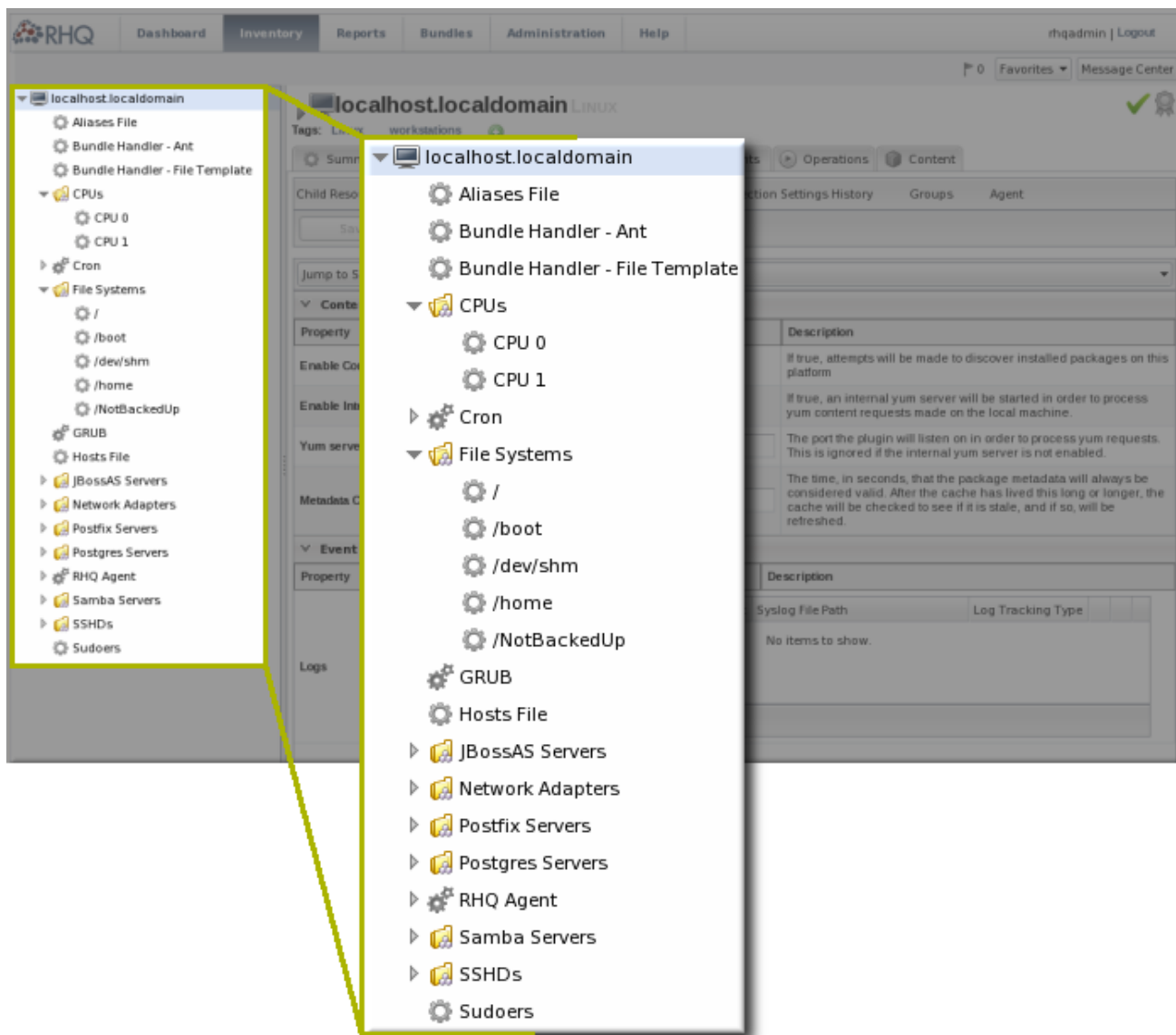


Figure 8. Resource Tree

Right-clicking any of the resources in the left navigation opens shortcuts to that entry's configuration.



NOTE

Resources have short names that are automatically assigned based on their type, instance or system name, or IP address. These names are used in the inventory and in the tree navigation.

The configuration area of a resource entry page (and other JBoss ON entities, like plug-ins and templates) has three information layers that provide all of the possible functionality and tasks available for that entry.

The entry's configuration page is tabbed according to each area that can be configured, and frequently has subtabs for additional configuration options and to show the history of that area (like fired alerts, previous content updates, or monitoring data).

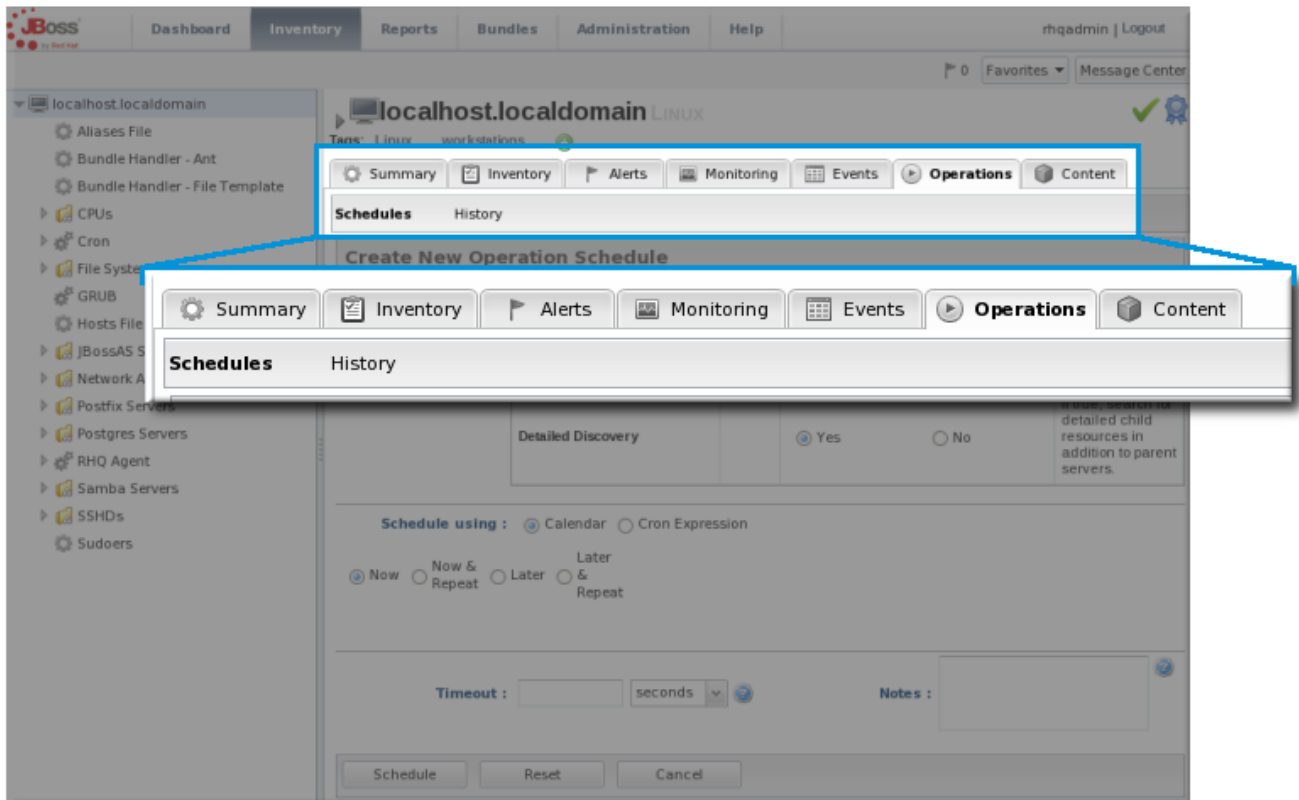


Figure 9. Tabs for a Resource Entry

The next section in the entry area shows lists of related configured entries for that task. For example, an **Operations** area will have a list of available operations in a table below the tabs. For **Inventory**, there is a list of configured child resources, while **Alerts** shows all of the configured alerts for that resource. All of those entries are listed in a table similar to the search results available in other parts of the JBoss ON UI.

Many elements are both a *details* page and an *edit* page. meaning that many fields are active automatically. This makes it possible to perform management tasks directly, without opening a separate page.

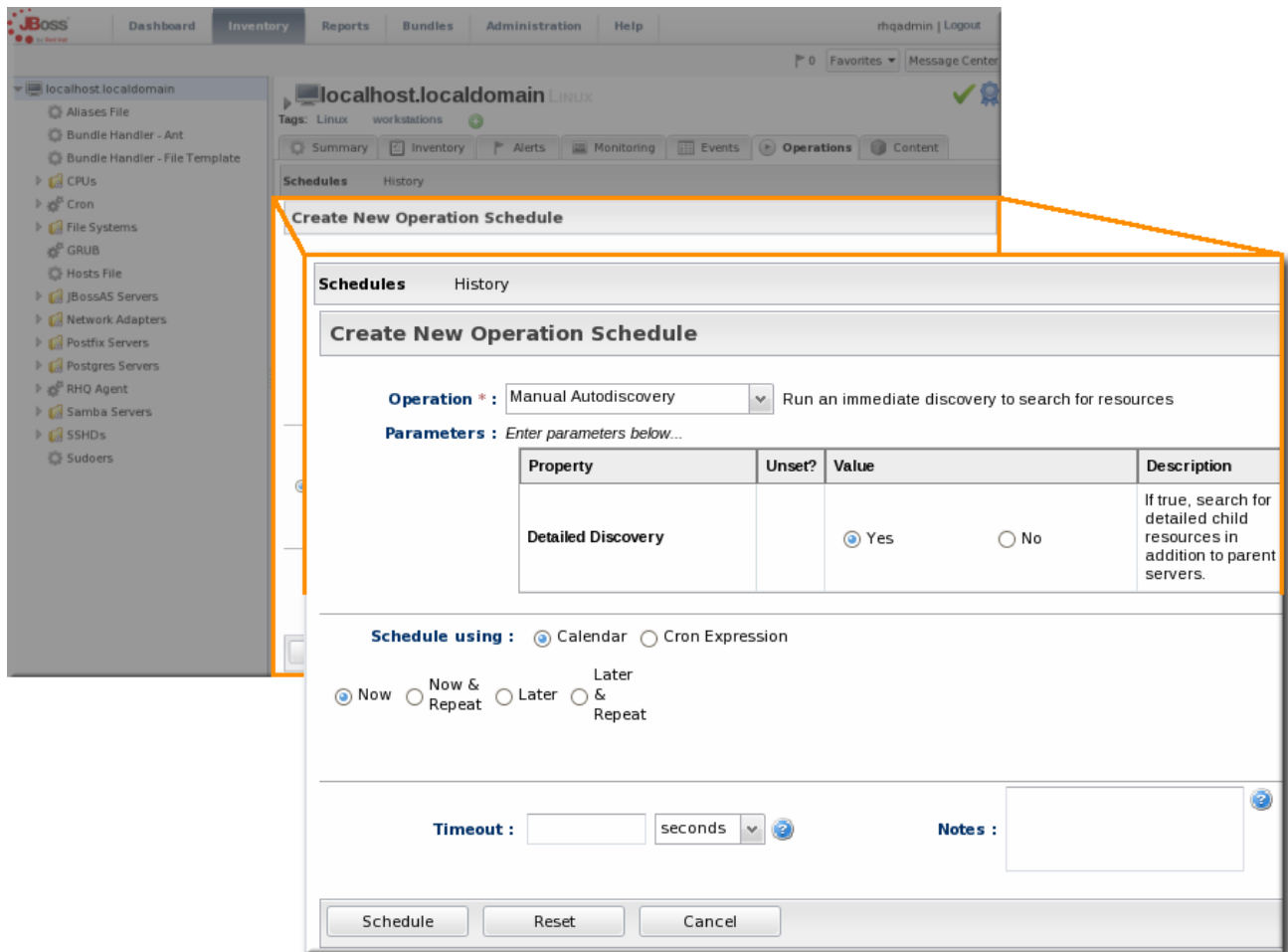


Figure 10. Editable Areas for a Resource Entry

1.3.6. Shortcuts in the UI

To the far right of the top menu is a small cluster of icons that provide very quick, targeted insight into JBoss ON.

- The Message Center shows all notifications that have been sent by the JBoss ON server. This includes alerts, configuration changes, changes to the inventory, or error messages for the server or UI.
- The alerts area shows the total number of current alerts for all resources in the inventory.
- The Favorites button can be used to navigate to selected resources and groups quickly, while the little blue ribbon on resource pages can be used to add that resource to the favorites list.
- The resource available is shown as a green check mark if the resource is available and a red X if the resource is down.

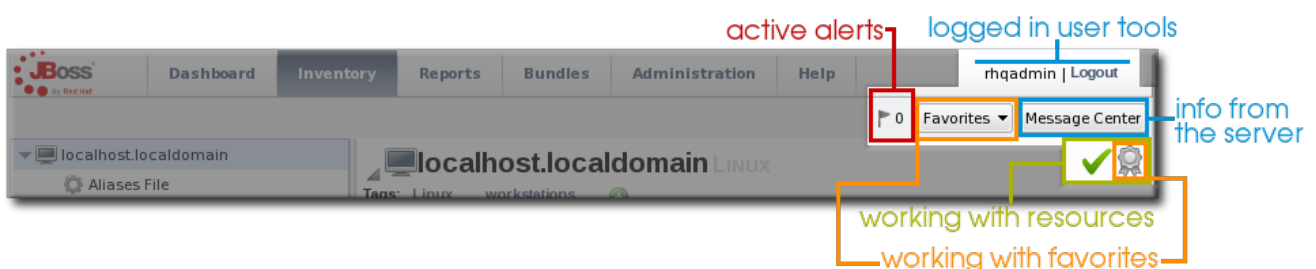


Figure 11. Shortcuts

1.4. Getting Notifications in the Message Center

The Message Center shows all of the messages that have been returned by the GUI for the current browser session. This includes any actions taken in the UI – like adding resources to the inventory, configuring resources, or uploading content – and it also includes any error messages that may have been returned during the session.

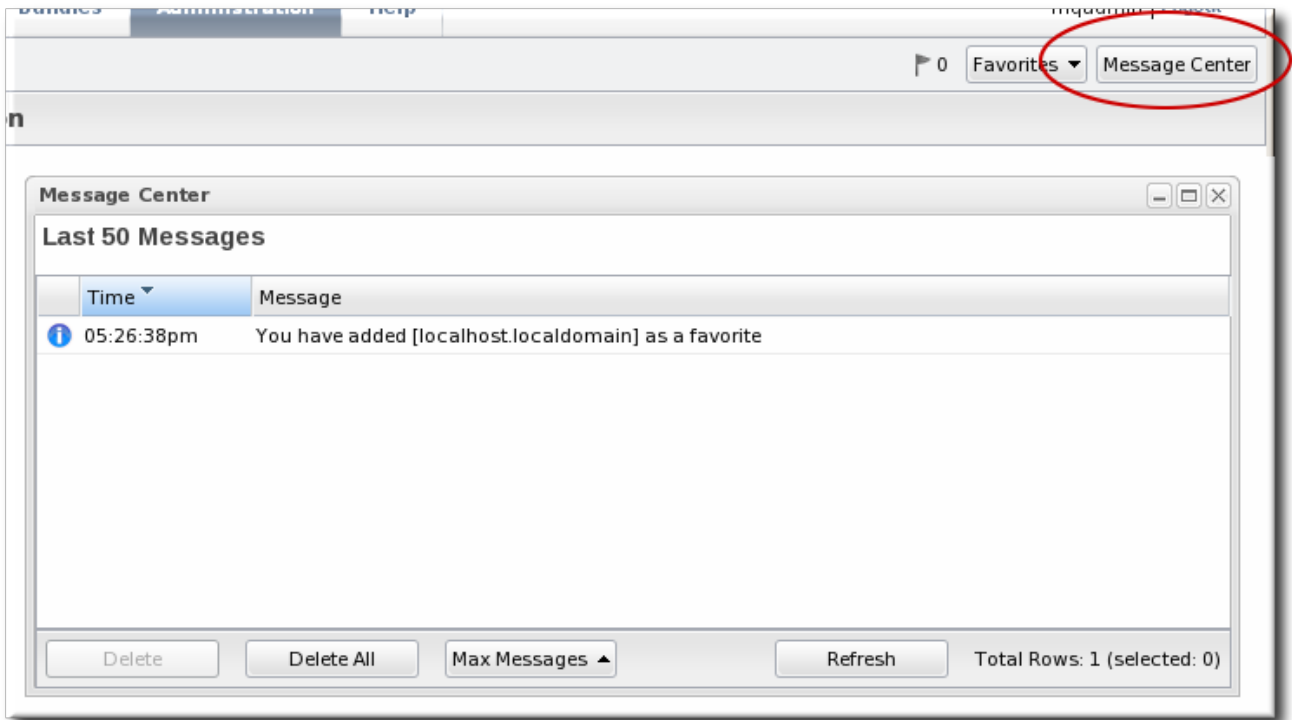


Figure 12. Message Center

1.5. Sorting and Changing Table Displays

Almost all of the information in JBoss ON is displayed in tables, from the resource inventory to the list of plug-ins for the agent. The SmartGWT UI has some versatility in how that table information is sorted and displayed.

A few tables use a very simple ascending/descending order based on the column being sorted, either numerically or alphabetically.

<input type="checkbox"/>	<u>Execution Time ▲ 1</u>	Type
<input type="checkbox"/>	4/13/11, 10:51:18 AM, EDT	OPERATION_MODE_CHANGE

Figure 13. Basic Table Sorting on the Partition Events List

Most areas in the UI allow a more complex method of displaying information. As with basic tables, simply clicking a column name will sort that column in ascending/descending order. However, advanced GWT tables also have an option to change the table layout and sort options, by clicking a menu arrow at the right of the column.

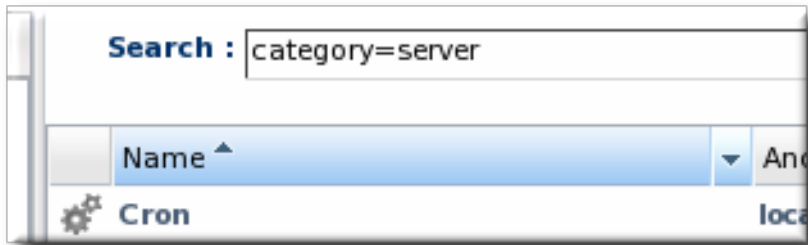


Figure 14. Basic Table Sorting on the Server Resources List

When a menu arrow is selected, the sort order for that column can be changed, or any other column. You can also change the column sizing and even the types of columns displayed. The options are generated dynamically, depending on what kind of entry is contained in the table.

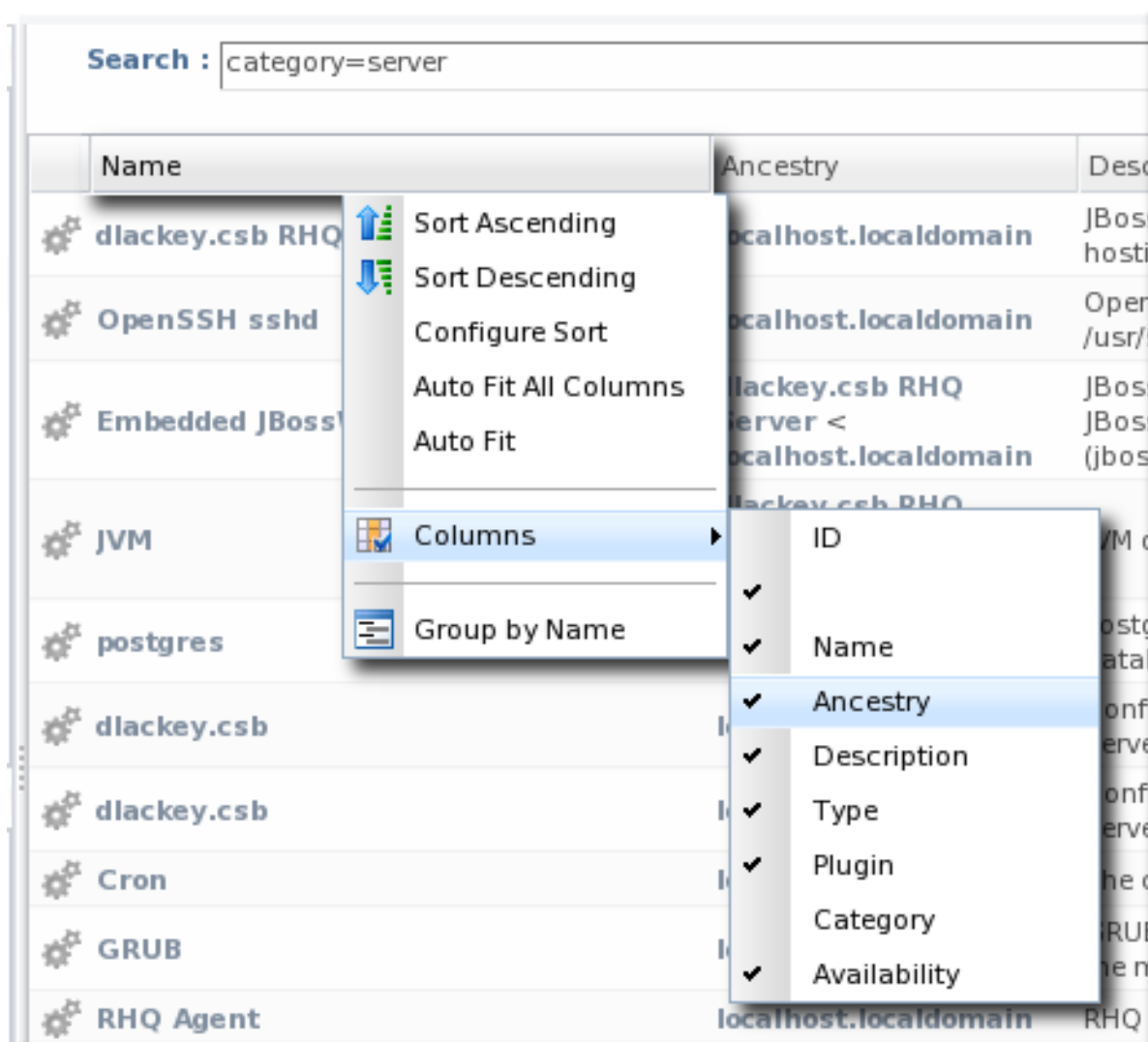


Figure 15. Advanced Table Sorting on the Server Resources List

The sort order can even be prioritized by specifying multiple criteria. For example, resources can be sorted by name, then by plug-in, then by ID. Since resources have standardized names, sorting by name or parent alone may not be specific enough to give a meaningful order to the entries; providing multiple, prioritized criteria can make the table display more accurate.

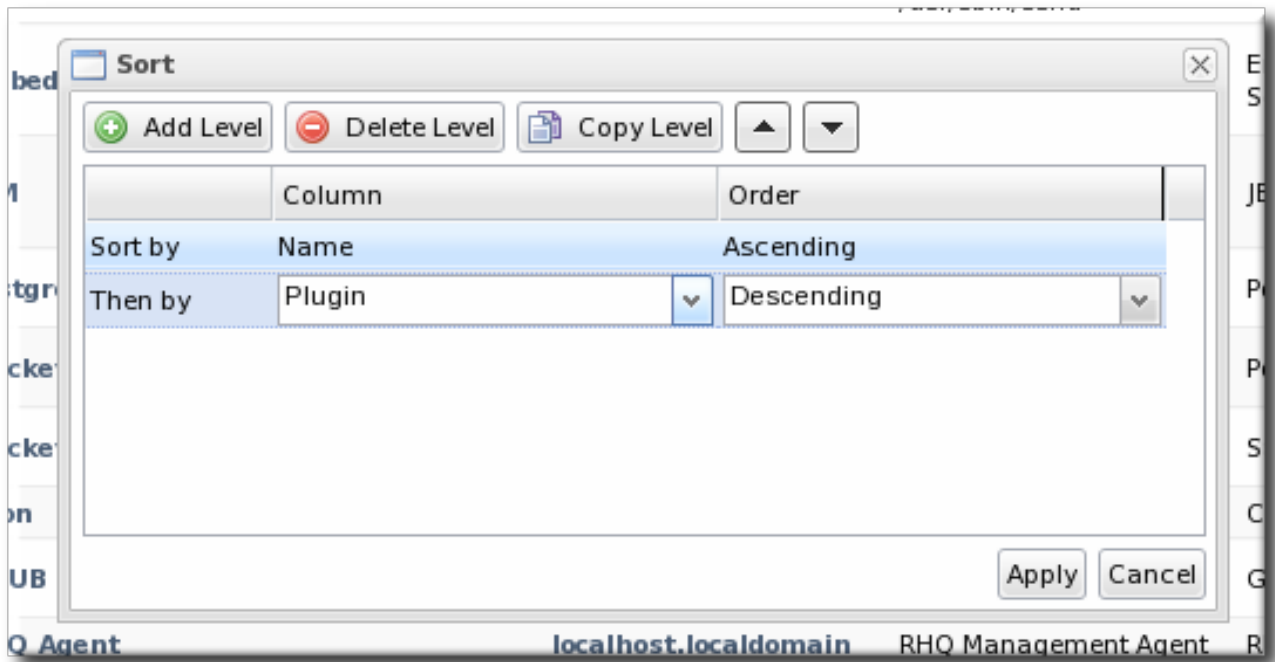
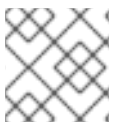


Figure 16. Changing the Sort Method

1.6. Customizing the Dashboard

The Dashboard is configurable. It is composed of individual portlets, and these portlets can be rearranged or independently refreshed through the icon menu displayed on each portlet. There can even be multiple Dashboards, with different portlets, which can be used to give different and specific views into JBoss ON and its resources.



NOTE

Dashboards are configured per user, not globally.

1.6.1. Editing Portlets

To move portlets within the Dashboard layout, use the arrows in the portlet tool bar. To get rid of a portlet in a current, click the minimize icon on the far left to collapse it or click the X icon on the far right to delete the portlet from the Dashboard entirely. Some types of portlet allow customization, which can be accessed by clicking the wrench icon.

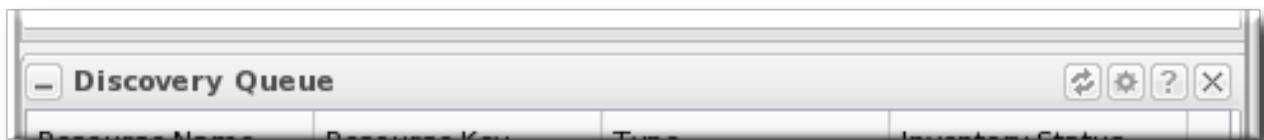


Figure 17. Portlet Icons

1.6.2. Adding and Editing Dashboards

The Dashboard page can actually contain multiple Dashboards, each with different portlets, column layouts, and refresh intervals. This makes it possible to get a logical grouping of information for a very fast assessment of the state of resources in JBoss ON. When multiple Dashboards are configured, they are displayed as tabs in the UI.

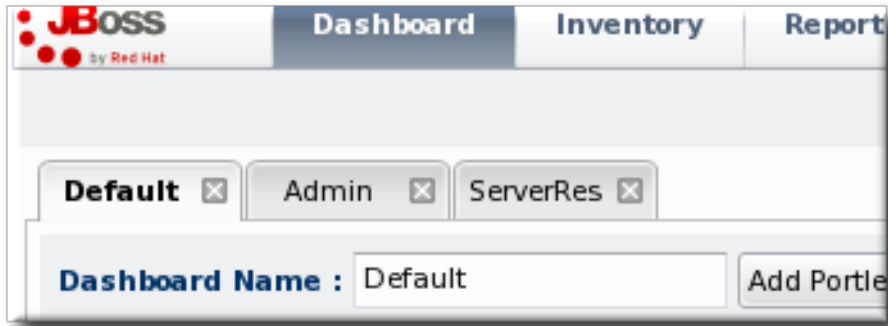
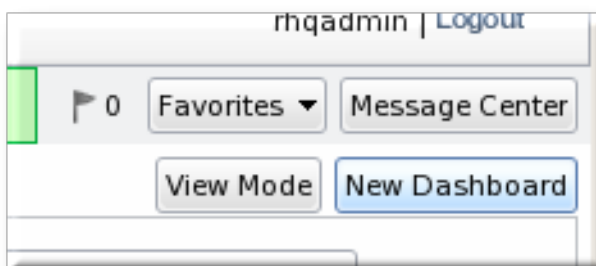


Figure 18. Tabbed Dashboards

To add a new Dashboard:

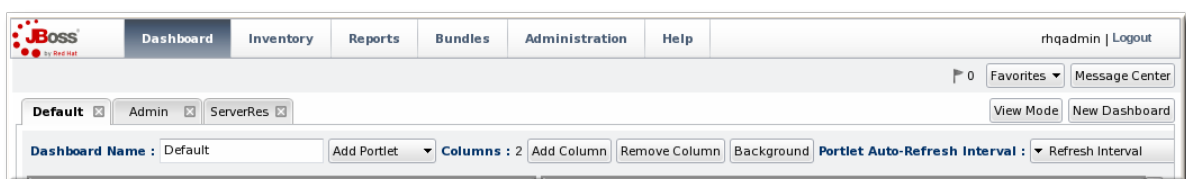
1. Click the **New Dashboard** button in the far right of the main Dashboard.



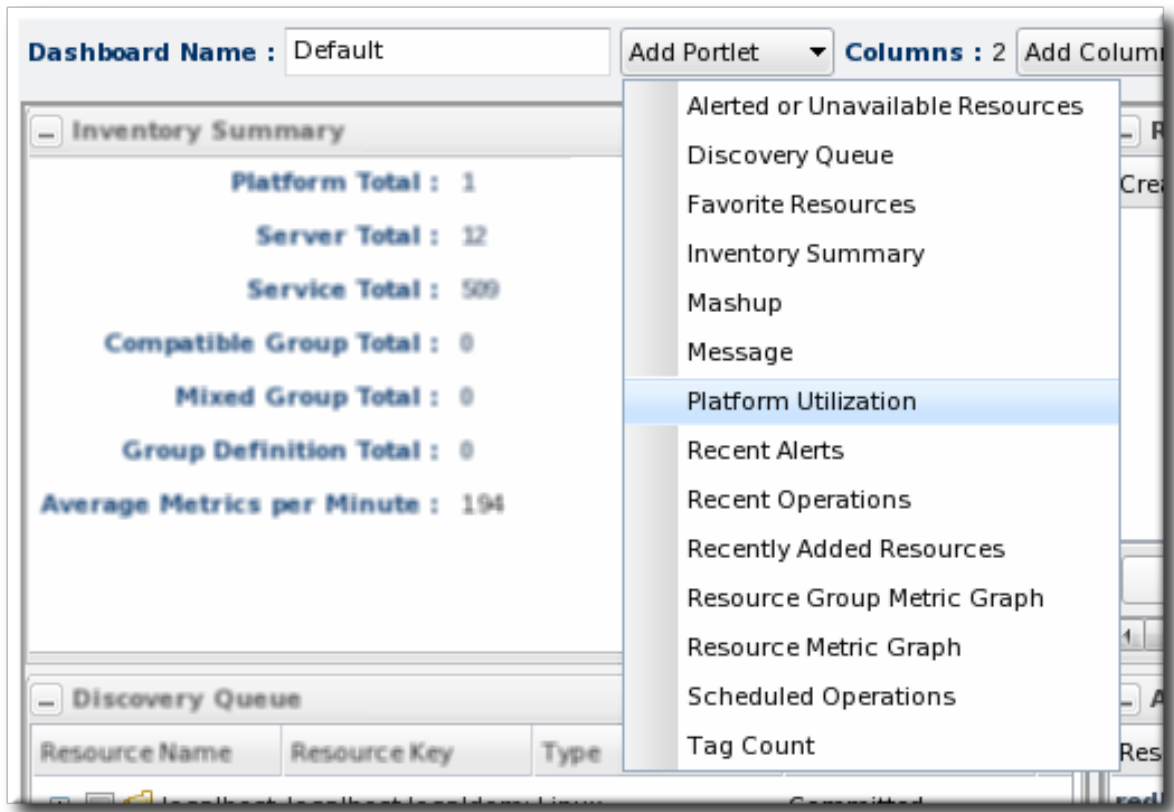
NOTE

The process of editing and adding Dashboards is very similar. The only difference is that to edit a Dashboard, you click the **Edit Mode** button.

2. The new Dashboard opens in the edit mode. Enter a name for the new Dashboard.



3. Add the desired portlets to the Dashboard. If necessary, change the number of columns to fit the number of portlets.



1.7. Setting Favorites

Using favorites makes it easy to navigate to resources that administrators need to access routinely for configuration updates, monitoring, or alerting.

Each resource has a small ribbon icon in the upper right corner of its details page. Clicking that icon automatically adds it to the resource favorites list.



Figure 19. Favorites Icon

The resource and group favorites are listed in the **Favorites** in the shortcuts on the right of the top menu. Clicking a resource on that list automatically opens its details page without having to search for the resource. Because multiple resources may share a name or some properties, the Favorites list includes a hover with more details about the resource so you can select the right one.

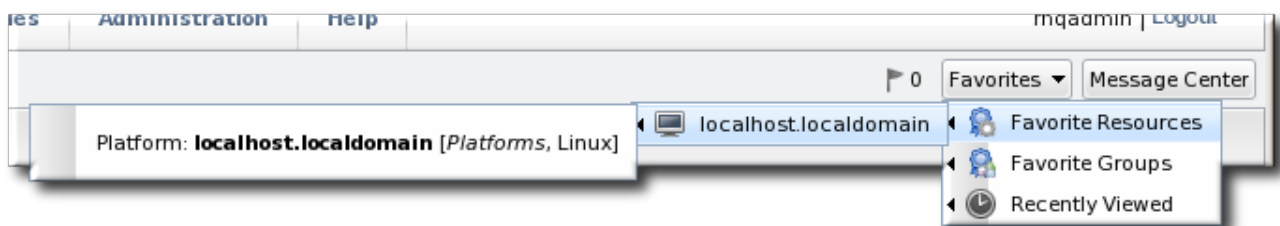


Figure 20. Favorites List

1.8. Deleting Entries

Resource-related entries can be deleted through the inventory browser or group browser. Most JBoss ON server configuration entries cannot be deleted. Only user-supplied elements, like plug-ins, content, roles, and users, can be deleted.

If an item can be deleted, then a delete button is available in the table list or details page for that item.

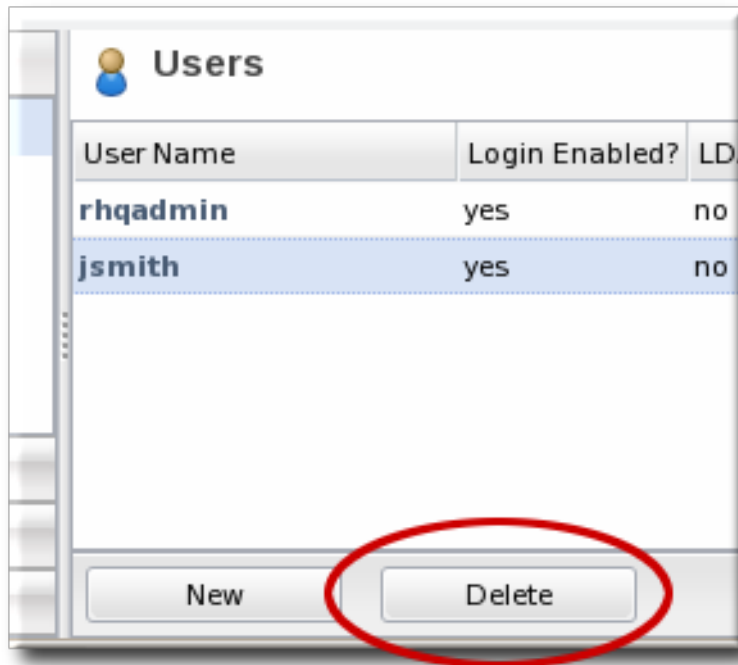


Figure 21. Delete Button in the Area Browser



NOTE

A user may have the right to *change* something, but that does not implicitly grant the right to *delete* something. For example, users with the configuration write permission can edit resource configuration and view configuration history and settings, but they cannot delete elements in the configuration history. Similar constraints are true for users with permission to create and edit operations and alerts – there is no right to delete elements in the resource history.

Deleting elements in the history requires the manage inventory permission.

2. DYNAMIC SEARCHES FOR RESOURCES AND GROUPS

The inventory area has a dynamic search to look for resources and groups.

The dynamic search is an additional tool that can help manage your JBoss ON resources. Dynamic searches in JBoss ON can be saved to provide fast and reproducible snapshots of your JBoss ON deployment that match criteria that are relevant to your infrastructure, a kind of quick report.

A dynamic search checks both resources and groups (recursively into group members, as well) much more effectively than either a subsystem views search or a quick search. A search can begin against a specific identifying attribute of a resource (such as its name, parent, type, or JBoss ON category) and then has rules that can set how the search handles the string. Multiple search parameters can be

strung together to make precise and complex searches. Dynamic searches can be saved and reused later so their results are reliably reproducible. (Section 2.2, “About the Dynamic Search Syntax” covers the details more.)

There are other aspects of dynamic searches like the autocomplete, hints, and highlight search strings that make it easier to use effectively than the limited substring and quick searches. These are covered in Section 2.1, “About Search Suggestions”.

2.1. About Search Suggestions

Dynamic searches are extremely powerful, past simply finding resources. Dynamic searches can run through values in a number of different resource traits, not only the resource name. Dynamic searches can even be saved, so they're repeatable and can be used as ad hoc reporting.

Dynamic searches are easy to use because of search suggestions. A drop-down menu for every search provides three different types of suggestions:

- Saved searches, which contain previous custom search strings and a count of resources which match that search
- Query searches, which provide prompts for available resource traits
- Text searches, which provide a list of resources based on some property in the resource which matches the text prompt

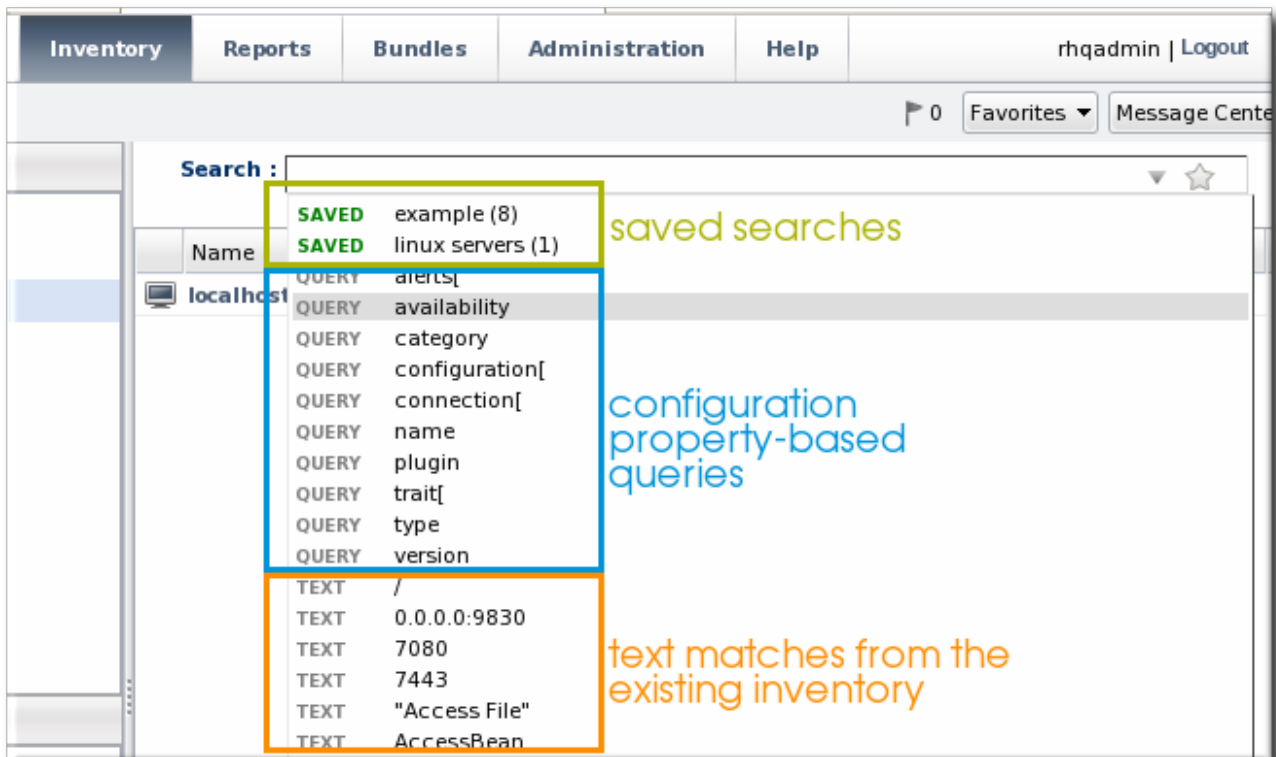


Figure 22. Types of Search Suggestions

When search terms are entered in the field, the matching substrings in possible matching resources are highlighted. By default, the suggestions can match any substring in the resource or in resource configuration traits. The suggestions can be limited to match the string at the beginning or end of the matching attribute using different operators (covered in Section 2.2, “About the Dynamic Search Syntax”).

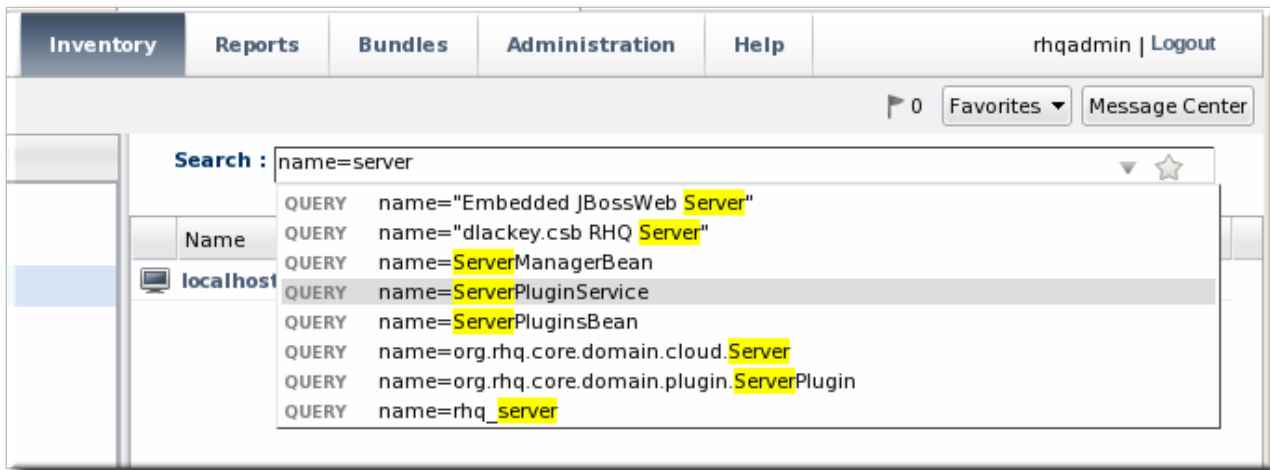


Figure 23. Highlighting Search Terms

2.2. About the Dynamic Search Syntax

JBoss ON has its own search syntax for dynamic searches. The syntax is supposed to be relatively simple while covering a wide array of search-able items and allowing different phrases to be coupled together.

The basic dynamic search matched whatever text is entered in the search box in a general substring search. The search can allow a more detailed and targeted syntax, in this form:

[search_area].[search_property] operator value operator additional_search

The *search_area* identifies what type of entry – resource or group – is being searched for. This is an optional value because the search area is implied by the location of the search; i.e., searching in the Resources area implies a resource search, so it's not necessary to include the `resource.` part of the search.

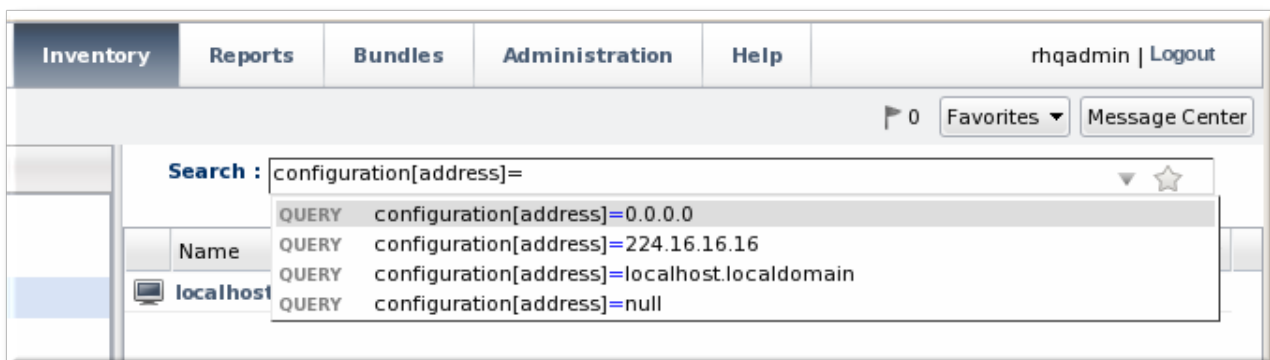


Figure 24. Searching by Resources Traits

2.2.1. Basic String Searches

The simplest search with the dynamic search is a substring search. The given search term can match any part or all of the returned value.

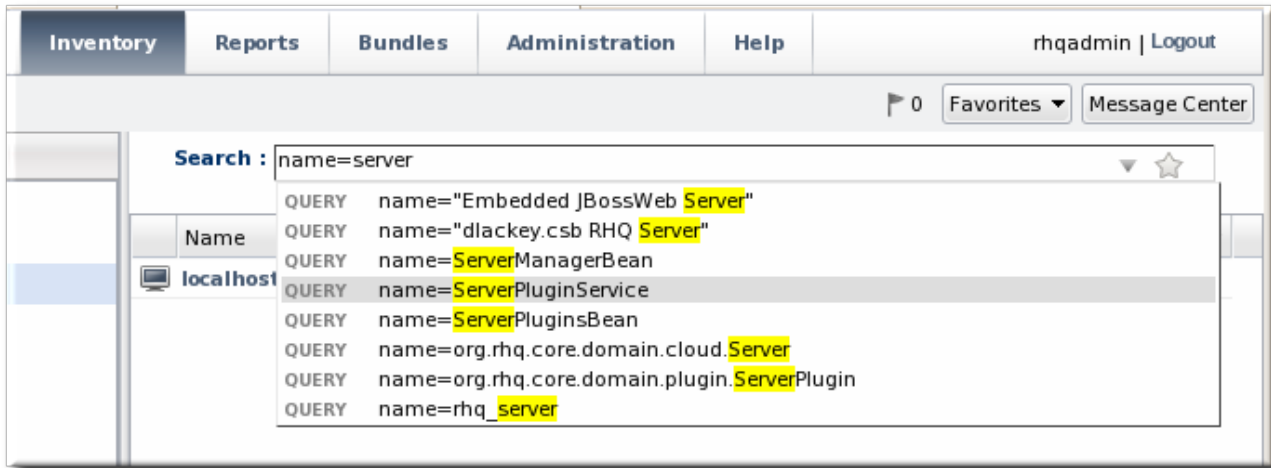
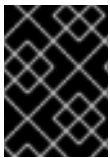


Figure 25. Matching the Search Term

The search term `agen` has search suggestions that match (case-insensitive) every resource that has that string anywhere in its name, at the beginning (**Agent Plugin Container**), middle (**RHQ Agent Launcher Script**), or end (`rhq_agent`). Likewise, the results include every resource with that string, regardless of where it appears in the resource entry or attribute.

When multiple words or terms are used together, the search treats them each as individual search terms in an implied AND search. (These complex searches are covered more in [Section 2.2.3, "Complex AND and OR Searches"](#).) However, there can be instances when a multi-word search term should be treated as a single search term, and, for that, the dynamic search allows quotation marks.



IMPORTANT

Dynamic searches do *not* support wildcard characters like asterisks (`*`) or regular expressions.

The search syntax can use either single (`'`) or double (`"`) quotation marks to enclose a search term that includes whitespace. (Obviously, search terms without whitespaces do not require quotes.) If a search term has a series of space-separated words, then each word is treated as a separate search term in an AND search. For example:

```
postgres table myexampletable
```

Using quotation marks tells the search to treat that as a literal phrase rather than individual search terms. As with other search terms, phrases can be strung together in a space-separated list:

```
"My Compatible Group"
'test box'
plugin=jboss 123.4.5.6
trait[partitionName]='my example group' server.example.com
```

If a search term contains double or single quotation marks in the name, then the other type of boundary character must be used. For example, this group name contains a single quotation mark (apostrophe) character:

```
name="Production's Main Group"
```

Make sure to use the same opening and closing term character (you cannot mix single and double

quotation marks.) Also, a single quotation mark is only treated as a search definition character when it occurs at the beginning of a search term. 'Hello world requires a closing single quote; **Production**'s does not.

A search can also look for a string where it occurs within a result. For this, the search allows *boundary characters* that set whether a string occurs at the beginning or end of a value or if it must be an exact match to the value.

The caret (^) character sets that a search term must appear at the beginning of the result string. For example:

```
resource.id=^100
```

This means that only resources with an ID which starts with 100 will be returned in the search.

Likewise, a string may occur only at the end of a value. The boundary character for an end value is a dollar sign (\$). For example:

```
script$
```

This returns any resource with any value that ends in *script*.

Using both a caret and a dollar sign, together, means that the matching result must exactly match the search term, with no additional characters.

The search characters in [Table 1, “String Operators”](#) limit the search string by setting where in the result value the string appears.

Table 1. String Operators

Operator	Description
<i>string</i>	The string can occur anywhere in the result string.
<i>^string</i>	The given string must appear at the beginning of the result value.
<i>string\$</i>	The given string must appear at the end of the result value.
<i>^string\$</i>	The result must be an exact match of the given string, with no leading or trailing characters.

**NOTE**

The dynamic search syntax treats null as an acceptable value for a search. Running a search which passes null will look for any resource or group with a null value for that property:

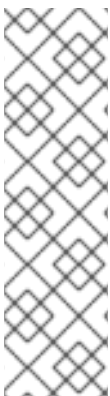
```
resource.trait[Database.startTime] = null
```

However, putting quotation marks around the term `null` will look for a resource or group with a value of the string `null`:

```
name = "null"
```

2.2.2. Property Searches

The search can be narrowed by looking for a specific value or type of attribute in the entry by using a search property. For example, looking for a resource with a CPU usage of 80% (`trait`) is different than looking for an entry with an ID that includes 80 (`id`). The available properties are listed in [Table 2, “Resource Search Contexts”](#) and [Table 3, “Group Search Contexts”](#).

**NOTE**

It's possible to search using group criteria in the resource search, and the reverse, by specifying the search area and the appropriate properties. For example, it's possible to do a search in the groups area to return the list of groups that a specific resource belongs to. This is done by explicitly passing the search context and search property. For example, in the **Groups** page, to list any group which contains a resource managed by the Postgres plug-in:

```
resource.type.plugin = Postgres
```

**IMPORTANT**

The parameter suggestions for `connection`, `configuration`, and `trait` use the *internal property names* for the property names (`connection[property_name]`) rather than the names used in the JBoss ON GUI.

Table 2. Resource Search Contexts

Property	Description
resource.id	The resource ID number assigned by JBoss ON.
resource.name	The resource name, which is displayed in the UI.
resource.version	The version number of the resource.
resource.type.plugin	The resource type, defined by the plug-in used to manage the resource.

Property	Description
resource.type.name	The resource type, by name.
resource.type.category	The resource type category (platform, server, or service).
resource.availability	The resource availability, either UP or DOWN.
resource.pluginConfiguration[<i>property-name</i>]	The value of any possible configuration entry in a plug-in.
resource.resourceConfiguration[<i>property-name</i>]	The value of any possible configuration entry in a resource.
resource.trait[<i>property-name</i>]	The value of any possible measurement trait for a resource.

There are slightly fewer search properties for groups, since groups have simpler entries than resources.

Table 3. Group Search Contexts

Property	Description
group.name	The name of the group.
group.plugin	For a compatible group, the plug-in which defines the resource type for this group.
group.type	For a compatible group, the resource type for this group.
group.category	The resource type category (platform, server, or service).
group.kind	The type of group, either mixed or compatible.
group.availability	The availability of resource in the group, either UP or DOWN.

The *operator* first refers to how the results should match the search string (*value*). This can require an exact match, every value but the one given in the search string. The *operator* then refers to how multiple search strings relate to each other (AND or OR); both explicit AND and OR statements and parenthetical statements are allowed. Complex searches are covered in [Section 2.2.3, “Complex AND and OR Searches”](#).

Table 4. Search String Operators

Operator	Description
=	Case-insensitive match.
==	Case-exact match.
!=	Case-insensitive negative match (meaning, the value is <i>not</i> the string).
!==	Case-exact negative match (meaning, the value is <i>not</i> the string).

2.2.3. Complex AND and OR Searches

The dynamic search bar assumes that each individual word is a search term (unless terms are defined using quotation marks). Implicitly, multi-word searches are treated as AND searches. For example:

```
postgres server myserver
```

This is treated as a series of AND terms:

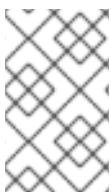
```
postgres AND server AND myserver
```

The dynamic search also allows OR searches, with terms separated by a pipe (|). For example:

```
postgres | jbossas
```

Both AND and OR searches can be entered, and complex searches can be written by stringing multiple search strings together. When there are both AND and OR search criteria, the AND terms are processed first. For example, this search term searches for both B and C, and then either A or B/C.

```
a | b c
```



NOTE

When there are both AND and OR terms used in a complex search, AND terms are given preference. However, terms in parenthesis are evaluated even before AND expressions, so parentheses can be used to override the natural search preference.

Search phrases can be nested to multiple levels using parentheses to group search terms. These parentheses can also be used to override the preferences for AND matches, forcing at least some OR expressions to be processed first. For example, this expression searches for the OR terms first, matching *a OR b* and *c OR d*, and then running an AND search on the results of the two OR searches:

```
(a | b) (c | d)
```

The results will contain several combinations of values: *a c*, *a d*, *b c*, and *b d*.

Multiple levels of nesting are allowed. For example, this expression requires *a* AND either *b OR c* AND *d*:

■

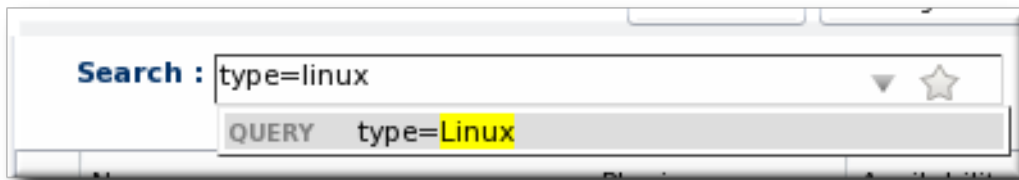
(a) (b | (c d))

The matching resources, then, can contain values matching *a c d* or *a b*.

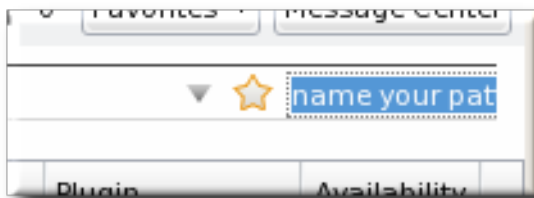
2.3. Saving, Reusing, and Deleting Dynamic Searches

Dynamic searches can be saved, which makes it much easier to reuse complex or common searches. To save a search:

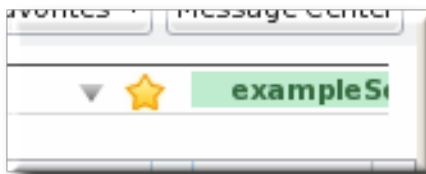
1. Run the search.



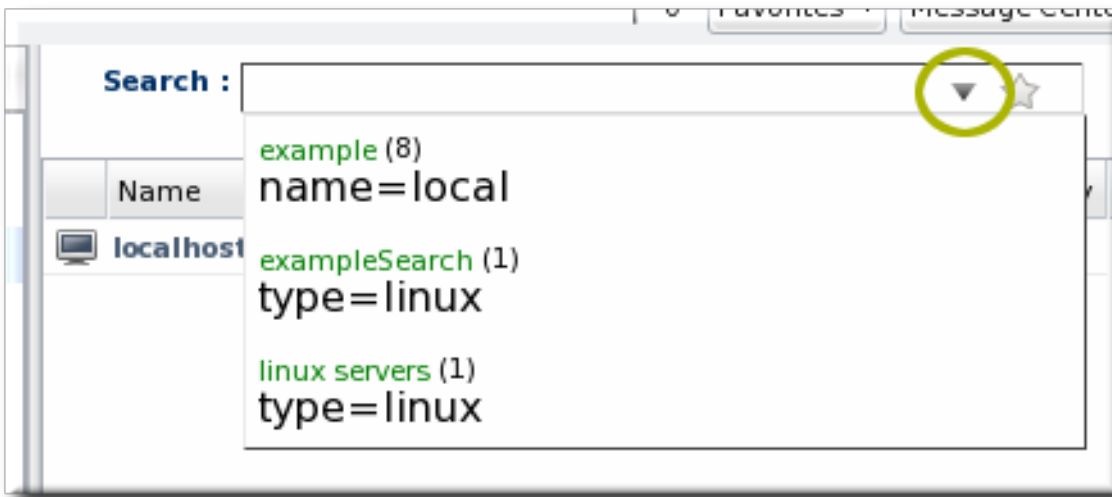
2. Click the star in the right of the search bar. When the field comes up, enter the name for the new search.



The search name is then displayed in green.



Saved searches are listed with other search results whenever the dynamic search criteria match any part of the saved search string and immediately whenever the dynamic search field is active, before search parameters are entered. The name of the search is shown in bright green. The drop-down option shows the string used in the saved search in black text beneath the name, to make it clear what the parameters for the search are.



To edit a saved search, select the search from the list of suggestions and then click the gold star. This deletes the search, but leaves the previous search settings in the search box. This allows the search parameters to be edited and then re-saved.

To delete a search, simply click the gold star or the trashcan icon by the search name when it is highlighted in the list. It is immediately removed from the saved searches list.



3. VIEWING AND EXPORTING REPORTS

The purpose of JBoss ON is to deliver information about the resources in your infrastructure. There are a lot of places in the UI where information is displayed for individual resources or for defined groups.

The **Reports** main tab has a list of predefined searches and views into different areas for *all* resources, not just a subset.

3.1. Types of Reports

All reports show a list of information that spans everything in the JBoss ON inventory. Reports are broken into two categories, one for JBoss ON server subsystem areas and one for different inventory counts.

Subsystem reports are related to *functionality* in JBoss ON, for different aspects of monitoring, alerting, drift, and configuration. Most subsystem reports have some corollary to a resource-level chart, with much the same information displayed. Subsystem reports include additional columns to list the resource name and the resource ancestry (parent and grandparent resources) to disambiguate each resource, since names are not unique.

Inventory reports give counts. These reports usually begin with a breakdown by resource type, with additional resource lists available as "subreports."

Resource Type	Plugin	Cat	Version	Count
Linux	Platforms	Linux	2.6.18-164.15.1.el5	1
Cron	Cron			1
GRUB	GRUB		1.0	1
Embedded Tomcat Server	JBossAS		2.0.1.GA	1
JBoss AS JVM	JBossAS		1.6.0_17	1
JBossAS Server	JBossAS		4.2.3.GA	1
JBossCacheSubsystem	JBossCache		1.0	1
SSHD	OpenSSH		1.0	1
Postfix Server	Postfix			1

Figure 26. Inventory Summary Report

Table 5. Types of Reports

Report Name	Description	Has Filters?
Subsystem Reports		
Suspect Metrics	Lists any metrics outside the established baselines for a given resource. All suspect metrics for all resources are listed, but the baselines which mark the metric may be different for each resource, even different between resources of the same type.	No
Configuration History	Lists all configuration changes, for all resources. Version numbers are incremented globally, not per resource. The configuration history shows the version number for the change, the date it was submitted and completed, its status, and the type of change (individual or through a group).	No
Recent Operations	Lists all operations for all resources, by date that the operation was submitted (not necessarily run), the operation type, and its status.	Yes

Report Name	Description	Has Filters?
Recent Alerts	Lists every fired alert for all resources, with the name of the resource, the alert definition which was fired, and the alerting condition.	Yes
Alert Definitions	Lists all configured alert definitions, for all resources, with their priority and whether they are enabled.	No
Recent Drift	Contains a list of all snapshots, for all resources and drift definitions.	Yes
Inventory Reports		
Inventory Summary	Contains a complete list of resources currently in the inventory, broken down by resource type and version number.	No
Platform Utilization	Shows the current CPU percentage, actual used memory, and swap space.	No
Drift Compliance	Shows a list of all resource types which support drift and then shows how many drift definitions are configured and whether the group is compliant. Clicking on a resource type shows the list of resources configured for drift and their individual compliance status.	No

3.2. Exporting Report Data to CSV

The **Reports** tab collects information that is not easily accessible in other parts of the GUI or even the CLI, without complex scripting. The information from any report can be exported to CSV simply by clicking the **Export** button.

	A	B	C	D	E
1	Resource Type	Plugin	Category	Version	Count
2	Linux	Platforms	Platform	Linux 2.6.32-2	1
3	Cron	Cron	Server		1
4	GRUB	GRUB	Server	1	1
5	JBossAS Server	JBossAS	Server	AS 4.2.3.GA	1
6	SSHD	OpenSSH	Server	1	1
7	Postfix Server	Postfix	Server		1
8	Postgres Server	Postgres	Server	8.4.9	1
9	RHQ Agent	RHQAgent	Server	4.4.0-SNAPSH	1
10	Samba Server	Samba	Server		1
11	Aliases File	Aliases	Service		1
12	Ant Bundle Handler	AntBundlePl	Service	4.4.0-SNAPSH	1
13	Cron Tab	Cron	Service		4
14	File Template Bundle H	FileTemplate	Service	4.4.0-SNAPSH	1

Figure 27. Exported Inventory Summary

Only the information displayed in the report, as displayed in the report, is exported to CSV. If there is a certain sort order applied to the report or if a filter is used to limit the displayed entries, that sort order and that filter are preserved in the exported report CSV file.

Date Submitted	Operation	Requestor	Status	Resource	Ancestry
May 8, 2012 10:46:18 PM	Execute Availability Scan	rhqadmin	Success	RHQ Agent	server.example.com
May 8, 2012 10:23:56 PM	Execute Availability Scan	rhqadmin	Success	RHQ Agent	server.example.com
May 7, 2012 2:42:14 PM	View Process List	rhqadmin	Success	server.exam...	
May 7, 2012 11:47:12 AM	Install RHQ user	rhqadmin	Success	EAP Domain Controller (127.0.0.1:99...	server.example.com

Figure 28. Report with Date Filters

4. MANAGING THE RESOURCE INVENTORY

The *inventory* in JBoss ON is the repository that contains all of the servers and applications that are managed or monitored by JBoss Operations Network. The inventory tells JBoss Operations Network which resources it can manage.

Once in the inventory, resources can be organized in several different ways. Resources can be grouped automatically by their type in autogroups, resources can be added manually to user-defined groups, and they can be added manually to another resource as a child.

This section covers the process of identifying and importing resources through discovery, adding children, and managing groups.

4.1. About the Inventory: Resources

The JBoss ON *inventory* is the central list of every managed resource that is recognized by the JBoss ON server.

4.1.1. Managed Resources: Platforms, Servers, and Services

Each JBoss ON agent periodically scans the platform where it's installed to check for services and servers. That is the *discovery* process. When a potential resource is discovered, then it is listed in the discovery scan results, and, from there, an administrator can choose whether it should be managed by JBoss ON. If a resource should be managed, then it must be *imported* into the JBoss ON server's inventory; otherwise, it can be ignored.

There are three categories of resources in JBoss ON:

1. Platforms (operating systems)
2. Servers
3. Services

The resource hierarchy in the JBoss ON inventory mimics how programs and processes are physically structured on a platform. The highest level is the platform. The platform can have both servers and services as children. Likewise, servers can have both other servers and services as children, while services can have only other services as children within the inventory.

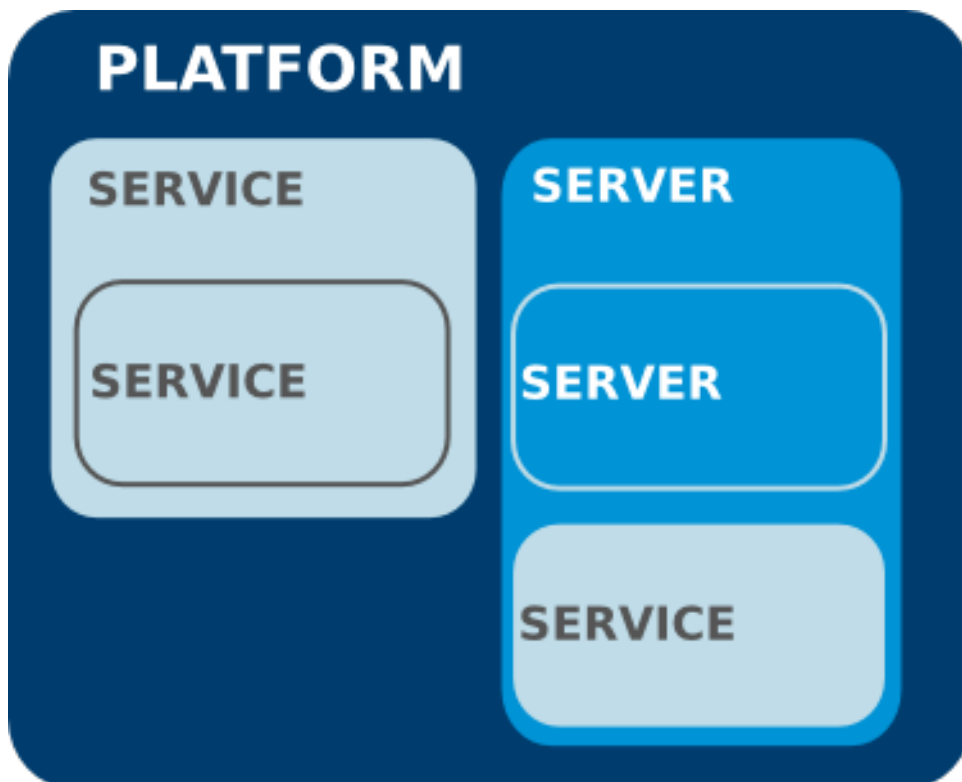


Figure 29. An Example Resource Hierarchy

A handful of rules govern the relationships between resources in inventory:

- A resource can only have one parent.
- A server can be a child of a platform (such as JBoss AS on Linux) or another server (such as Tomcat embedded in JBoss AS).
- A service can be a child of a platform, a server (such as the JMS queue on JBoss AS), or another service (e.g. a table inside a database).
- Platforms, servers, and services can have many children services.

JBoss ON can manage many different types of resources; each managed resource has a corresponding agent plug-in which defines things like the available monitoring metrics, operations, and supported versions for the resource. [Table 6, “Default Resources Supported in JBoss ON”](#) lists the default resource types that are supported in JBoss ON.



NOTE

Additional resources can be added by writing custom plug-ins for the resource type.

Table 6. Default Resources Supported in JBoss ON

Managed Platform			
AIX	FreeBSD	HP-UX	Java
Linux	Mac OS X	Solaris	Windows
Managed Servers			
Apache HTTP Server	JBoss BRMS (Drools)	JMS Manager Service (HornetQ)	Hibernate Services
Host Server	HTTPService	IIS Server	JBoss EAP/AS 4 and 5
JBoss Cache Services 2.x and 3.x	JBoss ESB and ESB 5	JMX Server	mod_cluster
Oracle	PostgreSQL	JBoss Developer Studio (ModeShape)	Tomcat Server
JBoss ON Agent	JBoss ON Server		

4.1.2. Content-Backed Resources

For application servers, there is a close conceptual link between a child resource and content which is deployed on the server. For example, EARs and WARs are both child resources of application servers and are versioned content which can be stored in a repository, selectively deployed, and reverted.

A *content-backed resource* is treated as a resource in that it has a place in the inventory hierarchy, can have operations run against it, and can have metrics collected for it. However, it is also managed as a software package, with bits that are uploaded and stored in the JBoss ON content system and maintained within a repository.

Content-backed resources can be manually added as children or, if deployed outside JBoss ON, can be detected in a package discovery scan (which runs every 24 hours by default). These child resources can also be created and updated by deploying content from the repositories in JBoss ON.

For more information on managing content-backed resources, see ["Deploying Applications and Content"](#) and ["How to ... Manage JBoss Servers with JBoss ON"](#).

4.1.3. Resources in the Inventory Used by JBoss ON

Some resources are automatically added to platforms to enable certain JBoss ON-specific functionality. For example, an Ant bundle handler resource is added to platforms as a child service to allow the agent to identify and process Ant recipes. ^[1] Without that Ant bundle handler resource, the JBoss ON agent cannot perform provisioning on that platform. Administrators do not have to interact directly with JBoss ON-specific child resources once they are in the inventory, but these child resources must be present for JBoss ON functionality to work. Because these children are required by JBoss ON, they are imported automatically with the platform.

Other resources can be added to the inventory for the JBoss AS server used by the JBoss ON server, the JBoss ON agent, and JBoss elements associated with the agent and server such as the agent JVM, JBoss Cache, and the agent launch script and `rhq-agent-env.sh` script. Adding these resources to the JBoss ON inventory allows JBoss ON to monitor and manage all of the agents and servers in the deployment.

4.2. Interactions with System Users for Agents and Resources

The agent runs as a specific system user, and so do servers such as JBoss and Apache which are managed by JBoss ON. The general assumption with many of the agent management tasks, including discovery, is that the agent user is the same as the resource user. If the users are different, then that can have an impact on how resources can be discovered and managed.

The common types of servers which JBoss ON manages are:

- JBoss EAP servers
- PostgreSQL databases
- Tomcat servers
- Apache servers
- Generic JVMs

For some management operations initiated by the JBoss ON agent, the agent system user is never even involved. For example, the JBoss EAP plug-in connects to the EAP instance using authentication mechanisms managed by JBoss EAP itself, so no system ACLs or user permissions are required. As long as the user can access the JBoss EAP instance, everything works.

Table 7. Cheat Sheet for Agent and Resource Users

Resource	User Information
PostgreSQL	No effect for monitoring and discovery. The agent user must have read/write permissions to the PostgreSQL configuration file for configuration viewing and editing.

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

4.2.1. The Agent User

There is a general assumption that the agent runs as the same user as the managed resources, and this is the cleanest option for configuration.

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

4.2.2. Agent Users and Discovery

An agent discovers a resource by searching for certain common properties, such as PIDs and processes or start scripts.

It does not necessarily matter whether the agent has superior privileges as the resource user.

For most resources, the agent simply requires read access to that resource's configuration. For resources like Apache and Postgres, as long as the agent can read the resource configuration, the resources can be discovered.

For some other resources, the agent user has to have very specific permissions:

- For JBoss EAP resources, the agent must have read permissions to the `run.jar` file, plus execute and search permissions for every directory in the path to the `run.jar` file.
- Tomcat servers can only be discovered if the JBoss ON agent and the Tomcat server are running as the same user. Even if the agent is running as root, the Tomcat server cannot be discovered if it is running as a different user than the agent.
- If a JVM or JMX server is running with JMX remoting, then it can be discovered if the agent is running as a different user. However, if it is running with using the attach API, it has to be running as the same user as the agent for the resource to be discovered.

4.2.3. Users and Management Tasks

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

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

The key thing to determine is what tasks need to be performed and who needs to perform that operation, based on limits on the resource or the operating system for permissions or authorization.

For some actions – discovery, deploying applications, or creating child resources – setting system ACLs that grant the agent user permission are sufficient.

For running operations or executing scripts, it may be necessary to run the task as a user other than the agent user. This can be done using **sudo**.

Whatever method, the goal is to grant the JBoss ON user all of the required system permissions necessary to carry out the operations.

4.2.4. Using sudo with JBoss ON Operations

The time to use **sudo** is for long-running operations, such as starting a service or a process, or for scripts which are owned by a resource user. The user which executes the script should be the same as the resource user because that user already has the proper authorization and permissions.

The user can really be the same, or the JBoss ON user can be granted **sudo** rights to the given command.

When elevating the agent user's permissions, two things must be true:

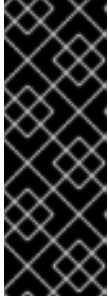
- There can be no required interaction from the user, including no password prompts.
- It should be possible for the agent to pass variables to the script.

To set up **sudo** for resource scripts:

1. Grant the JBoss ON agent user **sudo** rights to the specific script or command. For example, to run a script as the **jbosadmin** user:

```
[root@server ~]# visudo
jbosson-agent    hostname=(jbosadmin) NOPASSWD: /opt/jboss-
eap/jboss-as/bin/*myScript*.sh
```

Using the **NOPASSWD** option runs the command without prompting for a password.



IMPORTANT

JBoss ON passes command-line arguments with the start script when it starts an EAP instance. This can be done either by including the full command-line script (including arguments) in the `sudoers` entry or by using the `sudo -u user` command in a wrapper script or a script prefix.

The second option has a simpler `sudoers` entry

2. Create or edit a wrapper script to use. Instead of invoking the resource's script directly, invoke the wrapper script which uses `sudo` to run the script.



NOTE

For the EAP start script, it is possible to set a script prefix in the connection settings, instead of creating a separate wrapper script:

```
/usr/bin/sudo -u jbosson-agent
```

For example, for a start script wrapper, `start-myScript.sh`:

```
#!/bin/sh
# start-myScript.sh
# Helper script to execute start-myConfig.sh as the user jbosson-agent
#
sudo -u jbosson-agent /opt/jboss-eap/jboss-as/bin/start-myConfig.sh
```

3. Create the start script, with any arguments or settings to pass with the `run.sh` script. For example, for `start-myConfig.sh`:

```
nohup ./run.sh -c MyConfig -b jonagent-host 2>&1> jboss-MyConfig.out
&
```

4.3. Discovering Resources

Before any application or platform can be managed by JBoss ON, it must be imported into the inventory. There are different ways of adding resources to the inventory, depending on how the resource was discovered.

4.3.1. Finding New Resources: Discovery

When an agent is installed and every time it starts up, it *scans* the platform, and all applications on it, for any servers, services, or other items which can be included into the inventory. The process of finding potential resources is called *discovery*.

There are different scans for each type of resource: platform, server, and service. High level scans for servers and platforms are initiated by the agent every 15 minutes. A service scan detects lower-level services that are running in servers that have already been imported into the inventory. These scans run by default every 24 hours. Both of these intervals are configurable in the JBoss ON agent

configuration. The agent always runs a scan for new resources when it starts up, and then periodically at its configured intervals. JBoss ON agents send information about the platform and servers it discovers back to the JBoss ON server.

A server must be imported into the inventory before any of its child processes, servers, or services can be detected by the discovery scan.

When a platform is imported into the inventory, several of its child servers and services are imported automatically as well. This includes resources that are vital to the platform (like CPU, network adapters, and filesystems) as well as resources that are used by the JBoss ON server itself (such as the Ant bundle handler resource, which is used by the provisioning subsystem).

Although discovery is run automatically by the agent, discovery can also be initiated manually to capture infrastructure changes immediately.

4.3.2. Running Discovery Scans Manually

Discovery scans are run automatically by the agent to identify new resources as they are added to a platform. Server scans are run every 15 minutes and service scans every 24 hours. (Additionally, the agent runs a full discovery scan when it starts up.) New resources can be added between the discovery scans, so administrators can initiate a *manual* discovery scan apart from the scheduled discovery scan.

The simplest way to initiate a discovery scan is to run the agent's `discovery` command at the agent command prompt:

1. Click the **Inventory** tab in the top menu.
2. Open the **Servers - Top Level Resources** link on the left, and select the agent resource.
3. Open the **Operations** tab for the agent.
4. In the **Schedules** subtab, click the **New** button.
5. Select the **Manual Discovery** operation from the drop-down menu, and select whether to run a detailed discovery (servers and services) or a simple discovery (servers only).

The screenshot shows the 'Create New Operation Schedule' form in the JBoss ON interface. The 'Operation' dropdown is set to 'Manual Autodiscovery'. The 'Parameters' section contains a table with a 'Detailed Discovery' property and radio buttons for 'Yes' and 'No', with 'No' selected.

Property	Unset?	Value	Description
Detailed Discovery		<input type="radio"/> Yes <input checked="" type="radio"/> No	If true, search for detailed child resources in addition to parent servers.

6. In the **Schedule** area, select the radio button to run the operation immediately.

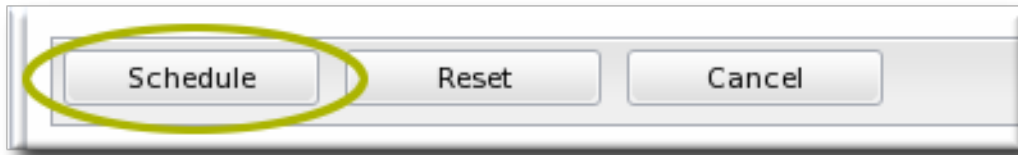
Schedule using : Calendar Cron Expression

Now Now & Repeat Later Later & Repeat

Timeout : seconds

Notes :

7. Click the **Schedule** button to set up the operation.

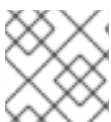


4.3.3. Importing Resources from the Discovery Queue

1. Click the **Inventory** tab in the top menu.
2. In the **Resources** menu on the left, select **Discovery Queue**.
3. Select the checkbox of the resources to be imported. Selecting a parent resource (such as a platform) gives the option to automatically import all of its children, too.
4. Click the **Import** button at the bottom of the UI.

4.3.4. Ignoring Discovered Resources

When the JBoss ON agent discovers an application or service which is to be ignored from the inventory, the server can be instructed to ignore those resources in the discovery queue.



NOTE

A resource can only be ignored if its parent is *already* added to the inventory.

1. Select **Inventory** from the top menu.
2. Select the **Discovery Queue** item under the **Resources** menu on the left side of the screen.
3. Select the checkbox of the resource to be ignored. Selecting a parent resource automatically selects all of its children.
4. Click the **Ignore** button at the bottom of the page.



NOTE

It is not possible to ignore a platform. If a platform should not be in the inventory, do not run an agent on that machine.

4.4. Importing New Resources Manually

Discovery scans are run on a defined schedule. There may be an instance where you add a new server or service on a platform and want to add it immediately to the JBoss ON inventory, before the next scheduled discovery run. It is possible to add that new child resource manually by importing it into the inventory of the parent resource – without waiting for the next discovery scan.



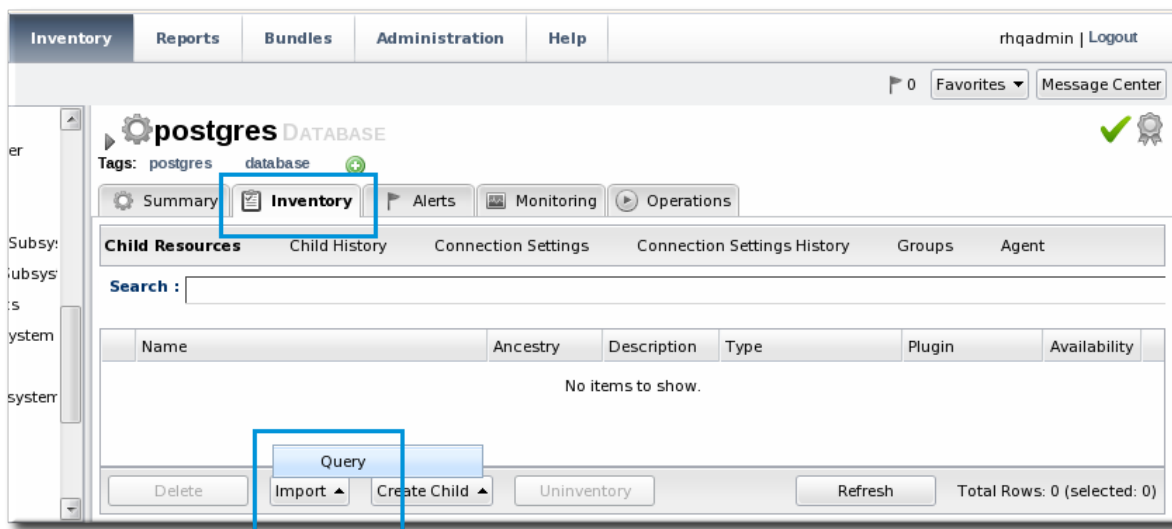
NOTE

The parent resource must be in an available state in order to import a child resource.

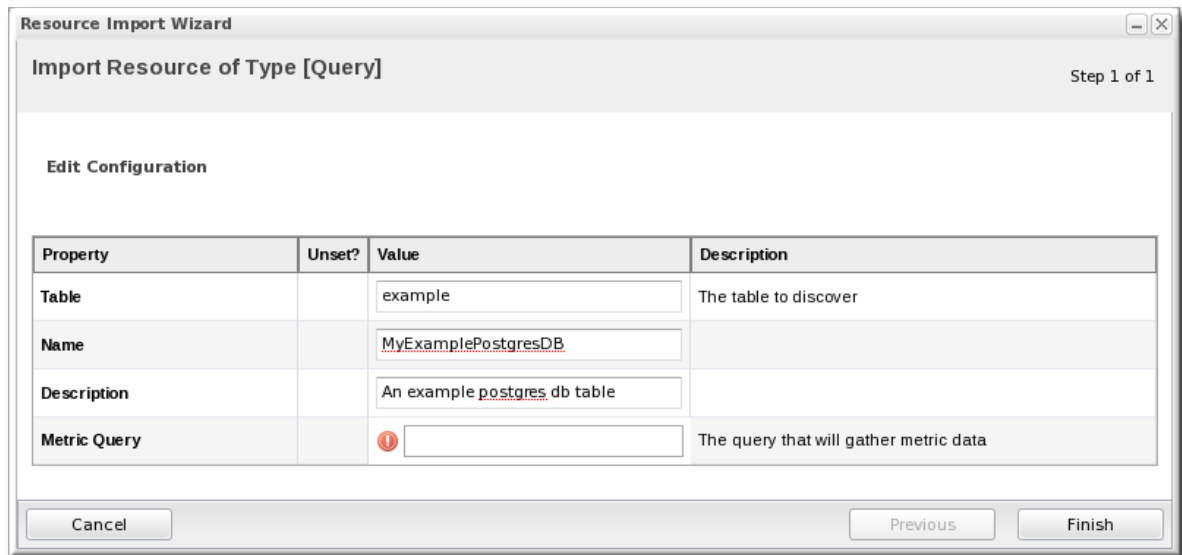
1. Click the **Inventory** tab in the top menu.
2. Search for the parent resource of the new resource.

[Section 2, “Dynamic Searches for Resources and Groups”](#) has information on searching for resources using dynamic searches.

3. Click the **Inventory** tab of the parent resource.
4. Click the **Import** button in the bottom of the **Inventory** tab, and select the type of child resource. The selection menu lists the possible types of child resources for that parent.



5. Fill in the properties to identify and connect to the new resource. Each resource type in the system has a different set of required properties.



4.5. Setting up a JVM for Discovery

The Generic JMX Plug-in detects Java processes. This is a very simple plug-in; aside from the ability to exclude some types of JVMs, the plug-in detects *any* properly-configured Java process and imports it as a JMX server. For a generic JMX server, all of its subsystems – logging, threads, memory, and others – are also imported as children of the JMX server.

All of this background information is covered in the JMX resource documentation [in the Resource Reference: Monitoring, Operation, and Configuration Options](#). Writing a custom plugin is covered in the [in Writing Custom Plug-ins](#)

A JVM resource itself has to be configured in a certain way for JBoss ON to be able to discover the resource to add it to the inventory.

4.5.1. Required JVM Configuration for Discovery

The agent discovers and subsequently manages a resource by identifying the resource based on certain parameters and then connecting to the agent over those discovered settings. For a Java process to be discovered using the Generic JMX Plug-in, it has to be configured to be connected to in one of two ways:

- Sun JMX remoting is enabled, with a port system property specified in the command line.

```
-Dcom.sun.management.jmxremote.port=12345 com.xyz.MyAppMain
```

- A Sun/Oracle-compatible Java process is accessible through the `com.sun.tools.attach` API, and the resource key is specified as a system property in the command line.

```
-Dorg.rhq.resourceKey=KEY com.xyz.MyAppMain
```

4.5.2. Excluding Java Processes from Discovery

The Generic JMX Plug-in is designed to be extended so that custom plug-ins can be based off it to detect and manage specific types of Java processes. For example, the JBoss ON agent, as a Java process, would normally be detected by the Generic JMX Plug-in, but it is discovered by the Agent Plug-in, which trumps the Generic JMX Plug-in. Similarly, JBoss EAP and Tomcat servers are also discovered by server-specific plug-ins, not the generic plug-in.

Resource which can be discovered by another plug-in should be excluded from the Generic JMX Server discovery, or there could be (spurious) conflict errors recorded in the agent log.

The Java processes to exclude from the discovery scan are defined in the JBoss ON agent configuration. There is a Java option for the agent, `rhq.jmxplugin.process-filters`, which lists strings to ignore specifically from the Generic JMX Plug-in discovery scan. If a Java process contains any of the strings in the filter, it is excluded from discovery as a JMX server^[2].

The default agent configuration filters out the classes for the agent, the JBoss ON server, and Tomcat servers:

```
RHQ_AGENT_ADDITIONAL_JAVA_OPTS="-Drhq.jmxplugin.process-
filters=org.rhq.enterprise.agent.AgentMain,org.jboss.Main,catalina.startup
.Bootstrap"
```

To add excludes filters:

1. Open the agent configuration file.

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

2. In the `RHQ_AGENT_ADDITIONAL_JAVA_OPTS` link, add the string to exclude to the `rhq.jmxplugin.process-filters` option. This can be the classname or any other identifying string which is in the command line for the given process.

For example:

```
RHQ_AGENT_ADDITIONAL_JAVA_OPTS="-Drhq.jmxplugin.process-
filters=org.rhq.enterprise.agent.AgentMain,org.jboss.Main,catalina.s
tartup.Bootstrap,com.abc.OtherAppMain"
```

The `rhq.jmxplugin.process-filters` value is a comma-separated list of strings.

3. Restart the agent with the `--config` option to load the new configuration.

```
[jboss-on-agent@server ~]$ agentRoot/rhq-agent/bin/rhq-agent.sh --
config
```

4.5.3. Manually Importing a JVM Resource

As [Section 4.5.1, “Required JVM Configuration for Discovery”](#) describes, there are only two connection configurations that can be used for a Java process for it to be discovered automatically by the agent. Specifically, those two connection settings both use a Sun management API, for remoting or for attach.

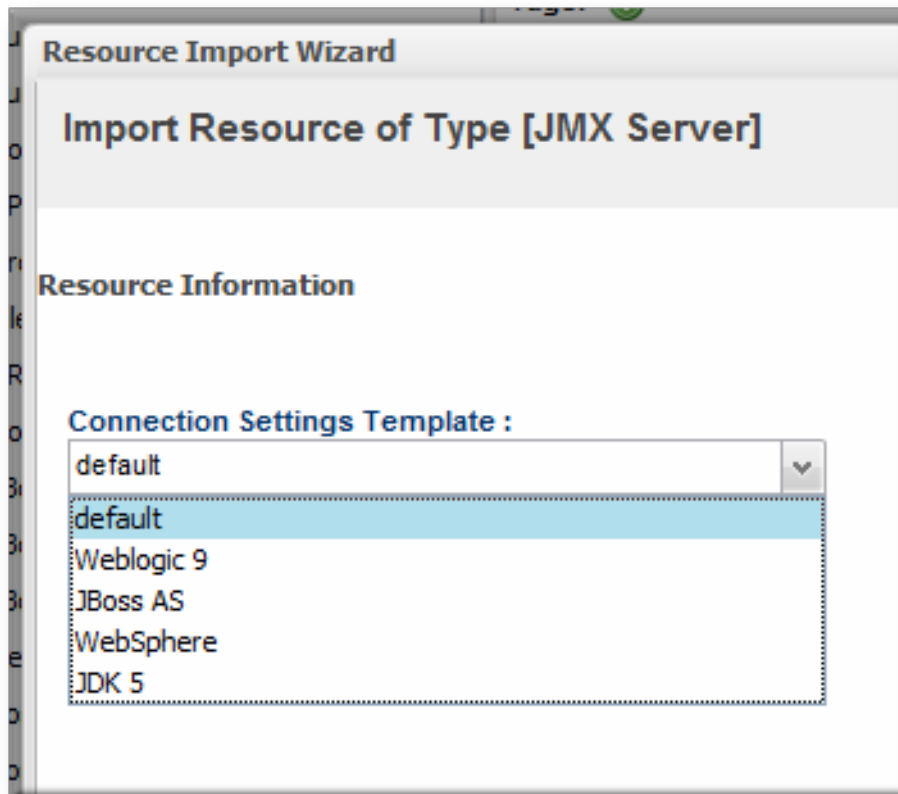
Any Java process can be imported into the inventory manually, even if it does not use the expected configuration for autodiscovery, as long as it enables JMX remoting in a supported form, meaning Sun or IBM remoting.



NOTE

The JVM instance has to be running for the resource to be discovered and imported.

1. Open the platform resource, and create the child as described in [Section 4.4, “Importing New Resources Manually”](#).
2. Select the type of JVM. The default JVM covers a lot of different types, including Tomcat servers, local VMs, and different IBM JVMs.



3. Fill in the connection information for the JVM. This varies depending on the JVM type, but it includes options like a URL and port, directory paths for client libraries, directory paths for classes, and login credentials.

Resource Import Wizard
Import Resource of Type [JMX Server] Step 2 of 2

Deployment Options

Property	Unset?	Value	Description
Type		Tomcat	The EMS connection type for this JMX Server
Connector Address	<input type="checkbox"/>	//tomcat.example.com:10080	The connection url in the form of a JMXServiceURL - this should only be set if the JVM has JMX Remoting enabled
Install URI	<input type="checkbox"/>	/etc/custom/tomcat	The installation path for the selected server type which will be used to find client libraries (if appropriate)
Principal	<input type="checkbox"/>	jvadmin	The login principal/username
Credentials	<input type="checkbox"/>	••••••••	The login credentials/password
Additional Class Path Entries	<input type="checkbox"/>	/etc/mbeans/*.jar	Comma-separated list of directories and filenames that contain resources and classes needed to communicate with the JMX Server and its MBeans. If you specify 'some/directory/*.jar', all jars found in the given directory will be added.
Command Line	<input checked="" type="checkbox"/>		the command line of the JVM at the time it was discovered - only used by JVMs with type Local; if the command line of the JVM changes, this property's value will need to be updated accordingly in order for RHQ to connect to the JVM

Timeout: seconds

4. Click the **Finish** to import the instance.

4.6. Configuring Tomcat/EWS Servers for Discovery (Windows)

Tomcat servers are discovered automatically on Linux and Unix systems, but they require additional configuration before they can be discovered on Windows systems.

1. Run **regedit**.
2. Navigate to Java preferences key for the Tomcat server, **HKEY_LOCAL_MACHINE\SOFTWARE\Apache Software Foundation\Procrun2.0\Tomcat Ver#\Parameters\Java**.
3. Edit the **Options** attribute, and add these parameters:

```
-Dcom.sun.management.jmxremote.port=9876
-Dcom.sun.management.jmxremote.ssl=false
-Dcom.sun.management.jmxremote.authenticate=false
```

4. Restart the Tomcat service.

After a few minutes, the Tomcat instance should show up in the Discovery Queue.

4.7. Creating Child Resources

JBoss ON can create certain types of children resources for parent resources. For example, a Postgres server can allow JBoss ON to create Postgres users. Not every allowed child resource type for a resource can be created through the JBoss ON UI; these children are usually limited to resource types that can be configured simply and remotely, such as scripts, WAR/EAR files, and server users.

**NOTE**

The parent resource must be available to add a child resource.

**NOTE**

It can take several minutes for the new child resource to be added and visible in the JBoss ON inventory because the new resource has to be created on the local system and then discovered by the agent. If the discovery scan is running when the resource is created, then it may take until the next discovery scan to be detected.

1. Click the **Inventory** tab in the top menu.
2. Search for the parent resource of the new resource.

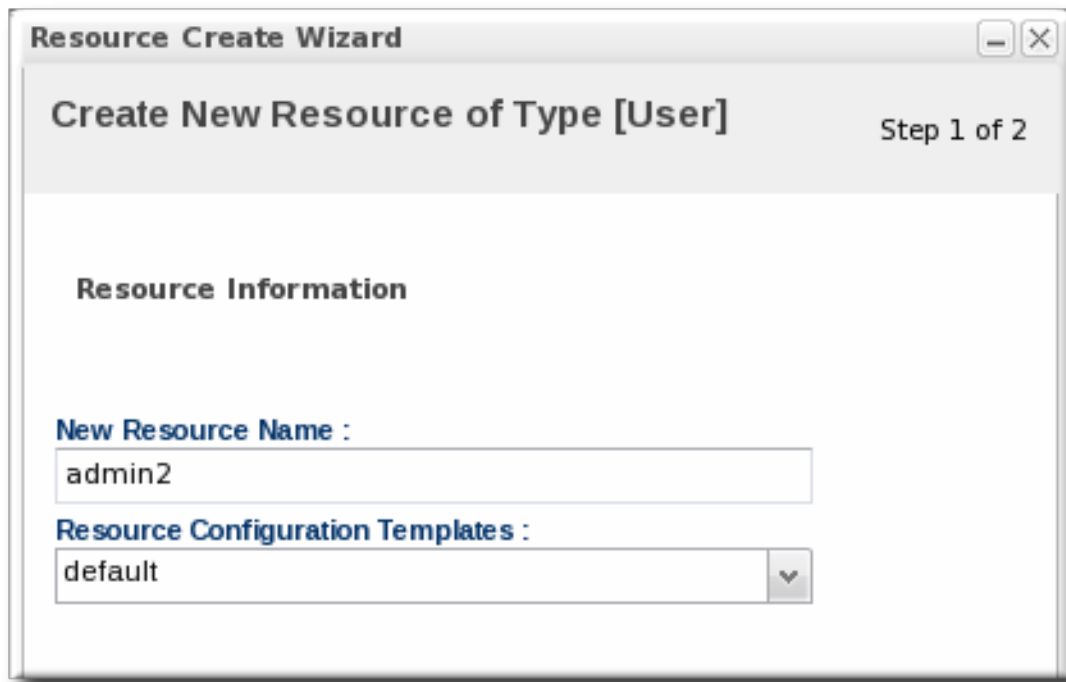
[Section 2, “Dynamic Searches for Resources and Groups”](#) has information on searching for resources using dynamic searches.

3. Click the **Inventory** tab of the parent resource.
4. Click the **Create Child** button in the bottom of the **Inventory** tab, and select the type of child resource. The selection menu lists the possible types of child resources for that parent.

The screenshot shows the JBoss ON web interface for a 'postgres' resource. The 'Inventory' tab is selected and highlighted with a yellow box. Below the 'Inventory' tab, there is a table of child resources. At the bottom of the interface, the 'Create Child' button is highlighted with a yellow box, and a dropdown menu is open showing 'User' as the selected option.

Name	Ancestry	Description	Type	Plugin	Availability
rhq	postgres < Linux Server 1	The rhq Postgres Database Instance	Database	Postgres	✓
postgres	postgres < Linux Server 1	A Postgres user	User	Postgres	✓
rhqadmin	postgres < Linux Server 1	A Postgres user	User	Postgres	✓

5. Give the name and description for the new resource.



Resource Create Wizard Step 1 of 2

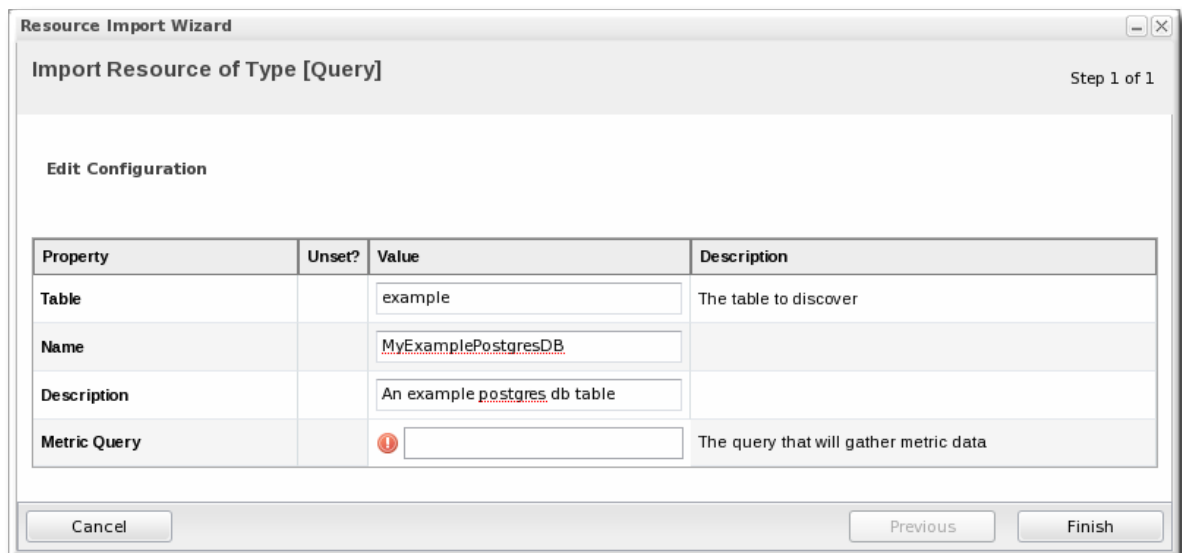
Create New Resource of Type [User]

Resource Information

New Resource Name :

Resource Configuration Templates :

6. Fill in the properties to identify and connect to the new resource. Each resource type in the system has a different set of required properties.



Resource Import Wizard Step 1 of 1

Import Resource of Type [Query]

Edit Configuration

Property	Unset?	Value	Description
Table		<input type="text" value="example"/>	The table to discover
Name		<input type="text" value="MyExamplePostgresDB"/>	
Description		<input type="text" value="An example postgres db table"/>	
Metric Query		<input type="text" value=""/>	The query that will gather metric data

Buttons: Cancel, Previous, Finish

4.8. Viewing and Editing Resource Information

Every resource has details about the server or service that can be viewed, such as its name, description, and version. (The specific information is different for each resource type.) These details are usually hidden when viewing the resource, but they can be viewed by clicking the arrow by the resource name to expand the details area.

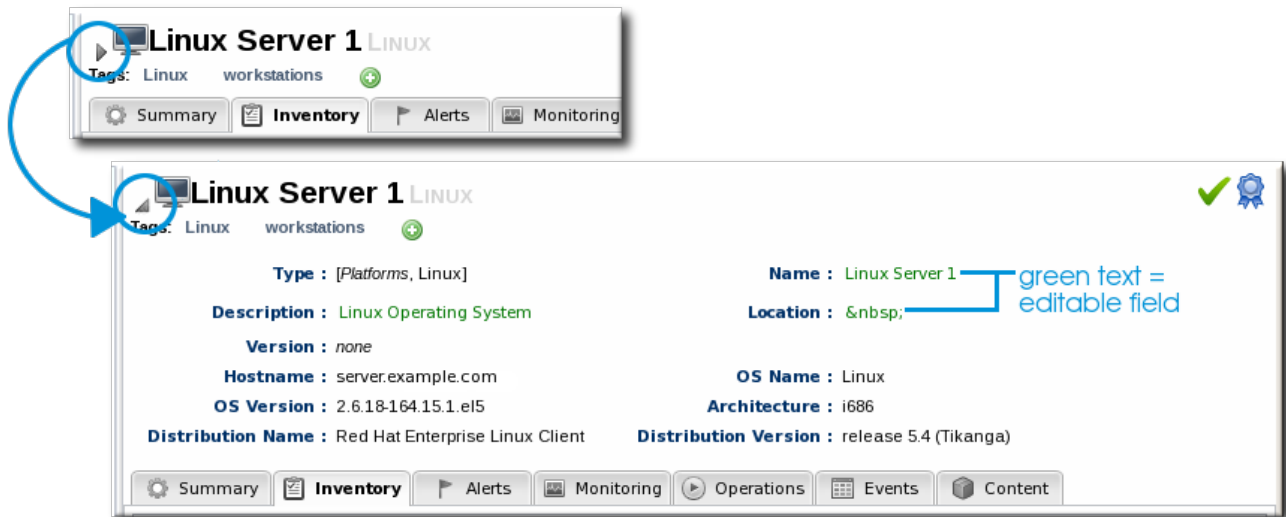
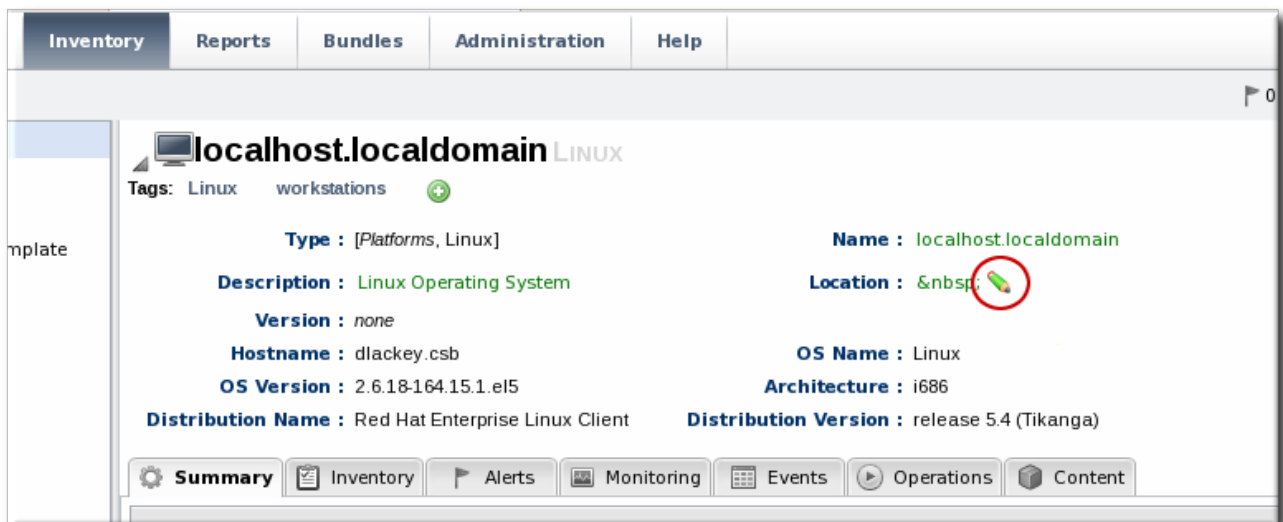


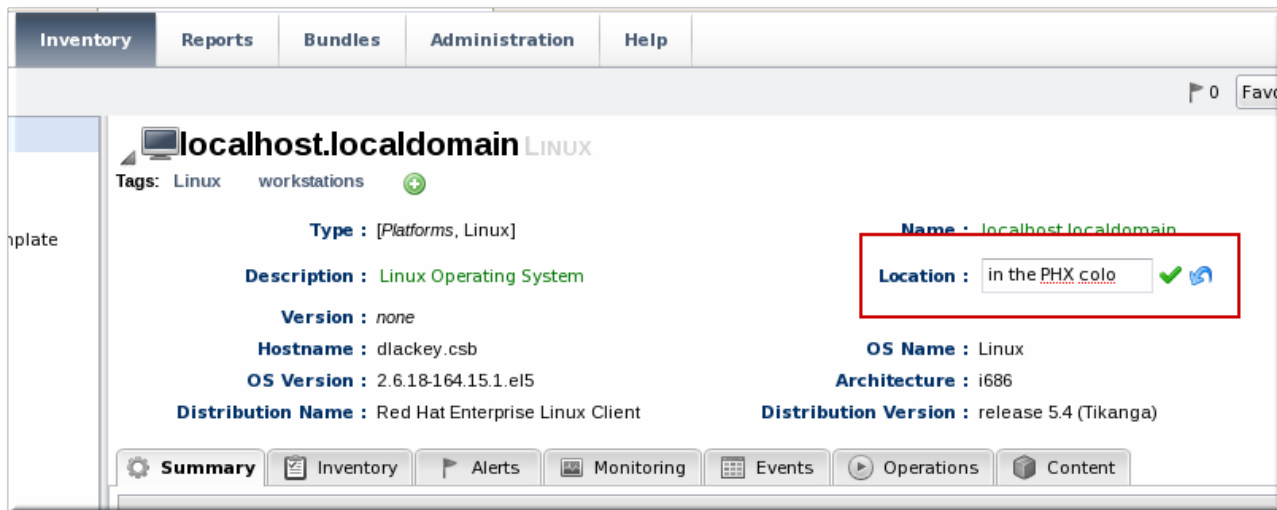
Figure 30. Expanding Resource Entry Details

Any fields with green text can be edited. This allows administrators to use more specific or useful information in areas that are supplied by the agent discovery, like the resource name, or to add information, like a description or, for example, a platform's physical location.

To edit a field, hover the cursor over the name and click the pencil icon that appears.



When the edits are made, click the green check mark to save the changes.



4.9. Managing Connection Settings

Connection settings define how an agent can connect to a resource. These are settings defined in the agent plug-in, so they are different for each resource type.

At a minimum, connection settings provide a way for the agent to connect to a resource, such as a port number, directory path, or user credentials.



NOTE

Often, if a resource shows down availability even when it is running, it is a problem with the connection settings. The agent may not have information it requires, such as a username or new port number, that it requires to connect to the resource. Since the agent cannot connect to the resource, it assumes it is down.

Connection settings can also provide configuration for other controls defined in the plug-in descriptor, like paths or options to use with scripts for operations, log file locations for event monitoring, and configuration files to allow for resource configuration editing.

To edit the connection settings:

1. Click the **Inventory** tab in the top menu.
2. Search for the resource.

[Section 2, “Dynamic Searches for Resources and Groups”](#) has information on searching for resources using dynamic searches.

3. Click the name of the resource to go to its entry page.
4. Open the **Inventory** tab for the resource, and click the **Connection Settings** subtab.
5. Change the connection information for the resource.

EAP Domain Controller (127.0.0.1:9990) JBossAS7 HOST ✔

CONTROLLER

Tags: +

Summary Inventory Alerts Monitoring Events Operations Configuration

Child Resources Child History **Connection Settings** Connection Settings History Groups Agent

Save

Jump to Section

General Properties

Property	Unset?	Value	Description
Hostname	<input type="checkbox"/>	10.16.65.172	Host name of the domain API
Port	<input type="checkbox"/>	9990	Port of the domain API
User	<input type="checkbox"/>	rhqadmin	Management user for a secured Host controller
Password	<input type="checkbox"/>	Password for the management user
Domain Configuration	<input type="checkbox"/>	domain.xml	Running configuration (domain part)
Host Configuration	<input type="checkbox"/>	host.xml	Running configuration (host part)
Home Directory	<input type="checkbox"/>	/opt/jboss-eap-6.0	Root directory of the server installation
Base Directory	<input type="checkbox"/>	/opt/jboss-eap-6.0/domain	Base directory for server content
Configuration Directory	<input type="checkbox"/>	/opt/jboss-eap-6.0/domain/configuration	Base configuration directory
Log Directory	<input checked="" type="checkbox"/>		the directory where log files will be written for this host controller
Domain Host	<input type="checkbox"/>	master	Host name within the AS7 domain
Product Type	<input type="checkbox"/>	EAP	Server product type (e.g. AS or EAP)

Operations

If a field is not editable immediately, select the **Unset** checkbox, and then enter new information in the field.

- Click the **Save** button.

4.10. Uninventorying and Deleting Resources

A resource can be removed from the JBoss ON inventory in one of two ways: it can be uninventoried (all resources) or it can be deleted (content-backed resources or configuration-related resources).

4.10.1. A Comparison of Uninventorying and Deleting Resources

Uninventorying a resource permanently and irrevocably removes all data about that resource from the JBoss ON inventory. It removes all historical monitoring data, configuration and operation histories, alerts, drift definitions, and any other stored data. However, the resource itself remains intact – it still exists on the machine and can be rediscovered (as a new resource) at a later time.

Any resource can be uninventoried.

Deleting a resource, on the other hand, completely removes the resource *from the machine itself*. So,

deleting an EAR resource deletes the EAR from its parent EAP instance. However, the inventory information about that resource – its historic metric data, configuration history, and version history (for content resources) – remains intact, so that if a new version of that child is ever deployed, it is retains all of its original history.

Only resources not discovered through the discovery queue (such as deployed EAR files or created datasources) can be deleted.

4.10.2. Use Caution When Removing Resources

Uninventory Irrevocably Deletes the Resource History and Data

Uninventorying a resource removes all of the data that JBoss ON has for that resource: its metric data and historical monitoring data, alerts, drift and configuration history, operation history, and other data. **Once the resource is uninventoried, its data can never be recovered.**

Uninventorying or Deleting a Resource Removes All of Its Children

If a parent resource is removed from JBoss ON, then all of its children are also removed. Removing an EAP server, for example, removes all of its deployed web applications from the JBoss ON inventory. Removing a platform removes all servers, services, and resources on that platform.

Uninventoried Resources Can Still Be Discovered

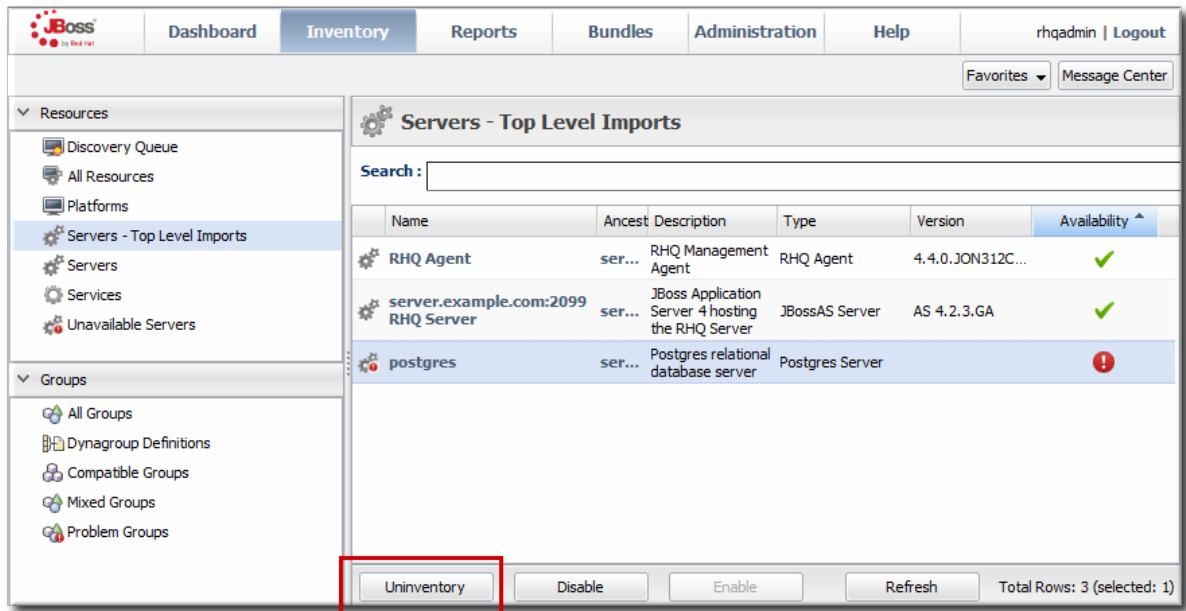
Even though a resource is uninventoried and all of its data in JBoss ON is permanently removed, the underlying resource still exists. This means that the resource can still be discovered. To prevent the resource from being discovered and re-added to the inventory, ignore the resource, as in [Section 4.3.4, “Ignoring Discovered Resources”](#).

Anything Depending on a Deleted Resource Could Fail

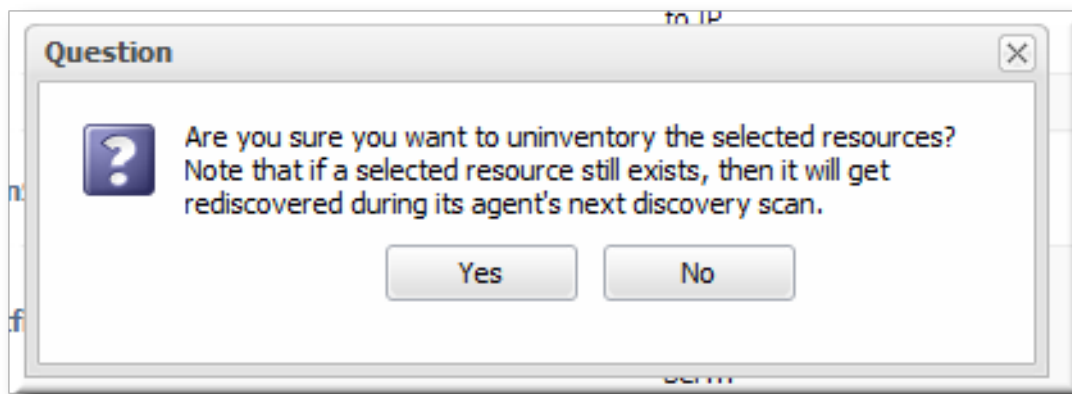
Some resource types can be deleted, meaning the resource itself is removed from the machine, not just from the JBoss ON inventory. Anything that relies on that resource can experience failures because the resource is deleted. For example, if a datasource for an EAP server is deleted, that datasource is removed from the EAP server itself. Any application which attempts to connect to that datasource will then stop working, since it does not exist anymore.

4.10.3. Uninventorying through the Inventory Tab

1. Click the **Inventory** tab in the top menu.
2. Select the resource category in the **Resources** table on the left, and, if necessary, filter for the resource.
3. Select the resource to uninventory from the list, and click the **Uninventory** button.



- When prompted, confirm that the resource should be uninventoried.

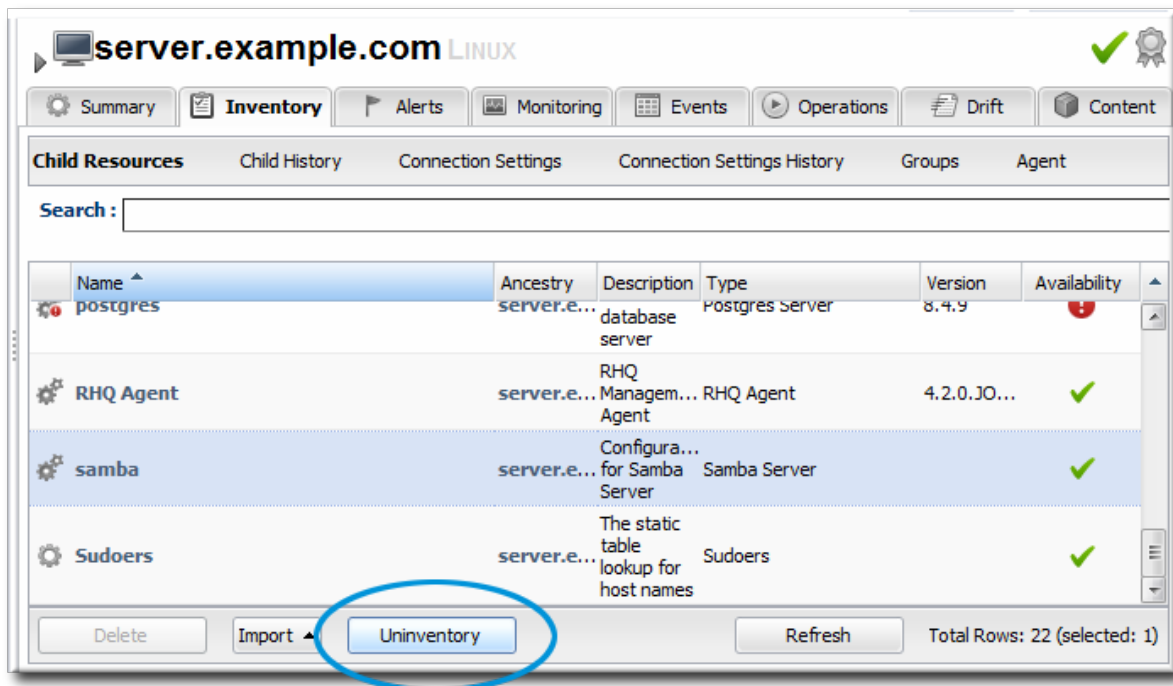


- To prevent the resource from being re-imported into the inventory, ignore it when it is discovered in the next discovery scan. This is covered in [Section 4.3.4, “Ignoring Discovered Resources”](#).

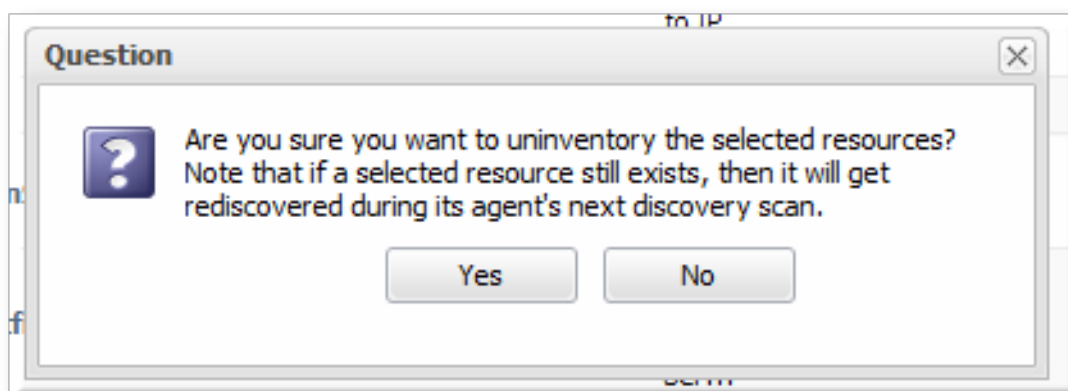
4.10.4. Uninventing through the Parent Inventory

- Click the **Inventory** tab in the top menu.
- Search for the parent resource of the resource.

[Section 2, “Dynamic Searches for Resources and Groups”](#) has information on searching for resources using dynamic searches.
- Click the **Inventory** tab for the parent resource.
- Click on the line of the child resource to uninventoried. To select multiple entries, use the **Ctrl** key.



5. Click the **Uninventory** button.
6. When prompted, confirm that the resource should be uninventoried.

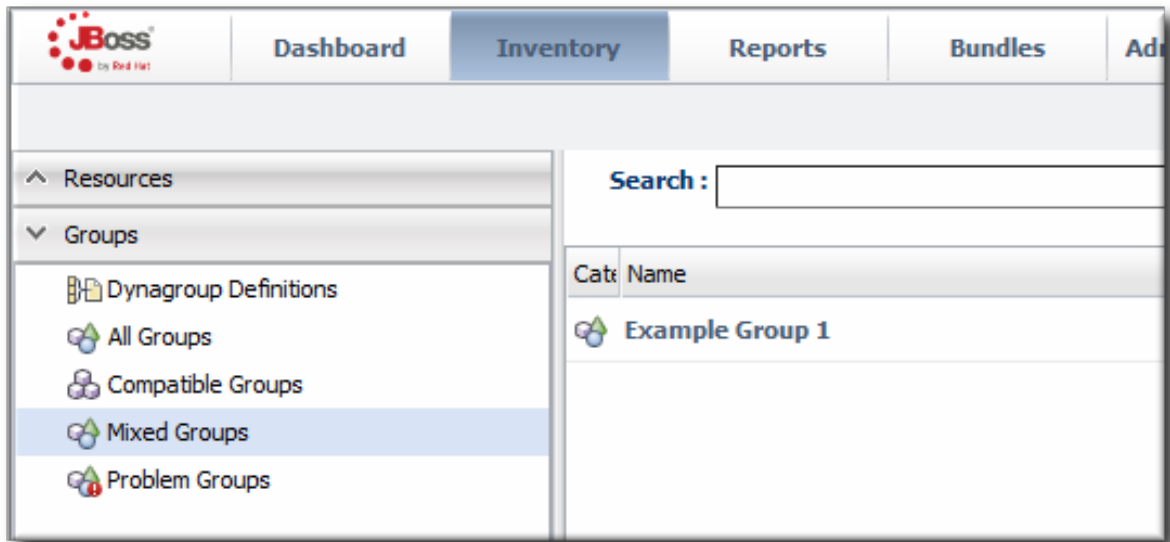


7. To prevent the resource from being re-imported into the inventory, ignore it when it is discovered in the next discovery scan. This is covered in [Section 4.3.4, "Ignoring Discovered Resources"](#).

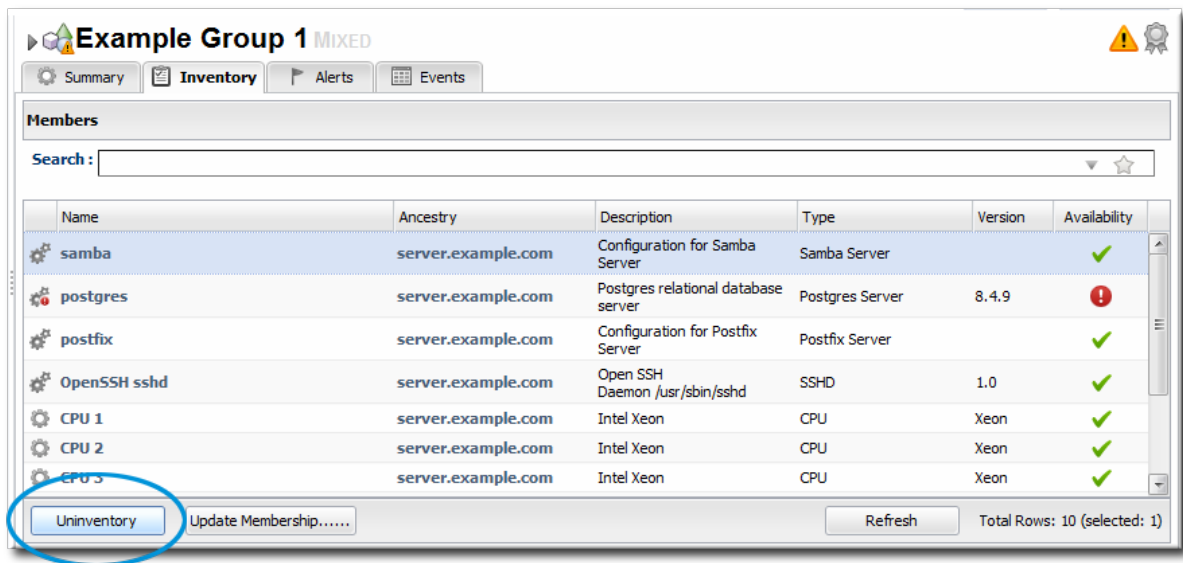
4.10.5. Uninventing through a Group Inventory

If a resource is a member of a group, then the resource can be uninventoried through the group management pages, as part of managing the group resources.

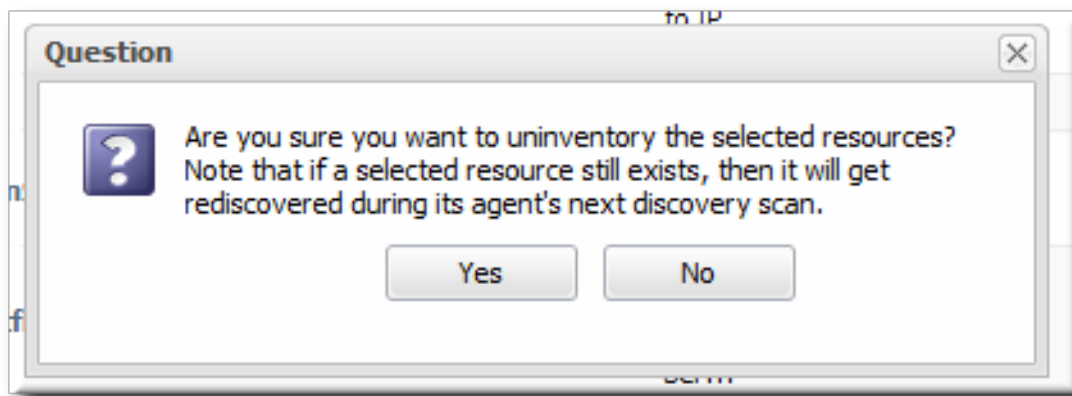
1. In the **Inventory** tab in the top menu, select the compatible or mixed groups item in the **Groups** menu on the left.



2. Click the name of the group.
3. Open the **Inventory** tab for the group, and open the **Members** submenu.
4. Click on the line of the group member to uninventory. To select multiple entries, use the **Ctrl** key.



5. Click the **Uninventory** button.
6. When prompted, confirm that the resource should be uninventoried.



- To prevent the resource from being re-imported into the inventory, ignore it when it is discovered in the next discovery scan. This is covered in [Section 4.3.4, “Ignoring Discovered Resources”](#).

4.10.6. Deleting a Resource

Deleting a resource does several things:

- Deletes the resource from the underlying machine.
- Removes the resource from the inventory.
- Removes any child resources from JBoss ON.
- **Preserves the inventory information in JBoss ON for the resource, including alerts, drift definitions, metric data, and configuration and operation histories.**

Only a resource **not** imported through the discovery queue can be deleted. This generally means that content-backed resources (EARs, WARs, and JARs) and other child resources like datasources can be deleted.



WARNING

Because the real, underlying resource is deleted (not just the inventory entry), anything relying on that resource can experience failures.

1. Click the **Inventory** tab in the top menu.
2. Search for the *parent* resource of the resource to delete.

[Section 2, “Dynamic Searches for Resources and Groups”](#) has information on searching for resources using dynamic searches.

3. Click the **Inventory** tab of the parent resource.
4. Select the resource to delete from the list of children.
5. Click the **Delete** button in the bottom of the **Inventory** tab.

The screenshot shows the JBoss ON interface for an EAP Domain Controller (127.0.0.1:9990) on a JBossAS7 host. The 'Inventory' tab is active, displaying a table of child resources. The table has columns for Name, Ancestry, Description, Type, and Version. The 'Delete' button in the bottom toolbar is circled in red.

Name	Ancestry	Description	Type	Version
	server.example.com	interface		
unsecure	EAP Domain Controller (127.0.0.1:9990) < server.example.com	A named network interface, along with required criteria for determining the IP address to associate with that interface	Network Interface	
default	EAP Domain Controller (127.0.0.1:9990) < server.example.com	One profile in a domain. Profiles are assigned to server groups.	Profile	
full	EAP Domain Controller (127.0.0.1:9990) < server.example.com	One profile in a domain. Profiles are assigned to server groups.	Profile	
EAP server-three	EAP Domain Controller (127.0.0.1:9990) < server.example.com	Managed JBoss Enterprise Application Platform 6 server	Managed Server	EAP 6.0.0.Bet...
master	EAP Domain Controller (127.0.0.1:9990) < server.example.com	Host involved in this domain	Host	
EAP server-one	EAP Domain Controller (127.0.0.1:9990) < server.example.com	Managed JBoss Enterprise Application Platform 6 server	Managed Server	EAP 6.0.0.Bet...
other-server-group	EAP Domain Controller (127.0.0.1:9990) < server.example.com	Server groups on this domain	ServerGroup	
EAP server-two	EAP Domain Controller (127.0.0.1:9990) < server.example.com	Managed JBoss Enterprise Application Platform 6 server	Managed Server	EAP 6.0.0.Bet...
standard-sockets	EAP Domain Controller (127.0.0.1:9990) < server.example.com		SocketBindingGroup	

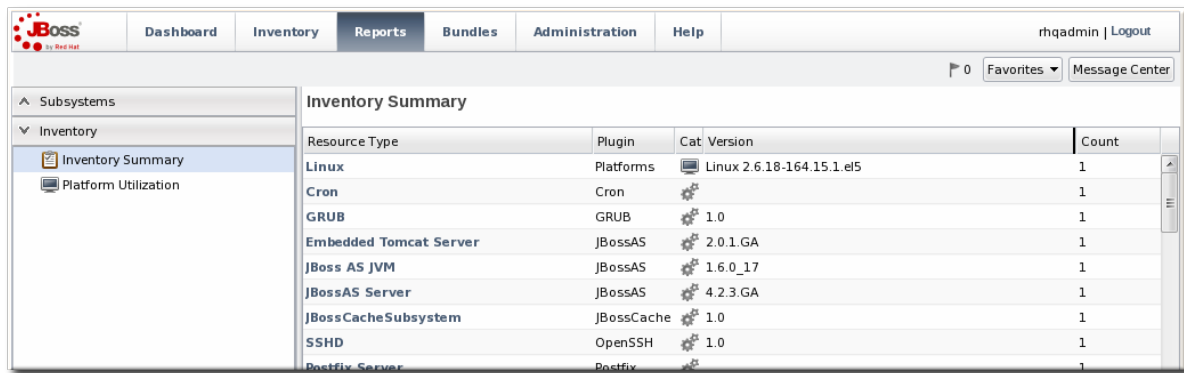
4.11. Viewing Inventory Summary Reports

One quick management tool in JBoss ON is an inventory report. The report summarizes the resources currently in the inventory, grouped by resource type and five summaries:

- Resource type
- The JBoss ON server plug-in which manages the resource
- The JBoss ON category for the resource (platform, server, or service)
- The version number or numbers for resource of the resource type in inventory
- The total number of resources of that type in the inventory

To generate the inventory report:

1. In the top menu, click the **Reports** tab.
2. In the **Inventory** menu box in the menu table on the left, select the **Inventory Summary** report.



Resource Type	Plugin	Cat	Version	Count
Linux	Platforms	Linux	2.6.18-164.15.1.el5	1
Cron	Cron			1
GRUB	GRUB		1.0	1
Embedded Tomcat Server	JBossAS		2.0.1.GA	1
JBoss AS JVM	JBossAS		1.6.0_17	1
JBossAS Server	JBossAS		4.2.3.GA	1
JBossCacheSubsystem	JBossCache		1.0	1
SSHD	OpenSSH		1.0	1
Postfix Server	Postfix			1

3. Click the name of any resource type to go to the inventory list for that resource type.



NOTE

Reports can be exported to CSV, which can be used for office systems or further data manipulation.

To export a report, simply click the **Export** button. The report will automatically be downloaded as `inventorySummary.csv`.

5. MANAGING GROUPS

Groups are a simple, yet effective, way to organize resources. Particularly where there are large numbers of resources or where there are logical divisions between resources across departments, IT environments, or physical locations.

Groups in JBoss Operations Network provide a way to manage resources easily and more consistently. Alerts, operations, and configuration can be applied to individual resources or to entire groups of resources, while groups can be monitored from a single view.

5.1. About Groups

Groups are simply a means to organize resources within the JBoss ON inventory. JBoss ON has several different kinds of groups, listed in [Table 8, “Types of Groups”](#), which allows an administrator to manage resources in different, flexible ways.

Table 8. Types of Groups

Type	Description	Static or Dynamic
Mixed groups	Contains resources of any resource type. There is no limit to how many or what types of resources can be placed into a mixed group. Mixed groups are useful for granting access permissions to users for a set of grouped resources.	Static

Type	Description	Static or Dynamic
Compatible groups	Contains only resources of the same type. Compatible groups make it possible to perform an operation against every member of the group at the same time, removing the need to individually upgrade multiple resources of the same type, or perform other operations one at a time on resources across the entire enterprise.	Static
Recursive groups	Includes all the descendant, or child, resources of resources within the group. Recursive groups show both the explicit member availability and the child resource availability.	Static (members) and dynamic (children)
Autogroups	Shows every resource as part of a resource hierarchy with the platform at the top, and child and descendant resources below the platform. Child resources of the same type are automatically grouped into an autogroup.	Dynamic

5.1.1. Dynamic and Static Groups

Groups are a way of organizing resources. The different types of groups are covered in [Table 8, “Types of Groups”](#), but all of these groups fall into one of two categories. Groups are either *static* or *dynamic*, depending on how resources are assigned to the group. Static groups have resources which are explicitly assigned to the group, so the membership does not change even if the inventory changes. Dynamic groups are based on some kind of search criteria, and the group members are all of the resources returned in that search. Whenever the inventory is updated, the search results change, and the group membership is automatically updated.

Both static and dynamic groups can be valuable for managing resources and keeping a perspective on the overall IT environment.

5.1.2. About Autogroups

There are two basic types of groups in JBoss ON: static groups, where resources are added manually, and dynamic groups, where resources are added automatically based on some kind of established criteria.

Administrators can configure dynamic groups based on defined searches, which is covered in [Section 6, “Using Dynamic Groups”](#). JBoss ON supports a different kind of dynamic group called an *autogroup*. Autogroups are used to construct the inventory navigation trees in the JBoss ON UI, and they are based on the underlying resource hierarchy, or parent-child relationships. Autogroups also

group along resource type. For example, in Figure 31, “PostgreSQL Autogroup”, there are autogroups under the Postgres resource for all its children, which are further divided based on the child resource type, databases and users.

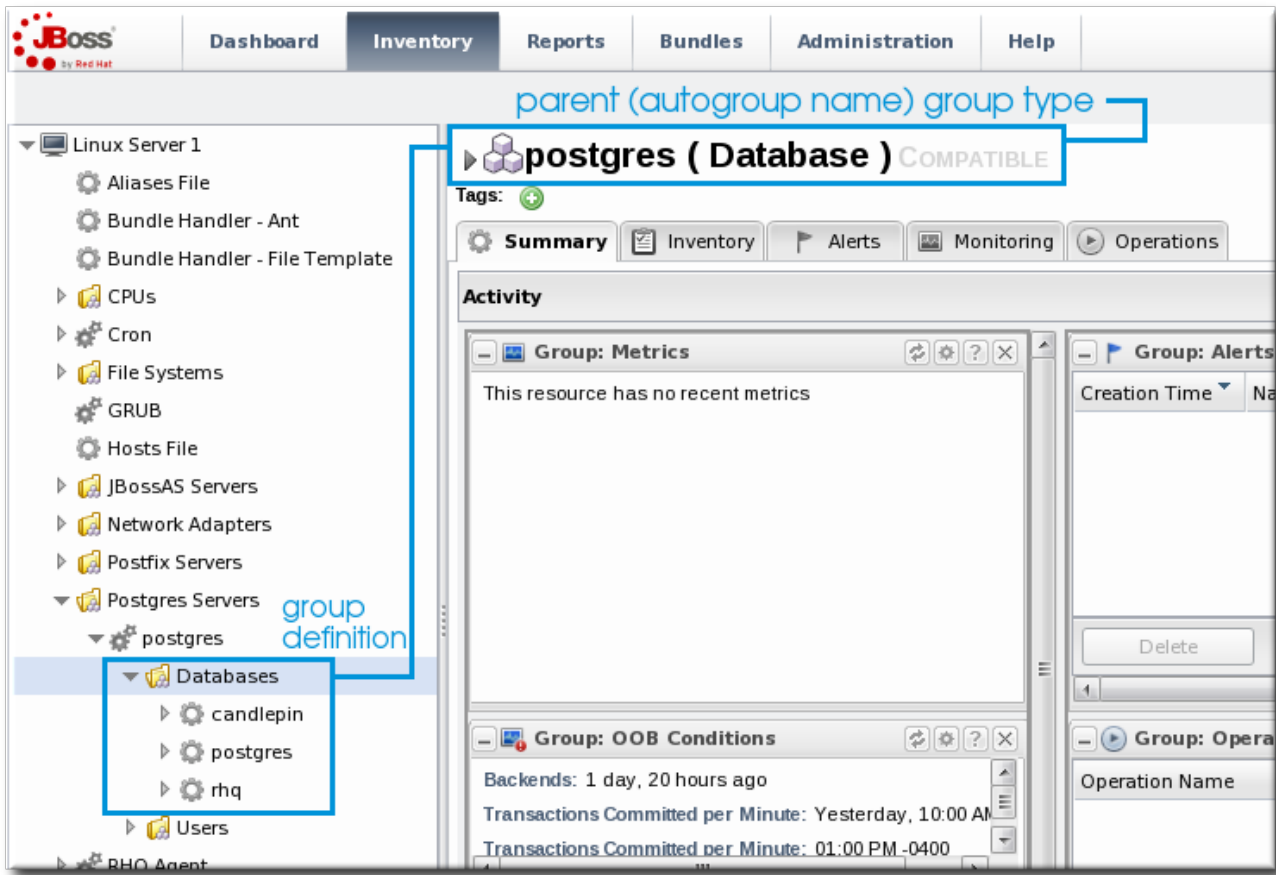


Figure 31. PostgreSQL Autogroup

Autogroups, unlike other groups in JBoss ON, are not configurable by JBoss ON users. Autogroups are defined internally in the JBoss ON server and are used by JBoss ON.

5.1.3. Comparing Compatible and Mixed Groups

Using groups allows multiple resources to be managed simultaneously. The type of group – compatible or mixed – specifies what kind of management can be performed on the group members.

Compatible groups, because they have members all of the same type, can be managed almost as easily as a single resource. Administrators can change resource configuration, launch operations, set alerts, and view individual and group-averaged monitoring data. Any changes can be made to a single group member, selected members, or the entire group. The list of group members, the group *inventory*, is managed through the **Inventory** tab.

Figure 32. Compatible Group Entry

Mixed groups can have members of different resource types, so group management is limited to updating the members (the group inventory) and viewing the history of alerts and events for the group members.

Figure 33. Mixed Group Entry

5.1.4. Leveraging Recursive Groups

Compatible and mixed groups have a set, explicit membership. This static structure makes them useful for creating policies within JBoss ON because they are reliable.

Compatible and mixed groups can have a setting on them making them *recursive*. A recursive group travels down the inventory of every member and implicitly adds all of their children to the group, too. In a sense, a recursive group has two tiers of members: the explicit members to the group and then the implicit set of child members.

This idea of two levels of members allows a group to retain its own type definition – meaning specifically that it does not change a compatible group to a mixed group just because its children are added as members. The explicit membership is called the *root node*. Group operations, from adding members to setting metrics schedules to changing connection settings, are only performed on that root node.

Recursive groups can be useful in a lot of different ways.

For example, recursive compatible groups can be used to look at a subset of autoclusters in the inventory because all of the child resources are grouped by type, as are their parent resources. This provides an easy view at a subset of resources. (Mixed groups, which do not group by type, show only the root node or explicit members in the hierarchy.)

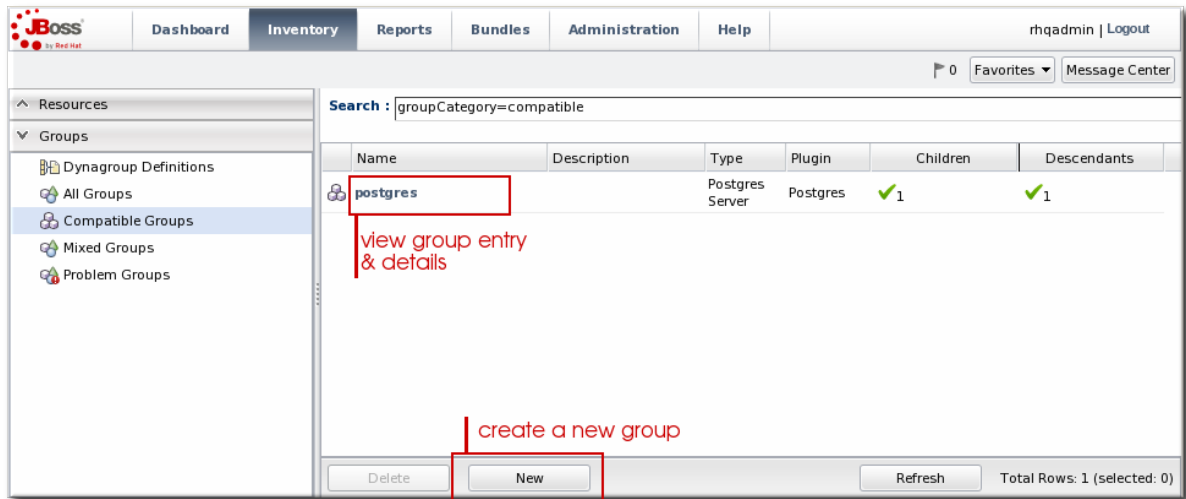
One of the more important uses is using recursive groups, particularly mixed recursive groups, for authorization control. This is covered more in [Section 8, “Managing Roles and Access Control”](#), but users have to be granted access to resources explicitly, by adding the user and the resource to the same role. Using a recursive group automatically includes all of those resources' children in the role, which makes the role easier to maintain and more accurate in granting access.

5.2. Creating Groups

A user must have the global security or inventory permission to create groups.

1. Click the **Inventory** tab in the top menu.
2. In the **Groups** box in the left menu, select the type of group to create, either compatible or mixed.

Compatible groups have resources all of the same type, while mixed groups have members of different types. The differences in the types of members means that there are different ways that compatible and mixed groups can be managed, as covered in [Section 5.1.3, “Comparing Compatible and Mixed Groups”](#).



3. Enter a name and description for the group.

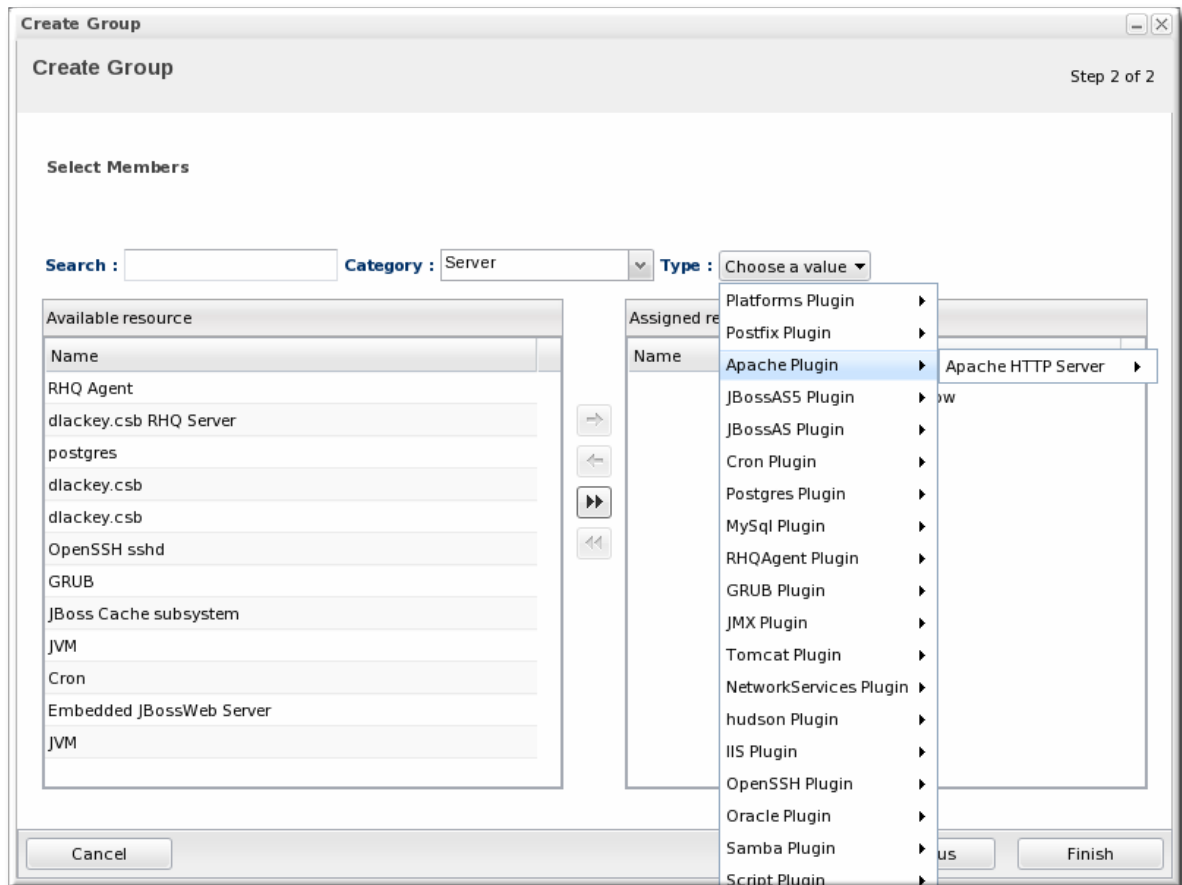
The 'Create Group' dialog box is shown, titled 'Create Group' and indicating 'Step 1 of 2'. The 'Group Settings' section contains the following fields:

- Name :** My Example Group
- Description :** For a lot of resources.
- Recursive

At the bottom of the dialog, there are three buttons: 'Cancel', 'Previous', and 'Next'.

Marking groups recursive can make it easier to manage resources, particularly when setting role access controls. For example, administrators can grant users access to the group and automatically include any child resources of the member resources.

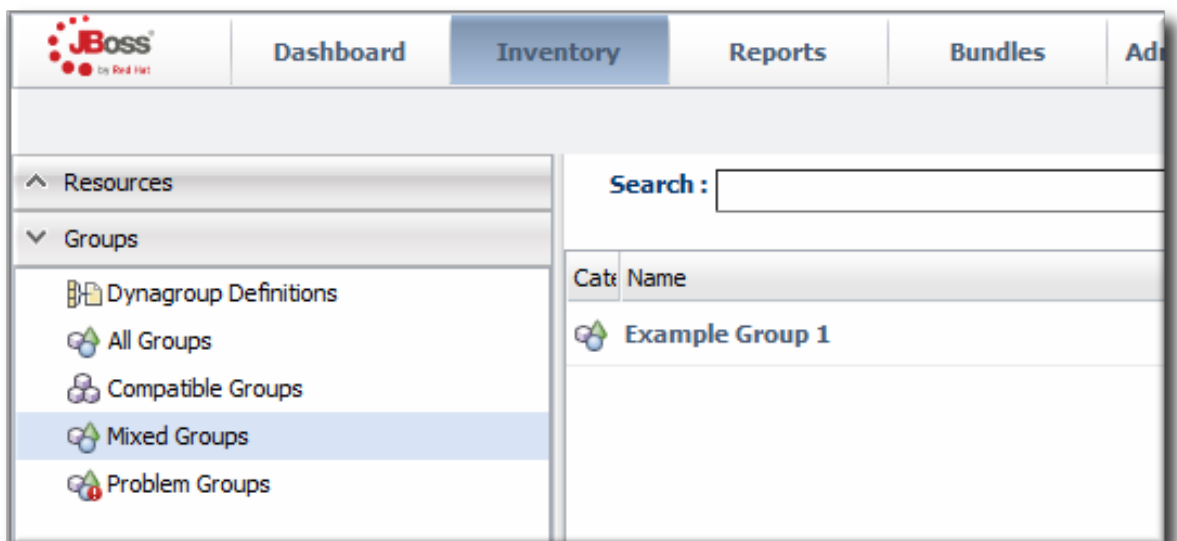
4. Select the group members. It is possible to filter the choices based on name, type, and category.



5.3. Changing Group Membership

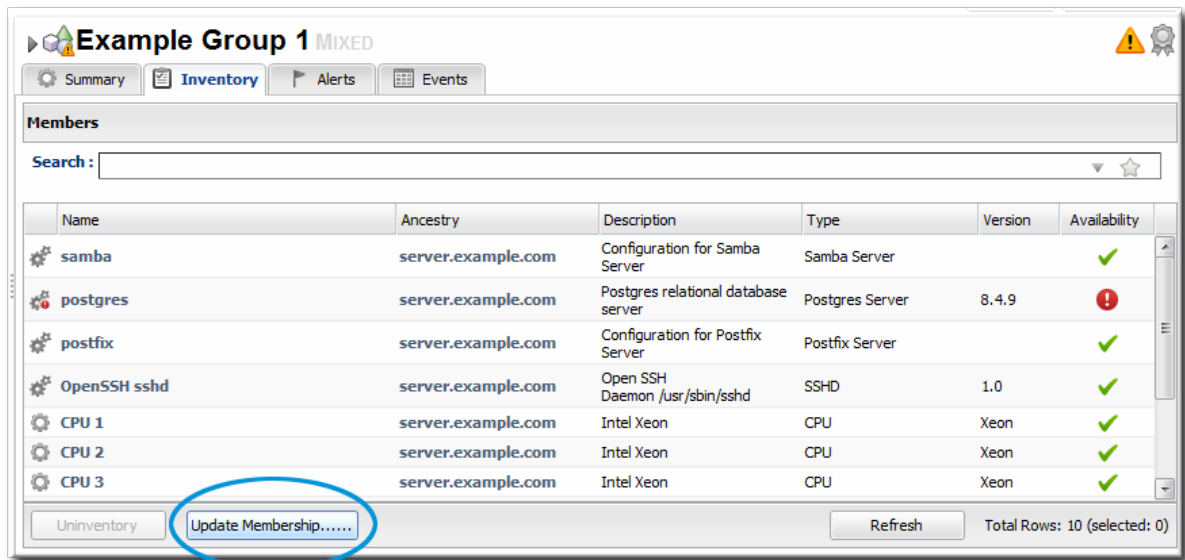
Compatible and mixed groups both have static members, which means that resources are manually assigned to the group rather than being assigned dynamically based on some attribute. The group membership can be changed as the resources in the JBoss ON inventory change.

1. In the **Inventory** tab in the top menu, select the compatible or mixed groups item in the **Groups** menu on the left.

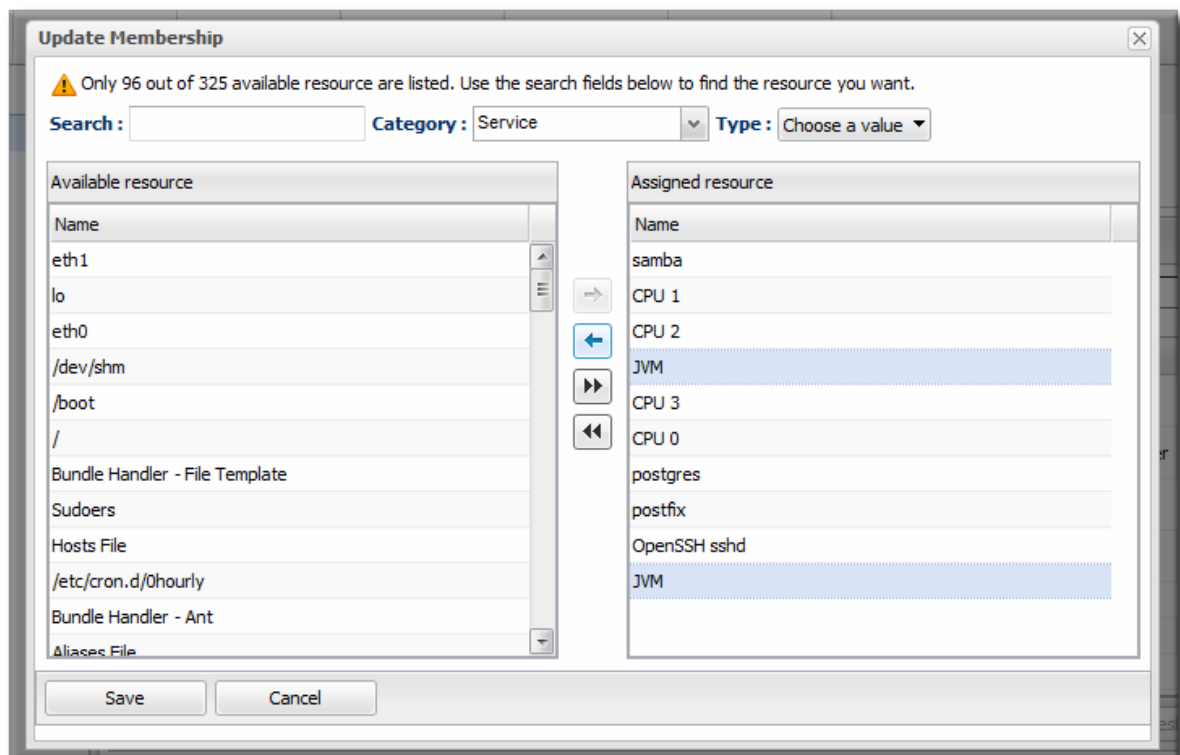


2. Click the name of the group.
3. Open the **Inventory** tab for the group, and open the **Members** submenu.

- Click the **Update Membership** button at the bottom of the page.



- Select the resources to add to the group from the box on the left; to remove members, select them from the box on the right. Use the arrows to move the selected resources. To select multiple resources, use **Ctrl+click**.



- Click the **Save** button.

5.4. Editing Compatible Group Connection Properties

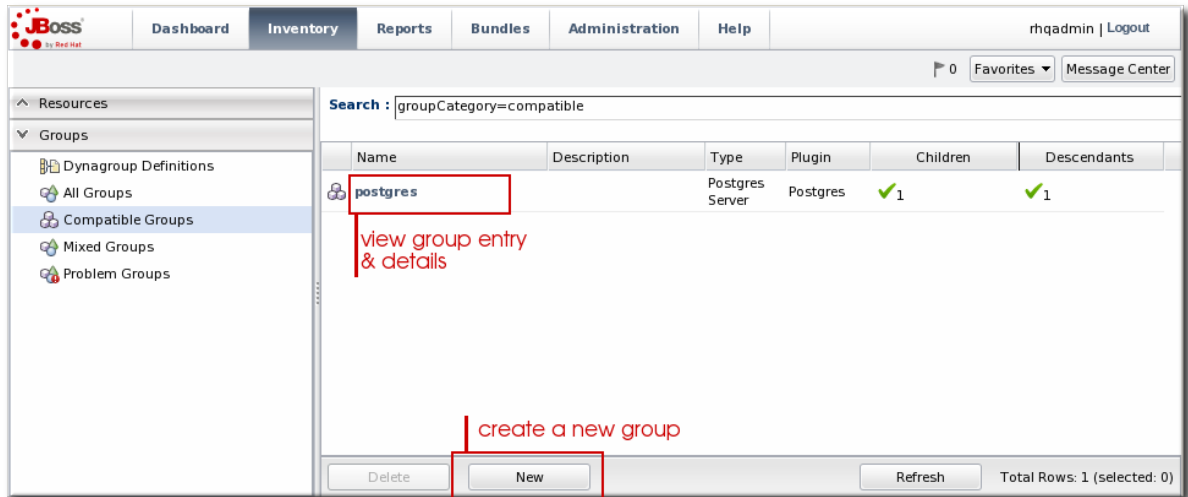
Compatible groups manage the *connection properties* of the group members as part of the inventory. Since a compatible group can only contain members of the same resource type, it is possible to see an aggregate view or average of all of their individual connection properties. The connection settings define how the agent or server connects to the resource.

The rules for the values of compatible group connections are simple:

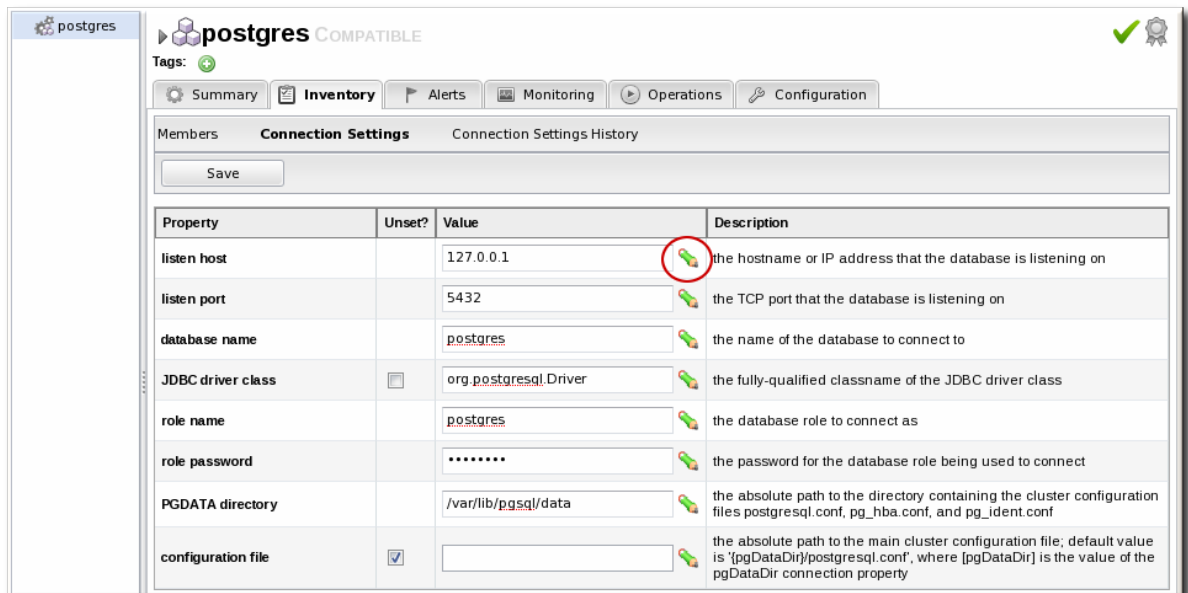
- If all of the resources in the group have identical values for a property, the group connection property is that exact value.
- If even one resource has a different value than the rest of the resources in the group, that property will have a special marker value of ~ *Mixed Values* ~.

To edit the connection properties:

1. In the **Inventory** tab in the top menu, select the **Compatible Groups** item in the **Groups** menu on the left.



2. Click the name of the compatible group.
3. Open the **Inventory** tab for the group, and click the **Connection Settings** sub-item.
4. To edit a property, click the green pencil by the field.



5. To change all resources to the same value, click the **Unset** checkbox for the field **Set all values to . . .** To change a specific resource, click the **Unset** checkbox for that resource and then give the new value.



NOTE

Refreshing the inventory tab shows the *current* values of the connection properties for each resource in the group. If the update has not yet completed, but it has successfully changed some of the resource's connection properties, intermediate values are displayed. Just ignore these values. Once the updates have completed, refresh the page to view final results.

The **Connection Settings History** sub-item shows the changes made to the connection properties. If there is a failure, clicking the hyperlink in the **Date Created** column opens any relevant error messages.

6. USING DYNAMIC GROUPS

A dynamic groups specifies a search term to use to search the inventory and identify matching resources to belong to the group. Since the search results change automatically as results are added and removed from the inventory, the group membership is always changing and always current. Using dynamic groups helps automate management tasks for large inventories.

NOTE

Dynamic groups are also referred to by the nickname *dynagroups*.

Enterprise resources can be grouped by cluster identifier, broadcast group, logical service layer, geographical location, security domain, or any other logical grouping.

Individual resources can potentially belong to multiple groups, in large inventories it is important to know how different group definitions will affect the enterprises resources.

6.1. About Dynamic Groups Syntax

Dynamic groups are configured through *group definitions*. A group definition uses *expressions* which define searches for resources, along with other information about the group like the recalculation interval.

Dynamic groups have an expression syntax very similar to the one used for dynamic searches ([Section 2.2, “About the Dynamic Search Syntax”](#)).

6.1.1. General Expression Syntax

An expression can either group resources by a specific resource attribute or value (a *simple* expression) or by the resource type (a *pivoted* expression). The expression is a search condition that is centered around a specific resource attribute, either by a specific value of an attribute or simply by the presence of an attribute.

A single group definition can have multiple expressions. The order of expressions in the group definition does not matter; for example, both of these expressions are interpreted exactly the same when calculating the group members:

```

expression 1
exprA1
exprA2
groupby exprB1
groupby exprB2

expression 2
exprA2
exprA1
groupby exprB2
groupby exprB1
    
```



NOTE

When multiple expressions are used in a group definition, they are treated as logical AND expressions, and a resource must match all the criteria to belong to the group.

Any empty lines between expressions in a dynagroup definition are ignored.

There are ten possible resource attributes, covering resource information like the resource name, type, plug-in, version, configuration property, and inventory ID number.

Table 9. Dynamic Group Properties

Type	Supported Attributes						
Related to the resource itself							
resource	<table border="1"> <tr><td data-bbox="817 1512 1428 1585">id</td></tr> <tr><td data-bbox="817 1585 1428 1659">name</td></tr> <tr><td data-bbox="817 1659 1428 1733">version</td></tr> <tr><td data-bbox="817 1733 1428 1807">parent</td></tr> <tr><td data-bbox="817 1807 1428 1881">grandparent</td></tr> <tr><td data-bbox="817 1881 1428 1955">children</td></tr> </table>	id	name	version	parent	grandparent	children
id							
name							
version							
parent							
grandparent							
children							
Related to the resource type							

Type	Supported Attributes			
resourceType	<table border="1"> <tr> <td>plug-in</td> </tr> <tr> <td>name</td> </tr> <tr> <td>category (platform, server, service)</td> </tr> </table>	plug-in	name	category (platform, server, service)
plug-in				
name				
category (platform, server, service)				
Related to the resource configuration				
plug-inConfiguration	Any plugin configuration property			
resourceConfiguration	Any resource configuration property			
Related to the resource monitoring data				
traits	Any monitoring trait			
availability	The current state, either UP or DOWN			

If an expression has the structure `resource.attribute`, then it applies to the resource which will be a member of the group. However, it is possible to use an attribute in an ancestor or child entry to identify a group member recursively.

For example, to add a resource as member which has an inventory ID of 10001, the expression is:

```
resource.id = 10001
```

To add all of the *children* of a resource with an ID of 10001 as group members, use the prefix `resource.parent`:

```
resource.parent.id = 10001
```

There are four possible prefixes for all of the resource attributes in [Table 9, “Dynamic Group Properties”](#):

- resource
- resource.child
- resource.parent
- resource.grandParent

If a definition is so restrictive that no resources match the filters, no group is created. JBoss ON will actually suppress a group from being created by a group definition if it would result in an empty group. Because there are no empty groups, there are no extraneous groups listed in the inventory, which makes managing inventories easier and better reflects your real infrastructure.

6.1.2. Simple Expressions

A simple expression uses an attribute-value pair or triad in this format:

```
resource.attribute[string-expression] = value
```

For example:

```
resource.parent.type.category = Platform
```

Not every resource attribute has an additional *string-expression*; a *string-expression* is basically a sub-attribute. For example, `resource.trait` is the generic resource attribute, and a sub-attribute like `partitionName` identifies the actual parameter.

Simple expressions usually search for resources based on an explicit value, but resources may have attributes present with null values, and those null values would be not returned with a simple expression. The `empty` keyword searches for resources which have a specific attribute with a null value:

```
empty resource.attribute[string-expression]
```

If the `empty` keyword is used, then there is no *value* given with the expression.

Simple expressions can also use a `not empty` keyword, which looks for every resource with that attribute, regardless of the attribute value, as long as it is not null. As with the `empty` keyword, there is no reason to give a *value* with the expression, since every value matches the expression.

```
not empty resource.attribute[string-expression]
```

6.1.3. Pivot Expressions

Simple expressions create a single group, because they are based on the specific results of the search. Alternatively, a *pivot expression* creates multiple groups because it identifies resources which belong to the group solely based on whether an attribute exists; it creates subgroups based on the values. A pivot expression uses the `groupby` keyword:

```
groupby resource.attribute
```

Pivoted expressions create groups based on unique occurrences of an attribute value. For example, the `parent.name` attribute creates a unique group based on every parent resource.

```
groupby resource.parent.name
```

For the resources in [Figure 34, “Resources and Parents”](#), the pivot expression creates groups for the three parents within the resource hierarchy: ResourceParentA, ResourceParentB, and ChildA2.

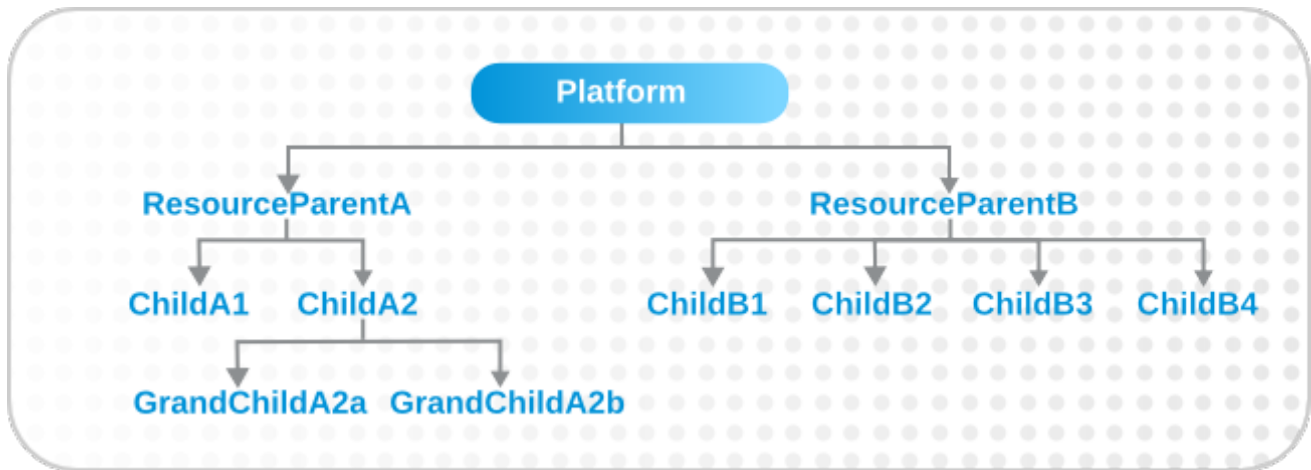


Figure 34. Resources and Parents

If the overall group definition includes resources with null values, then the pivot expression creates a special subgroup that contains those resources.

6.1.4. Compound Expressions

Multiple expressions can be used in a single dynagroup definition; these are *compound* expressions.

When multiple expressions are used in a group definition, they are treated as logical AND expressions, and a resource must match all the criteria to belong to the group. (Any empty lines between expressions in a dynagroup definition are ignored.)

For example, this basic expression searches for every resource which has the platform as its parent:

```
resource.parent.type.category = Platform
```

That could return a very long list of servers and services. That initial list can be further filtered by adding another simple expression, which filters by name:

```
resource.parent.type.category = Platform
resource.name.contains = JBossAS
```

Only resources with the platform as the parent *and* with the string *JBossAS* in their name will be added to the group.

Pivoted expressions can also be used in compound expressions. Every line must have the *groupby* keyword, not only the first line.

```
groupby resource.type.plugin
groupby resource.type.name
groupby resource.parent.name
```

A compound expression can contain both simple and pivoted expressions. This creates a compatible group for every unique server type on the platform.

```
resource.type.category = server
groupby resource.type.plugin
groupby resource.type.name
groupby resource.parent.name
```

Lastly, compound expressions can include the empty and not empty keywords. For example, simple expressions can be used to identify JBoss servers based on the resource type and name. Then, to identify which JBoss servers are unsecured, the expression can filter for JBoss servers with a principal connection property with an empty value.

```
resource.type.plugin = JBossAS
resource.type.name = JBossAS Server
empty resource.pluginConfiguration[principal]
```

6.1.5. Unsupported Expressions

There are restrictions on how multiple expressions can be used together.

All expressions must be in the same configuration area.

All given configuration properties in an expression must be only from the resource configuration or only from the plug-in configuration. Expressions cannot be taken from both.

Each property must only be used once.

A property can only be used once in a dynagroup definition.

```
valid
resource.trait[x] = foo

not valid
resource.trait[x] = foo
resource.trait[y] = bar
```

For example, a `resource.trait` expression can only occur once in a definition:

```
resource.grandParent.trait[Trait.hostname].contains = stage
resource.parent.type.plugin = JBossAS5
resource.type.name = Web Application (WAR)
```

If it is used a second (or more) time, then the subsequent use fails and causes the definition not to be parsed.

```
resource.grandParent.trait[Trait.hostname].contains = stage
resource.parent.type.plugin = JBossAS5
resource.type.name = Web Application (WAR)
resource.trait[contextRoot] = jmx-console
```

This results in calculation errors:

```
There was a problem calculating the results:
java.lang.IllegalArgumentException: org.hibernate.QueryParameterException:
could not locate named parameter [arg2]
```

The `[arg2]` error is a sign that multiple expressions of the same type were used and that the second and subsequent expressions caused a calculation failure.

This is true even if the property type is used with *different* resource contexts.

```
resource.parent.trait[x] = foo
```

```
resource.grandParent.trait[y] = bar
```

In the previous example, one trait applied to the grandparent resource and one trait applied to the resource itself. This still failed because the trait property was used twice, even though it was for different resources.

6.1.6. Dynagroup Expression Examples

A single group definition can have multiple expressions, even mixing simple and pivoted expressions in a single definition. Many of these examples require multiple expressions to complete the definition.

Example 1. JBoss Clusters

```
resource.type.plugin = JBossAS
resource.type.name = JBossAS Server
groupby resource.trait[partitionName]
```

Example 2. A Group for Each Platform Type

```
resource.type.plugin = Platforms
resource.type.category = PLATFORM
groupby resource.type.name
```

Example 3. Autogroups

```
groupby resource.type.plugin
groupby resource.type.name
groupby resource.parent.name
```



NOTE

This could create a large number of groups in large inventories.

Example 4. Raw Measurement Tables

```
resource.type.plugin= Postgres
resource.type.name = Table
resource.parent.name = rhq Database
resource.name.contains = rhq_meas_data_num_
```

Example 5. Only Agents with Multicast Detection

```
resource.type.plugin= RHQAgent
resource.type.name = RHQ Agent
resource.resourceConfiguration[rhq.communications.multicast-
detector.enabled] = true
```

Example 6. Only Windows Platforms with Event Tracking

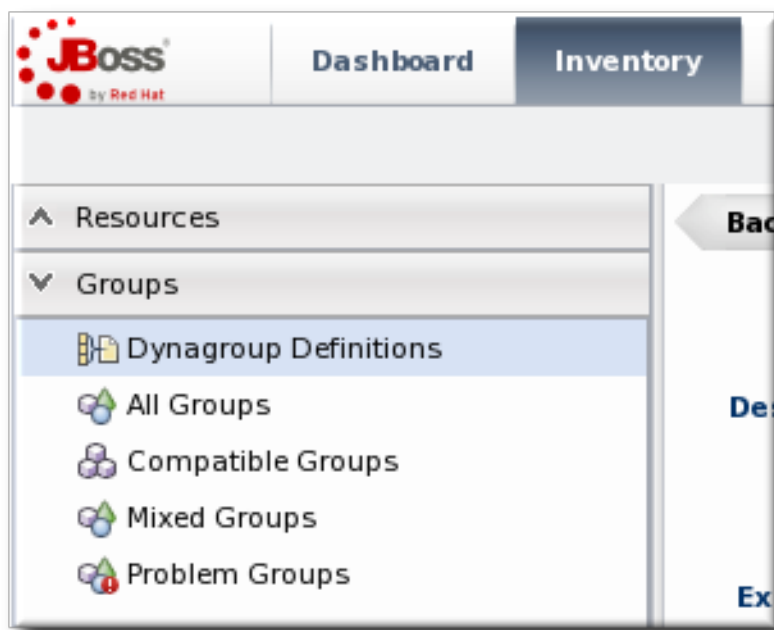
```
resource.type.plugin= Platforms  
resource.type.name = Windows  
resource.pluginConfiguration[eventTrackingEnabled] = true
```

Example 7. JBoss AS Servers by Machine

```
groupby resource.parent.trait[Trait.hostname]  
resource.type.plugin = JBossAS  
resource.type.name = JBossAS Server
```

6.2. Creating Dynamic Groups

1. Click the **Inventory** tab in the top menu.
2. In the **Groups** menu box on the left, click the **Dynagroup Definitions** link.



3. Click the **New** button to open the dynamic group definition form.



Inventory					Reports	Bundles	Administration	Help	rhqadmin Logout	
					0		Favorites	Message Center		
DynaGroup Definitions										
Name	Description	Expression Set	Last Calculation Time	Next Calculation Time						
All Platforms	for all platform resources	resource.type.category = Platform	Jun 1, 2011 10:53:03 AM	Jun 1, 2011 10:54:03 AM						
					Delete		New		Recalculate	
					Refresh		Total Rows: 1 (selected: 0)			

- Fill in the name and description for the dynamic group. The name can be important because it is prepended to any groups created by the definition, as a way of identifying the logic used to create the group.

[Back to List](#)

Name * :

Description :

Saved Expression :  

Expression * :

Recursive

Recalculation Interval (ms) :

DynaGroup Children

Name	Description	Type	Plugin
No items to show.			

Total Rows: 0 (selected: 0)

- Fill in the search expressions. This can be done by entering expressions directly in the **Expression** box or by using a saved expression.

Saved expressions have a wizard to help build and validate the expressions. To create a saved expression, click the green button by the drop-down menu. Several options for the expression are active or inactive depending on the other selections; this prevents invalid

expressions.

The **Expression** box at the top shows the currently created expression.

6. After entering the expressions, set whether the dynamic group is recursive.
7. Set an optional recalculation interval. By default, dynamic groups do not recalculate their members automatically, meaning the recalculation value is set to 0. To recalculate the group membership, set the **Recalculation interval** to the time frequency, in milliseconds.



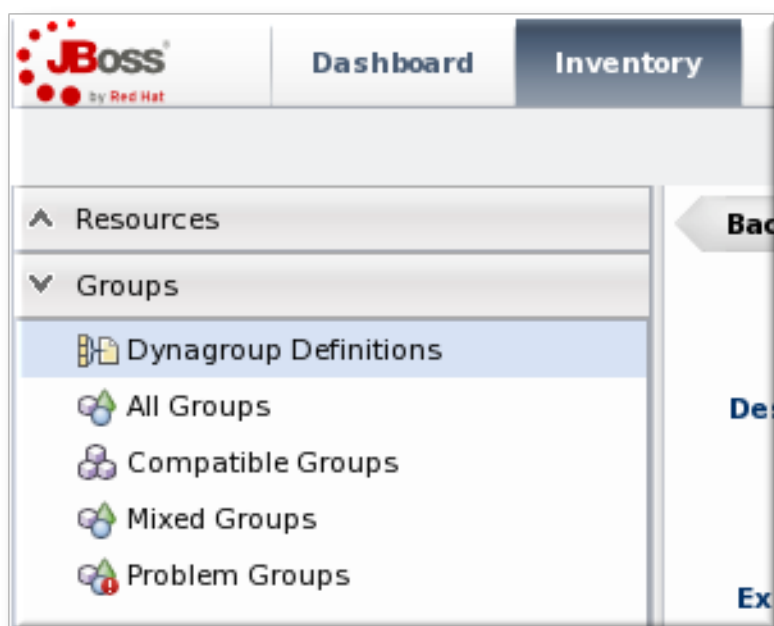
NOTE

Recalculating a group definition across large inventories could be resource-intensive for the JBoss ON server, so be careful when setting the recalculation interval. For large inventories, set a longer interval, such as an hour, to avoid affecting the JBoss ON server performance.

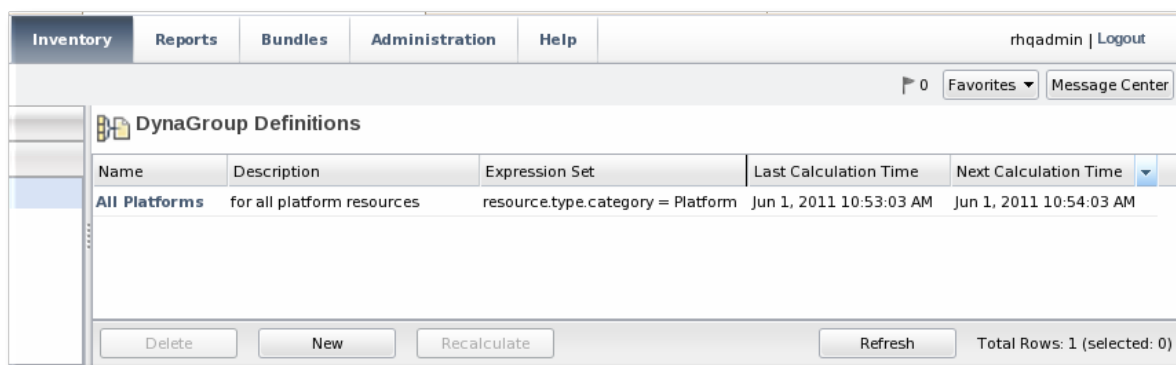
6.3. Recalculating Group Members

Dynamic groups can be recalculated apart from whatever interval is set in the group definition. The recalculation interval in the group definition is a relative value based on the last update time, so initiating a recalculation manually will not conflict or interfere with the normal recalculation; it simply proceeds after the specified amount of times elapses.

1. Click the **Inventory** tab in the top menu.



2. In the **Groups** menu on the left, click the **Dynagroup Definitions** link.
3. In the list of dynagroups, select the row of the dynagroup definition to calculate.



4. Click the **Recalculate** button at the bottom of the table.

7. CREATING USER ACCOUNTS

Users are part of the overall security planning for JBoss ON, even though they don't have access controls set on their accounts individually.

7.1. Managing the rhqadmin Account

When JBoss ON is installed, there is a default superuser already created, `rhqadmin`. This superuser has the default password `rhqadmin`.

**NOTE**

The `rhqadmin` account cannot be deleted, even if other superuser accounts are created. Additionally, the role assignments for `rhqadmin` cannot be changed; it is always a superuser account.

**IMPORTANT**

If a user is deleted, scheduled operations owned by the user are canceled.

When you first log into JBoss ON after installation, change the superuser password.

1. Click the **Administration** tab in the top menu.
2. In the **Security** table on the left, select **Users**.

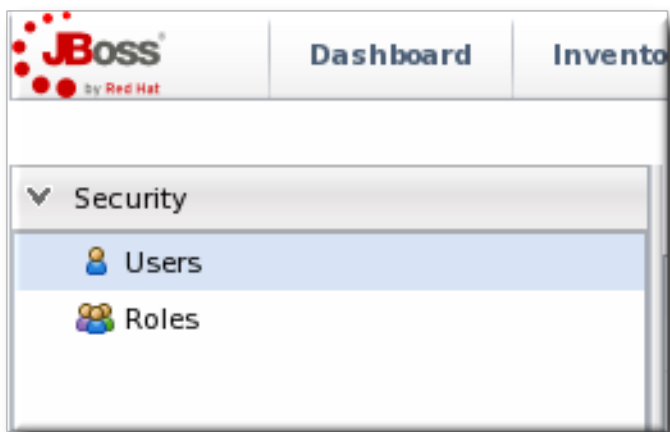


3. Click the name of `rhqadmin`.
4. In the edit user form, change the password to a new, complex value.

Edit User [rhqadmin]	
User Name * : rhqadmin	LDAP Login? : no
Password * :	Verify Password * :
First Name * : RHQ	Last Name * : Administrator
Email Address * : rhqadmin@localhost	Phone Number :
Department :	Login Enabled? * : yes
Assigned Roles	
Name	
Super User Role	

7.2. Creating a New User

1. Click the **Administration** tab in the top menu.
2. In the **Security** table on the left, select **Users**.



3. Click the **NEW** button at the bottom of the list of current users.
4. Fill in description of the new user. The **Enable Login** value must be set to **Yes** for the new user account to be active.

 A screenshot of the 'Create New User' form. The form is titled 'Create New User' and contains several input fields and checkboxes. The fields are:

- User Name ***: bjensen
- Password ***: [masked with dots]
- First Name ***: Barbara
- Email Address ***: bjensen@example.com
- Department ***: [empty]
- LDAP Login?**: no
- Verify Password ***: [masked with dots]
- Last Name ***: Jensen
- Phone Number ***: [empty]
- Login Enabled? ***: yes no

 Below the form, there are two panels:

- Available Roles**: A table with a header 'Name' and two rows: 'Super User Role' and 'All Resources Role'.
- Assigned Roles**: A table with a header 'Name' and the text 'No items to show'.

 Between the two panels are four arrow buttons: a right-pointing arrow, a left-pointing arrow, a double right-pointing arrow, and a double left-pointing arrow.

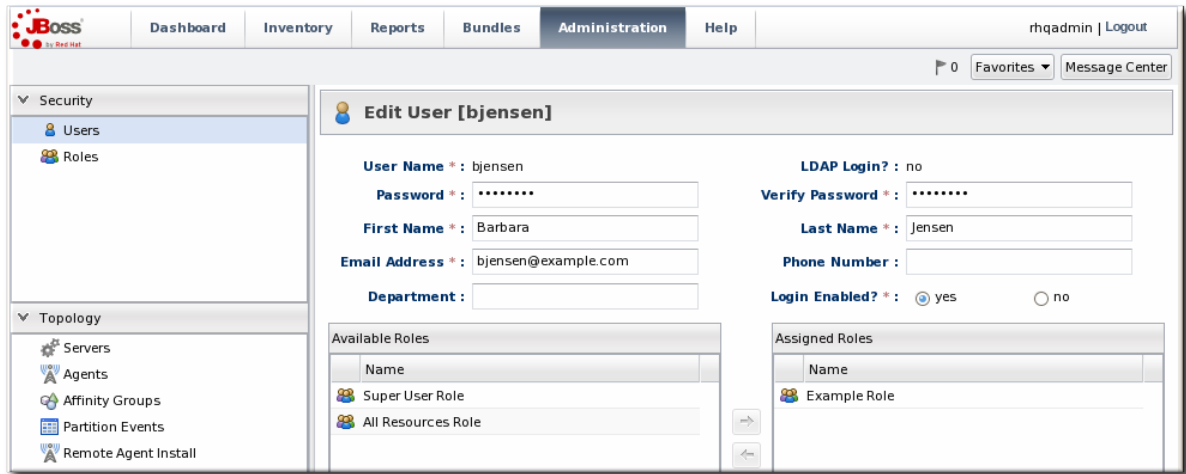
5. Select the required role from the **Available Roles** area, and then click the arrow pointing to the **Assigned Roles** to assign the role.
6. Click the **Save** button to save the new user with the role assigned.

7.3. Editing User Entries

All users can edit their own account details, and users with administrative rights (who belong to a role which grants them rights over user entries) can edit other users' entries.

1. Click the **Administration** tab in the top menu.

2. From the **Security** menu, select **Users**.
3. Click the name of the user whose entry will be edited.
4. In the edit user form, change whatever details need to be changed, and save.



7.4. Disabling User Accounts

User accounts can be temporarily disabled. This can be done for a security review or when there is some kind of breach, but users don't need to be deleted. The **Enable Login** property can prevent the user from logging into the JBoss ON UI and managing resources or making configuration changes.

1. Click the **Administration** tab in the top menu.
2. In the **Security** table on the left, select **Users**.



3. Click the name of the user whose entry will be edited.
4. In the edit user form, change the **Enable Login** radio button to **No**.

Last Name * : Jensen

Phone Number :

Login Enabled? * : yes no

Assigned Roles

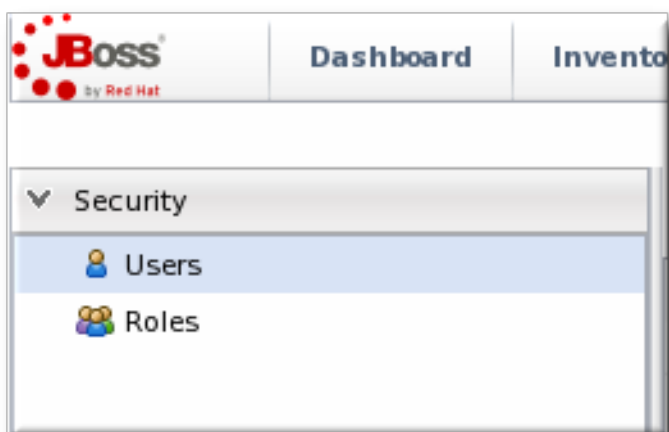
Name

5. Click the **Save** button to save the new user with the role assigned.

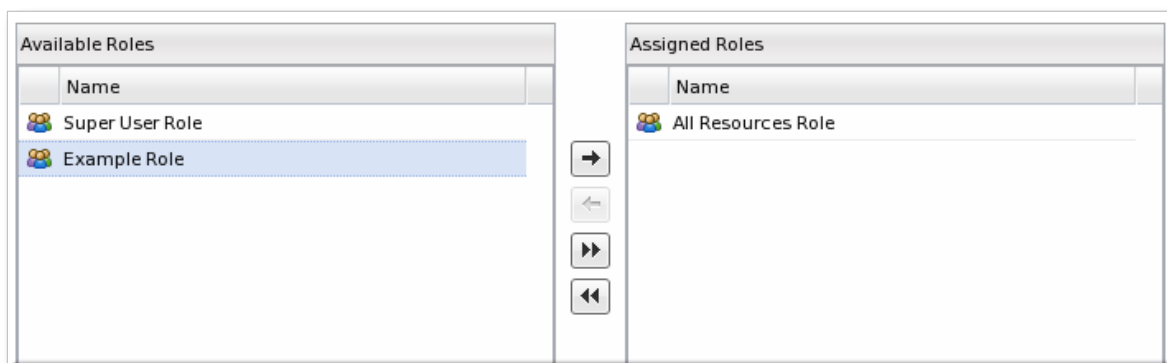
The user account can be re-enabled at any time by changing the **Enable Login** value back to **Yes**.

7.5. Changing Role Assignments for Users

1. Click the **Administration** tab in the top menu.
2. From the **Security** menu, select **Users**.



3. Click the name of the user to edit.
4. To *add* a role to a user, select the required role from the **Available Roles** area, click the arrow pointing to the **Assigned Roles** area. To *remove* a role, select the assigned role on the right and click the arrow pointing to the left.



5. Click **Save** to save the role assignments.

8. MANAGING ROLES AND ACCESS CONTROL

In JBoss Operations Network, security is implemented through rules that are set on *users and roles*. Restrictions are set on what users and roles can access and what operations they can perform.

8.1. Security in JBoss ON

Security establishes precise relationships between users, resources, and the tasks users can perform. Interactions between users and resources are ordered by including or excluding those users and resources (through groups) in defined *roles*, and then granting the role the ability to perform tasks.

8.1.1. Access Control and Permissions

When a user is allowed to perform a certain operation, that is called a *permission*. All permissions must be explicitly granted to explicit resources. If a user is not given permission to a specific resource group, then the user, by default, has no access to that group – even if the user has permission to perform a task. Likewise, if a user has access to a group but has no permissions assigned, then the user cannot perform any tasks.

Any permissions set in JBoss ON are given to a role, and the members of the role inherit those permissions. Allowing or restricting permissions is *access controls*.

In JBoss ON, there are two levels where access control is granted:

- *Global permissions* apply to JBoss ON server configuration. This covers administrative tasks, like creating users, editing roles, creating groups, importing resources into the inventory, or changing JBoss ON server properties.
- *Resource-level permissions* apply to actions that a user can perform on specific resources in the JBoss ON inventory. These cover actions like creating alerts, configuring monitoring, and changing resource configuration. Resource-level permissions are tied to the subsystem areas within JBoss ON.

All JBoss ON permissions are listed in [Table 10, “JBoss ON Access Control Definitions”](#).

For resource-level rights, read and write permissions are granted independently. A user can be granted a right to view (read) resource data without automatically being granted the right to edit that configuration. For example, any user can view the operations history of a resource or view the configured alerts for a resource within the role *even if* that role has not been given edit access to those subsystem areas.

By default, read access is enabled by default for all resource-level rights, with one exception: resource configuration. Resource configuration can be considered a security risk, so even read access is denied by default. Of course, read access, like write access, can be enabled or disabled for any resource-level permission.

Resource Permissions			
Name	Read?	Write?	Description
Inventory	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Read: (IMPLIED) view Resource properties (name, description, version, etc.), connection settings, and connection settings history Write: update Resource name, version, description, and connection settings; delete connection settings history items
Manage Measurements	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Read: (IMPLIED) view metric data and collection schedules Write: update metric collection schedules
Manage Alerts	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Read: (IMPLIED) view alert definitions and alert history Write: create, update, and delete alert definitions; acknowledge and delete alert history items
Configure	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Read: view Resource configuration and Resource configuration revision history Write: update Resource configuration; delete Resource configuration revision history items
Control	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Read: (IMPLIED) view available operations and operation execution history Write: execute operations; delete operation execution history items
Manage Events	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Read: (IMPLIED) view events


Figure 35. Read Access Option

**NOTE**


Granting a role the right to *change* something does not implicitly grant the right to *delete* something. For example, users with the configuration write permission can edit resource configuration and view configuration history and settings, but they cannot delete elements in the configuration history. Similar constraints are true for users with permission to create and edit operations and alerts – there is no right to delete elements in the resource history.

Deleting elements in the history requires the manage inventory permission.

Table 10. JBoss ON Access Control Definitions

Access Control Type	Description
Global Permissions	
Manage Security	Equivalent to a superuser. Security permissions grant the user the rights to create and edit any entries in JBoss ON, including other users, roles, and resources, to change JBoss ON server settings, and to control inventory.
	<div style="background-color: #fff9c4; padding: 10px; border: 1px solid #ccc;">  <p>WARNING</p> <p>The Security access control level is extremely powerful, so be cautious about which users are assigned it. Limit the number of superusers to as few as necessary.</p> </div>

Access Control Type	Description
Manage Inventory	Allows any operation to be performed on any JBoss ON resource, including importing new resources.
Manage Settings	Allows a user to add or modify any settings in the JBoss ON server configuration itself. This includes operations like deploying plug-ins or using LDAP authentication.
Manage Bundles	Allows a user to upload and manage bundles (packages) used for provisioning resources.
Manage Repositories	Allows a user to access any configured repository, including private repositories and repositories without specified owners. Users with this right can also associated content sources with repositories.
View Users	Allows a user to view the account details (excluding role assignments) for other users in JBoss ON.
Resource-Level Permissions	
Inventory	Allows a user to edit resource details and connection settings – meaning the information about the resource in the JBoss ON inventory. This does not grant rights to edit the resource configuration.
Manage Measurements	Allows the user to configure monitoring settings for the resource.
Manage Alerts	Allows the user to create alerts and notifications on a resource. Configuring new alert senders changes the server settings and is therefore a function of the global Settings permissions.
Control	Allows a user to run operations (which are also called <i>control actions</i>) on a resource.

Access Control Type	Description
Configure	<p>Allows users to change the configuration settings on the resource through JBoss ON.</p> <div data-bbox="815 338 922 539" style="display: inline-block; vertical-align: top;">  </div> <p>NOTE</p> <p>The user still must have adequate permissions on the resource to allow the configuration changes to be made.</p> <p>This access area has two options:</p> <ul style="list-style-type: none"> • Read, which grants read-only access to the resource configuration • Write, which grants modify access and, implicitly, read access <p>If one of these permissions is not granted to a role, then the users in the role are denied any access to the resource configuration.</p>
Manage Drift	<p>Allows the user to create, modify, and delete resource and template drift definitions. It also allows the user to manage drift information, such as viewing and comparing snapshots.</p>
Manage Content	<p>Allows the user to manage content providers and repositories that are available to resources.</p>
Create Child Resources	<p>Allows the user to manually create a child resource for the specified resource type.</p>
Delete Child Resources	<p>Allows the user to delete or uninventory a child resource for the specified resource type.</p>

8.1.2. Access and Roles

JBoss ON handles access to both resources and JBoss ON configuration through *roles*. A role has certain permissions assigned to it, meaning things that members of the role are allowed to do.

Only two types of JBoss ON identities can belong to a role: users and groups.

Users are assigned to a role to be granted those permissions. Users can be added to a role individually or be added as a member of an LDAP group.

Resource groups are assigned to a role to provide a list of resources that those users can perform actions on. Another way of looking at it is that users can only manage resources that they are expressly given access to. Roles define that access.

**NOTE**

Be sure to create resource groups to assign to any custom roles you create. If no resource groups are assigned to a role, then none of the members of the role can see any resources. Creating groups is described in [Section 5.2, “Creating Groups”](#).

One convenient feature of roles is that users can be automatically assigned to roles by assigning an LDAP group to the role ([Section 9.3.2, “Associating LDAP User Groups to Roles”](#)). All of the LDAP users who belong to that group are automatically members of the role. (This is similar to the simplicity of using LDAP user to create JBoss ON users by enabling LDAP authentication, in [Section 9.2.3, “Configuring LDAP User Authentication”](#).)

There are two roles already configured in JBoss ON by default:

- A superuser role provides complete access to everything in JBoss ON. This role cannot be modified or deleted. The user created when the JBoss ON server was first installed is automatically a member of this role.
- An *all resources* role exists that provides full permissions to every resource in JBoss ON (but not to JBoss ON administrative functions like creating users). This is a useful role for IT users, for example, who need to be able to change the configuration or set up alerts for resources managed by JBoss ON but who don't require access over JBoss ON server or agent settings.

8.2. Creating a New Role

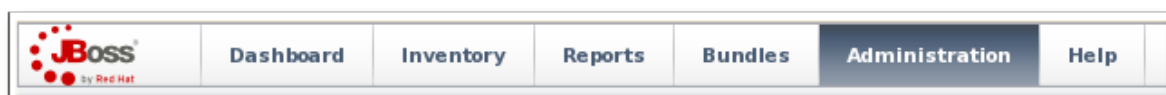
**NOTE**

Be sure to create resource groups to assign to any custom roles you create. If no resource groups are assigned to a role, then none of the members of the role can see any resources. Creating groups is described in [Section 5.2, “Creating Groups”](#).

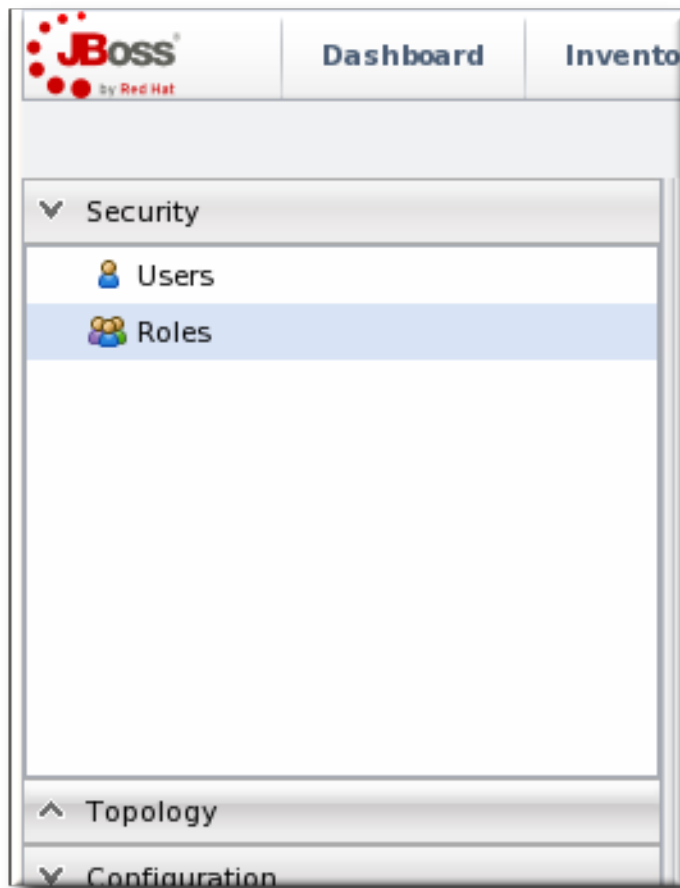
1. Create any resources groups which will be associated with the role. Creating groups is described in [Section 5.2, “Creating Groups”](#).

By default, JBoss ON uses only *resource groups* to associate with a role, and these are required. However, optional user groups from an LDAP directory can also be assigned to a role, so that the group members are automatically treated as role members. *LDAP groups* must be configured in the server settings, as described in [Section 9.3.2, “Associating LDAP User Groups to Roles”](#).

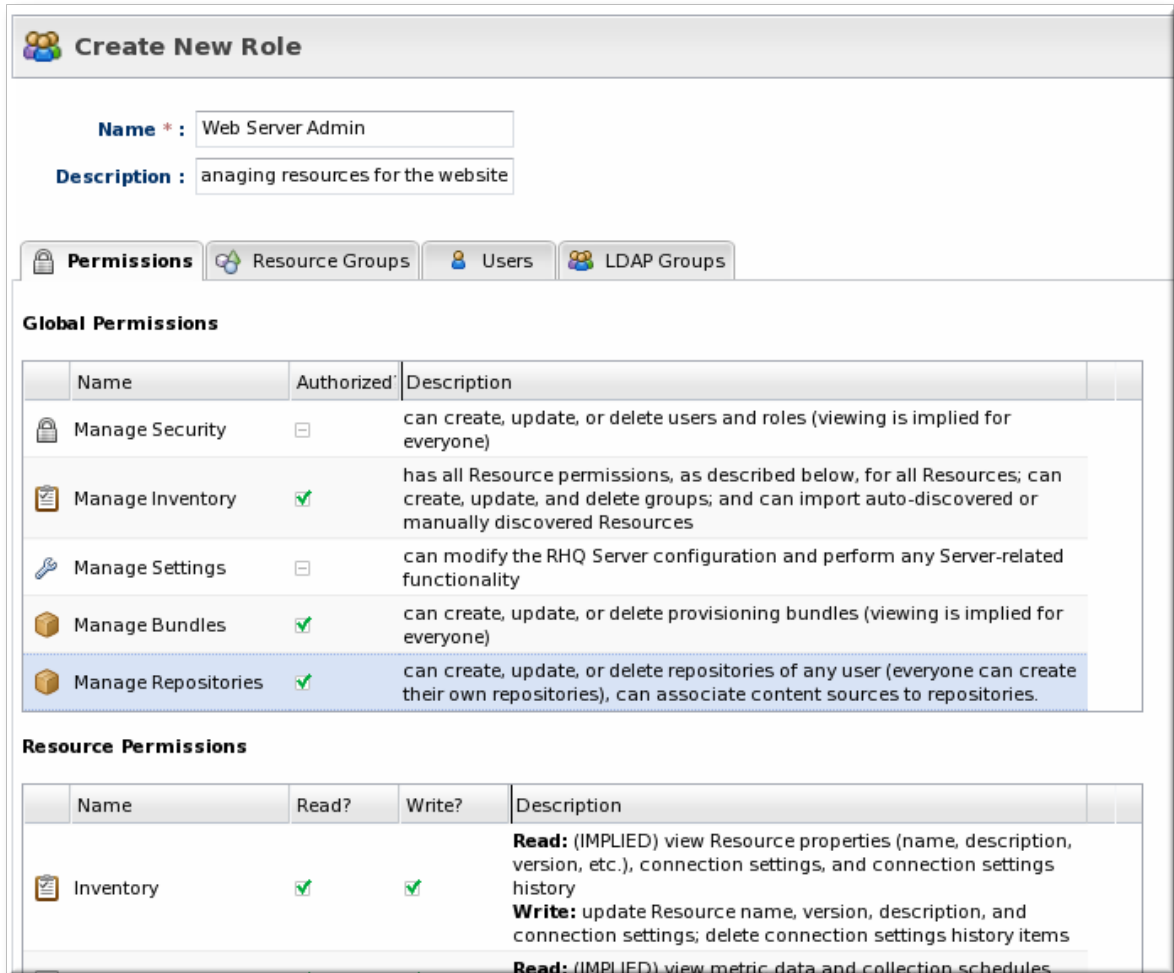
2. In the top menu, click the **Administration** tab.



3. In the **Security** menu table on the left, select the **Roles** item.

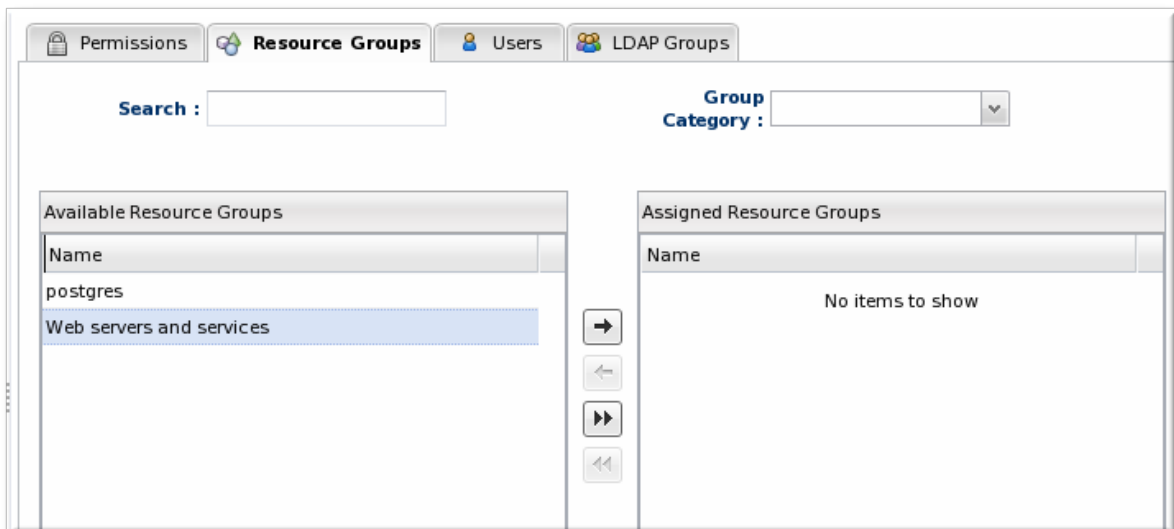


4. The list of current roles comes up in the main task window. Click the **New** button at the bottom of the list.
5. Give the role a descriptive name. This makes it easier to manage permissions across roles.
6. Set the access rights for the role in the **Permissions**. There are two categories of permissions:
 - o *Global permissions* grant permissions to areas of the JBoss ON server and configuration.
 - o *Resource permissions* grant permissions for managing resources.



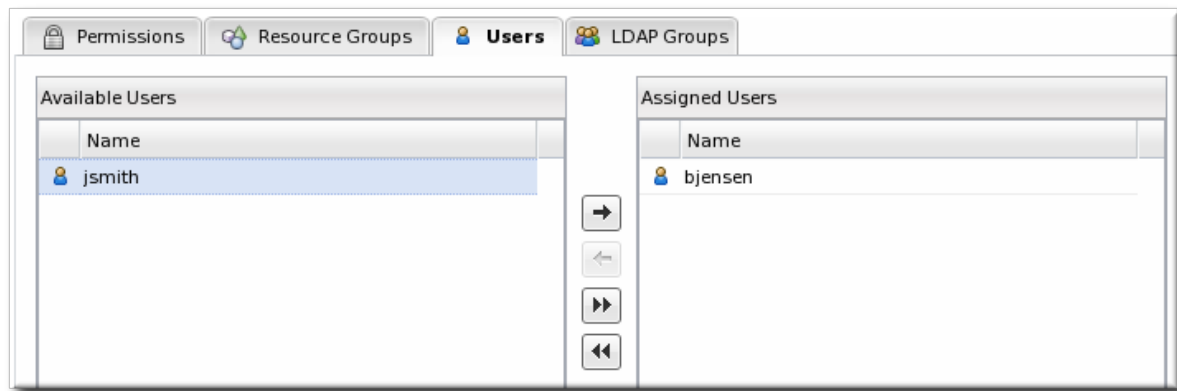
The specific access permissions are described in [Table 10, “JBoss ON Access Control Definitions”](#).

7. Select the **Resource Groups** tab to assign groups to the role.



Move the required groups from the **Available Resource Groups** area on the left to the **Assigned Resource Groups** on the right as required.

8. At the bottom, click the **Save** button.
9. Select the **Users** tab to assign users to the role.



Move the required user from the **Available Users** area on the left, to the **Assigned Users** on the right as required.

- Click the arrow in the upper right to close the create window.

8.3. Extended Example: Read-Only Access for Business Users

JBoss ON distinguishes between read permissions and write permissions. Neither read nor write access is implicit to any object or functional area in JBoss ON, which gives administrators some flexibility in where and what access is granted.

The Setup

Example Co. needs some of its management team to be able to read and access JBoss ON data to track infrastructure performance and maintenance, define incident response procedures, and plan equipment upgrades. While these business users need to view JBoss ON information, they should not be able to edit any of the configuration, which is handled by the IT and development departments.

Tim the IT Guy plans to create a special business managers role that will allow users to "see but not touch" the resource configuration.

The Plan

Tim the IT Guy first defines *what* actions the business users need to perform, and they need to be able to see everything:

- View resources in the inventory and histories for adding and deleting resources.
- View monitoring information, including measurements and events.
- View alerts.
- View content and bundles and any deployments to resources.
- View configuration drift.
- View all resource histories for configuration and operations.
- View user details to get information for auditing actions.

All of the global permissions relate to creating entries and managing configuration in JBoss ON and the inventory – which none of the business managers need to be able to do. There is one exception: the view users permission, which allows regular users to see the details of other users. That is necessary,

because many actions such as running operations and changing resource configuration list what user initiated the action. Being able to view user information is required for adequate auditing of infrastructure changes.

The default selection for roles is for all resource-level permissions to grant read access to users, with the exception of configuration rights, which have no access. Tim the IT Guy decides to grant read-only access to the configuration so that managers can check the configuration history, which could be useful for policy planning. The group has read-only access to all resources and to items like reports.

The Results

Business users are given access to all of the information they need, without being able to change any configuration or inventory accidentally.

9. INTEGRATING LDAP SERVICES FOR AUTHENTICATION AND AUTHORIZATION

JBoss ON can incorporate LDAP directories to help manage users, authentication, and membership in roles. This simplifies user management in JBoss ON and also leverages existing organizational configuration (user accounts, groups, passwords, and account lockout policies) so that JBoss ON mirrors other infrastructure configuration.



IMPORTANT

If LDAP is used for user account management, then the LDAP directory should be the authoritative source for creating and managing user accounts. Otherwise, there can be inconsistencies in role memberships, account settings, or other user account conflict. See [Section 9.2.2, “Issues Related to Using LDAP for a User Store”](#) .

9.1. Supported Directory Services

JBoss ON supports major directory servers for user authentication and group authorization:

- Red Hat Directory Server 8.1, 8.2, and 9.0
- Microsoft Active Directory 2003 and 2008

9.2. LDAP for User Authentication

9.2.1. About LDAP Authentication and Account Creation

By default, JBoss ON stores authentication information in its internal database. JBoss ON can also use an external LDAP repository to store this user information. With LDAP authentication, the JBoss ON server sends all login requests to the LDAP directory to process.

First, the JBoss ON server searches the LDAP directory for a matching username, and then it attempts to log into (bind to) the LDAP server using the given username and password. If the bind attempt is successful, then the user is successfully authenticated to the JBoss ON server.

After the JBoss ON server is configured to use LDAP for authentication, all login attempts are authenticated against the LDAP server.

**WARNING**

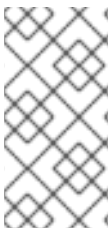
When the JBoss ON server is reconfigured to use LDAP for authentication, the LDAP information isn't validated yet. Any errors with the LDAP authentication configuration won't show up until a user attempts to log into the UI.

**NOTE**

The LDAP directory can't create JBoss ON users automatically. However, using LDAP for authentication allows new users to register themselves to JBoss ON. A new user can authenticate to JBoss ON as long as they have an LDAP account. At their first login attempt, they're redirected to a registration page which records the additional JBoss ON user information.

The JBoss ON server constructs the LDAP entry name to look for based on the JBoss ON username and information about the LDAP directory, like the parent distinguished name in the directory tree and the naming attribute used for user entries; from there, it dynamically constructs a search filter every time someone logs into JBoss ON. Custom attributes can be added to the LDAP schema, such as `JONUser=true`, which can make it easier and more precise to locate entries.

The LDAP directory *only* verifies the login credentials. The LDAP server doesn't store any other JBoss ON user data, and it doesn't create, delete, or edit entries in JBoss ON. Likewise, JBoss ON doesn't create, delete, or edit entries in the LDAP directory. The only attributes in the LDAP database that relate to JBoss ON user accounts are the username and password. Other settings in the JBoss ON user entry are stored in the JBoss ON internal database (like the user's first name and surname, email address, and role assignments).

**NOTE**

The LDAP directory is used only to check the login credentials – it doesn't store any other information about the JBoss ON users, including role assignments, and it cannot create a JBoss ON user. The JBoss ON server also cannot *create* LDAP users, so the LDAP user has to be created separately.

Because the LDAP directory doesn't store other attributes related to JBoss ON, it can't store a user certificate. This means that JBoss ON cannot use an LDAP directory for certificate-based authentication.

9.2.2. Issues Related to Using LDAP for a User Store

Integrating LDAP directories introduces another area where users can be created and managed and where the membership of roles can be changed. On the one hand, this can make managing users much easier, especially by allowing existing users to register themselves seamlessly and by automatically updating role membership. However, because users can still be created in JBoss ON and added manually to JBoss ON, user and role management can become messy.

The first problem is simply determining which datastore to use to authenticate users. Even after LDAP authentication is enabled, JBoss ON still checks credentials against its own user store – and it checks its own database first. This means that a user can authenticate to JBoss ON without that request being

sent to the LDAP database. All of the security features of the LDAP directory – particularly password policies and account inactivation – are lost because that is not the primary authentication mechanism.

Second, using two resources for user accounts introduces the problem of erroneously mapping JBoss ON and LDAP user accounts, creating duplicate entries, or allowing ghost entries. For example, John Smith is added as a user manually to JBoss ON and also has an LDAP user account. First, he has two duplicate, separately-managed user entries. Then, John Smith goes to a different division, and his LDAP entry is automatically deleted, but his JBoss ON user account remains because JBoss ON user accounts and LDAP user accounts aren't linked. He could still log into JBoss ON. Having duplicate user accounts can introduce other problems if there are accounts with identical *names*. For example, Jane Smith logs into JBoss ON with her JBoss ON user account (`jsmith`) and password, but is improperly assigned the JBoss ON role membership of LDAP user John Smith (LDAP UID `jsmith`) because her JBoss ON user ID was the same as his LDAP user ID, and her account was incorrectly mapped to his LDAP account and, therefore, his LDAP group membership.

Trying to maintain user accounts in both locations also impacts roles, at least in an administrative way. LDAP groups are added as members to the role, and then the group members are listed as user members for the role. However, the list of users assigned to the role does *not* show where those users came from. This means that the user list can be a mix of LDAP group members and JBoss ON members who were manually added to the list. Ultimately, it becomes difficult to add or remove users because it's not clear where the role users are coming from. Limiting role membership to LDAP groups simplifies maintenance; the roles are automatically updated when users are added or deleted to the groups on the LDAP side and those changes are synchronized over to the JBoss ON role dynamically.

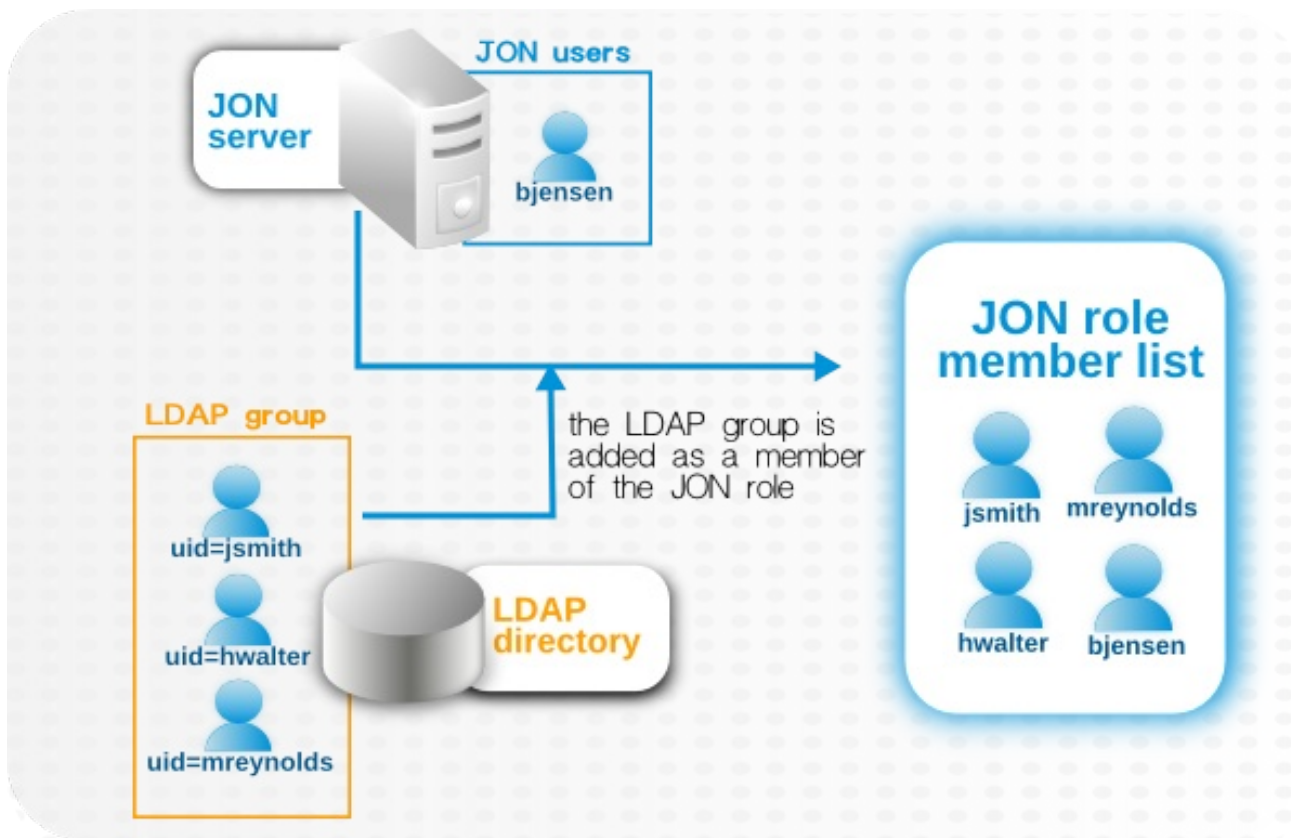


Figure 36. LDAP Groups, JBoss ON Roles, and Role Members

What all of this means is that there are three things to consider when using LDAP services for authentication or authorization:

- Only create regular user accounts in one place. If LDAP should be used for authentication, then only add or delete user accounts in the LDAP directory.

- Ideally, limit JBoss ON user accounts to special, administrative users and rely on the LDAP directory for regular accounts.
- Try to design roles around LDAP groups, meaning that JBoss ON user accounts in those roles should be limited to admin accounts or avoided altogether.

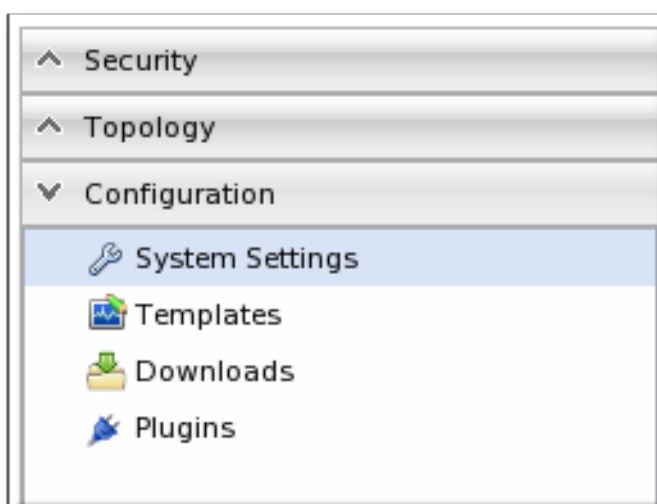
9.2.3. Configuring LDAP User Authentication

Authentication is the process of verifying the identity of an entity who attempts to access a server. JBoss ON uses simple authentication, meaning it uses simple username-password pairs to verify identity.

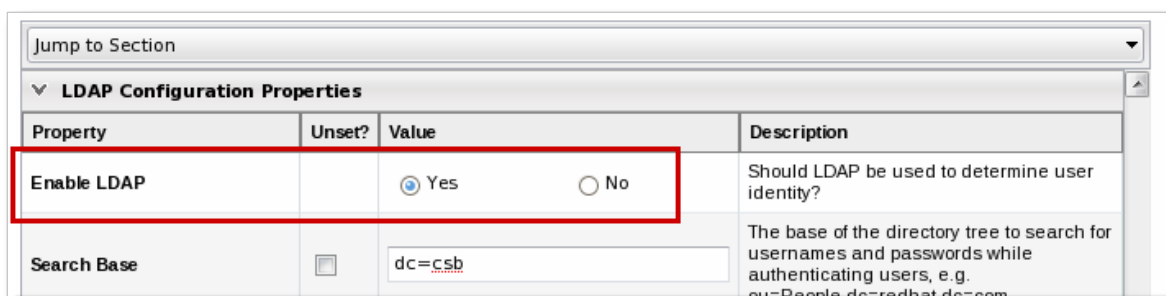
1. In the top menu, click the **Administration** tab.



2. In the **Configuration** menu table on the left, select the **System Settings** item.



3. Jump to the **LDAP Configuration Properties** area.
4. Check the **Use LDAP Authentication** checkbox so that JBoss ON will use the LDAP user directory as its identity store.



5. Configure the connection settings to connect to the specific LDAP directory.

Property	Unset?	Value	Description
Enable LDAP	<input type="checkbox"/>	<input checked="" type="radio"/> Yes <input type="radio"/> No	Should LDAP be used to determine user identity?
Search Base	<input type="checkbox"/>	dc=example,dc=com	The base of the directory tree to search for usernames and passwords while authenticating users, e.g. ou=People,dc=redhat,dc=com
Username	<input type="checkbox"/>	cn=directory manager	The username to connect to the LDAP server when querying the LDAP user database. This is typically the full LDAP distinguished name (DN) of a manager user, e.g. cn=Manager,dc=redhat,dc=com
Password	<input type="checkbox"/>	The credentials of the user used to connect to the LDAP server when querying the LDAP user database.
			Any additional filters to apply when doing the LDAP search. This is useful if the

- o Give the LDAP URL of the LDAP server. This has the format `ldap://hostname[:port]`. For example:

```
ldap://server.example.com:389
```

By default, this connects to the localhost over port 389 (standard LDAP port) or 636 (secure LDAP port, if SSL is selected).

- o To use a secure connection, check the **Use SSL** checkbox. When using SSL, make sure that the LDAP directory is actually running over SSL, and make sure that the connection URL points to the appropriate SSL port and protocol:

```
ldaps://server.example.com:636
```

- o Give the bind credentials to use to connect to the server. The username is the full LDAP distinguished name of the user as whom JBoss ON binds to the directory.



NOTE

The user *must* exist in the LDAP directory before configuring the LDAP settings in JBoss ON. Otherwise, login attempts to the JBoss ON server will fail.

Also, make sure that the JBoss ON user has appropriate read and search access to the user and group subtrees in the LDAP directory.

6. Set the search parameters that JBoss ON uses when searching the LDAP directory for matching user entries.

- o The *search base* is the point in the directory tree where the server begins looking for entries. If this is used only for user authentication or if all JBoss ON-related entries are in the same subtree, then this can reference a specific subtree:


```
ou=user,ou=JON,dc=example,dc=com
```

If the users or groups are spread across the directory, then select the base DN:

```
dc=example,dc=com
```

- Optionally, set a search filter to use to search for a specific subset of entries. This can improve search performance and results, particularly when all JBoss ON-related entries share a common LDAP attribute, like a custom *JonUser* attribute. The filter can use wild cards (`objectclass=*`) or specific values (`JonUser=true`).
- Set the LDAP naming attribute; this is the element on the farthest left of the full distinguished name. For example, in `uid=jsmith,ou=people,dc=example,dc=com`, the far left element is `uid=jsmith`, and the naming attribute is *uid*.

The default naming attribute in Active Directory is *cn*. In Red Hat Directory Server, the default naming attribute is *uid*.

7. Save the LDAP settings.



NOTE

The **Group Filter** and **Member Property** fields aren't required for user authentication. They're used for configuring LDAP groups to be assigned to roles, as in [Section 9.3.2, “Associating LDAP User Groups to Roles”](#).

9.3. Roles and LDAP User Groups

9.3.1. About Group Authorization

Many LDAP directories already contain organizational groups with users who will need to access resources in JBoss ON. Configuring JBoss ON to connect to these directories allows JBoss ON to assign LDAP groups to roles and then pull in those member lists dynamically, so the roles are populated with pre-existing member lists. All of the LDAP users automatically inherit the permissions of that role.

In the role details page, these LDAP user groups are separated from the resource groups, so it's easy to distinguish which types of group are being added to the role.

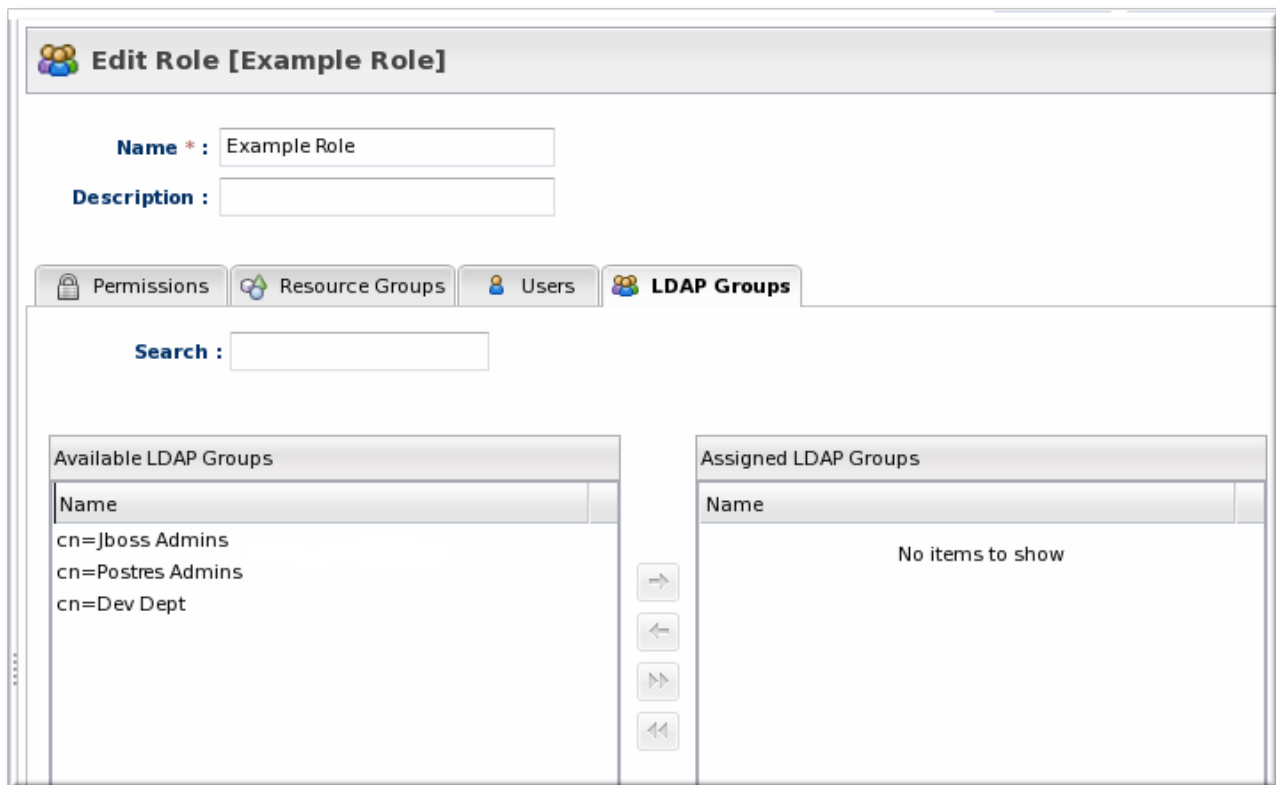


Figure 37. Groups Assigned to a Role

JBoss ON determines what LDAP groups a user belongs to with a simple search. Whenever a user logs into JBoss ON and an LDAP connection is configured, JBoss ON maps that JBoss ON username to a user entry in the LDAP directory server. The specific LDAP distinguished name (DN) for the user is used as part of a search to find matching member attributes in LDAP group entries. That is, the LDAP server can check the member lists in group entries to see what groups the person with that DN belongs to.

For LDAP groups to be added to roles, three things are required:

- An LDAP directory server connection has to be configured.
- There has to be an LDAP attribute given to search for *group entries*.

For Active Directory, this is generally the **group** object class. For Red Hat Directory Server, this is generally **groupOfUniqueNames**. Other standard object classes are available, and it is also possible to use a custom, even JBoss ON-specific, object class.

- There has to be an LDAP attribute given to identify *members* in the group.

Common member attributes are *member* and *uniqueMember*.

JBoss ON constructs an LDAP search based on the group object class and member attribute in the server configuration, plus the DN of the user given when the user logs in.

```
( &(group_filter)(member_attribute=user_DN) )
```

For example, this looks for the *member* attribute on an Active Directory group:

```
ldapsearch -h server.example.com -x -D
"cn=Administrator,cn=Users,dc=example,dc=com" -W -b "dc=example,dc=com" -x
'(&(objectclass=group)(member=CN=John Smith,CN=Users,DC=example,DC=com))'
```

Red Hat Directory Server uses the *uniqueMember* attribute on **groupOfUniqueNames** groups more commonly than *member* and *group*. For example:

```
/usr/lib64/mozldap6/ldapsearch -D "cn=directory manager" -w secret -p 389
-h server.example.com -b "ou=People,dc=example,dc=com" -s sub "&
(objectclass=groupOfUniqueNames)
(uniqueMember=uid=jsmith,ou=People,dc=example,dc=com)"
```

This search returns a list of all groups to which the user is a member. If any of these LDAP groups is assigned to a JBoss ON role, then that user is also automatically a member of that JBoss ON role.



NOTE

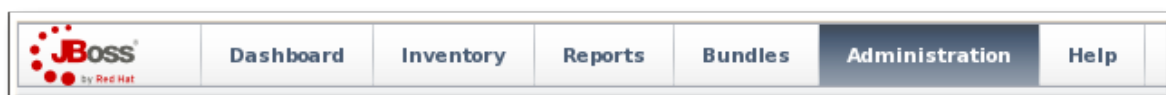
Using custom LDAP group object classes can allow you to be very specific about which groups to use for JBoss ON roles.

9.3.2. Associating LDAP User Groups to Roles

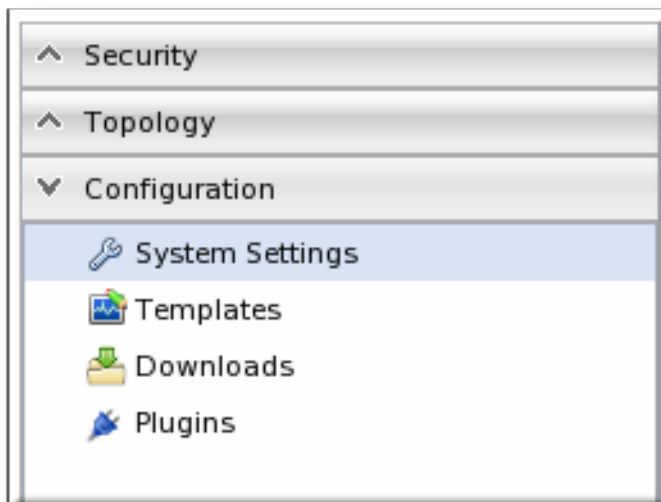
Section 9.2.3, “Configuring LDAP User Authentication” describes how an LDAP directory server can be used to authenticate users to JBoss ON. Any time a user attempts to log in, that request – with the username and password – is simply forwarded to the specified LDAP directory server to see if the credentials are correct.

Members of LDAP groups can be pulled in, automatically, as members of JBoss ON roles. The LDAP group is associated with a JBoss ON role and then the group members are *authorized* to do whatever the JBoss ON role is configured to allow. Any changes made in the LDAP group members are automatically reflected in JBoss ON, without having to edit the JBoss ON role.

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



2. In the **Configuration** menu table on the left, select the **System Settings** item.



3. Jump to the **LDAP Configuration Properties** area.
4. Set up the LDAP connections, as described in [Section 9.2.3, “Configuring LDAP User Authentication”](#). It is not required that the LDAP directory be used as the identity store in order to configure LDAP authorization, but it is recommended.
5. Set the parameters to use for the server to use to search for LDAP groups and their members.

The search filter that JBoss ON constructs looks like this:

```
(&(group_filter)(member_attribute=user_DN))
```

- o The **Group Search Filter** field sets how to search for the group entry. This is usually done by specifying the type of group to search for through its object class:

```
(objectclass=groupOfUniqueNames)
```

- o The **Group Member Filter** field gives the attribute that the specified group type uses to store member distinguished names. For example:

```
uniqueMember
```

The *user_DN* is dynamically supplied by JBoss ON when a user logs into the UI.

6. Save the LDAP settings.

9.4. Extended Example: memberOf and LDAP Configuration

The Setup

Authentication is the process of verifying someone's identity. *Authorization* is the process of determining what access permissions that identity has. Users are authorized to perform tasks based on the permissions granted to their role assignments.

All of the users and identities for Example Co. are stored in a backend Red Hat Directory Server LDAP database. To maintain a single, central user store, Tim the IT Guy wants to use existing LDAP users in JBoss ON and to determine user access to JBoss ON based on group membership, so, fundamentally, both authentication and authorization rules are determined by the LDAP configuration.

The Plan

There are two things to configure: how to identify users for authentication and how to organize users for authorization.

Groups are going to play a two-fold part in managing the LDAP configuration for JBoss ON for Example Co.:

- A single group to identify JBoss ON users in the LDAP directory
- Multiple, existing LDAP groups which are used to determine different levels of access to JBoss ON

The first thing that Tim the IT Guy determines is the way to identify users. As [Section 9.2.3, “Configuring LDAP User Authentication”](#) describes, JBoss ON identifies users to authenticate based on the results of an LDAP search, which uses a search base and optional search filter. The search filter specifies an *attribute=value* pair. One recommended method for identifying users is to create custom schema elements, like **JONUser**, which make it easy to search for matching users.

However, Tim the IT Guy has limited administrative access to the Red Hat Directory Server database. He has the ability to create groups and manage membership, but he cannot edit the schema. With no way to create an attribute that flags JBoss ON users, Tim the IT Guy has to use other configuration. Depending on the layout of the directory, he can use other kinds of configuration: views, manager, a class of service (CoS) virtual attribute, or group membership.

Using group membership is a good way to manage user assignments easily and dynamically while only having to manage a single entry (instead of individual group entries). In Directory Server, the *memberOf* attribute is automatically added to user entries to indicate a group that the user belongs to.

What Tim the IT Guy can do is set up a special group for all JBoss ON users, and then whatever users he likes. Because the Directory Server automatically adds and removes the *memberOf* attribute to user entries as members are added and removed to the group. Tim the IT Guy only has to use the *memberOf* attribute on those user accounts as the search filter for authentication.

```
dn: uid=jsmith,ou=people,dc=example,dc=com
uid: jsmith
cn: John Smith
... 8< ...
memberOf: cn=JON User Group,ou=groups,dc=example,dc=com
memberOf: cn=IT Administrators,ou=groups,dc=example,dc=com
```

The JBoss ON LDAP authentication search filter, then, would target the *memberOf* attribute for that specific JBoss ON group:

```
memberOf='cn=JON User Group,ou=groups,dc=example,dc=com'
```

Using groups for access control requires an entirely different set of group definitions, which do not have to be JBoss ON-specific. These groups relate to functional areas within Example Co., and Tim the IT Guy can map existing LDAP groups to JBoss ON roles. There are three relevant LDAP groups for Example Co. for managing JBoss ON:

- IT Administrators Group is mapped to a role with manage inventory permissions.
- IT Manager Group is mapped to a role with view (but no write) permissions for all of the resources and with view users permissions.

- Business Manager Group is mapped to a role with permissions to read all resource configuration, bundles, drift, measurements, operations, and alerts, but no write permissions.

The Results

Tim the IT Guy only has to create and manage one LDAP group, the JON Users Group, to set up all authentication and users for JBoss ON. He does not have to change the LDAP schema or even modify user entries directly.

For authorization, Tim the IT Guy designs JBoss ON roles around the functional groups already defined in the LDAP directory, in Example Co.'s organization, groups for IT admins, IT managers, and business managers and the level of access each requires.

As LDAP users authentication to the JBoss ON UI for the first time, they set up their own JBoss ON user details. After authenticating, they are automatically granted the appropriate level of access based on their LDAP group membership.

10. DOCUMENT INFORMATION

This guide is part of the overall set of guides for users and administrators of JBoss ON. Our goal is clarity, completeness, and ease of use.

10.1. Giving Feedback

If there is any error in this *Initial Setup: the Resource Inventory, Groups, and Users* or there is any way to improve the documentation, please let us know. Bugs can be filed against the documentation for the community-based RHQ Project in Bugzilla, <http://bugzilla.redhat.com/bugzilla>. Make the bug report as specific as possible, so we can be more effective in correcting any issues:

1. Select the **JBoss** products group.
2. Select **JBoss Operations Network** from the list.
3. Set the component to **Documentation**.
4. Set the version number to 3.1.2.
5. For errors, give the page number (for the PDF) or URL (for the HTML), and give a succinct description of the problem, such as incorrect procedure or typo.

For enhancements, put in what information needs to be added and why.

6. Give a clear title for the bug. For example, "**Incorrect command example for setup script options**" is better than "**Bad example**".

We appreciate receiving any feedback – requests for new sections, corrections, improvements, enhancements, even new ways of delivering the documentation or new styles of docs.

10.2. Document History

Revision 3.1.2-2.400 Rebuild with publican 4.0.0	2013-10-31	Rüdiger Landmann
Revision 3.1.2-2	January 23, 2013	Ella Deon Lackey

Fixing improper example for using --nostart to start the agent prompt, Bugzilla 895186.

Revising the deleting/uninventory sections to clarify and correct what happens to the underlying resource and what happens to the inventory entry for each operation, Bugzilla 865918.

Revision 3.1.1-1**September 19, 2012****Ella Deon Lackey**

Bug fixing for JBoss Operations Network 3.1.1.

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

Initial release of JBoss Operations Network 3.1.

INDEX

A

access controls, [Security in JBoss ON](#)

about read rights, [Access Control and Permissions](#)

applying to resources, [Creating a New Role](#)

assigning to users, [Changing Role Assignments for Users](#)

list of permissions, [Access Control and Permissions](#)

users, [Creating User Accounts](#)

authentication

and LDAP, [Configuring LDAP User Authentication](#)

configuring LDAP, [Configuring LDAP User Authentication](#)

storing credentials, [LDAP for User Authentication](#)

authorization

for LDAP groups and roles, [Roles and LDAP User Groups](#)

B

boundary characters

in searches, [Basic String Searches](#)

D

discovery

ignoring resources, [Ignoring Discovered Resources](#)

importing, [Importing Resources from the Discovery Queue](#)

manual, [Running Discovery Scans Manually](#)

resources, [Importing Resources from the Discovery Queue](#)

dynamic search

groups and resources, [Dynamic Searches for Resources and Groups](#)

null, [Basic String Searches](#)

saving, reusing, and deleting, [Saving, Reusing, and Deleting Dynamic Searches](#)

syntax, [About the Dynamic Search Syntax](#)

using quotation marks, [Basic String Searches](#)

G

groups

and roles, [Access and Roles](#)

assigning to roles, [Creating a New Role](#)

dynamic search, [Dynamic Searches for Resources and Groups](#)

empty groups, [General Expression Syntax](#)
overview, [Managing Groups](#)
search context, [Property Searches](#)

I

importing

discovery, [Importing Resources from the Discovery Queue](#)

inventory

importing, [Managing the Resource Inventory](#)
overview, [About the Inventory: Resources](#)
reports, [Viewing Inventory Summary Reports](#)

J

JBoss ON

access control, [Security in JBoss ON](#)
and LDAP, [Configuring LDAP User Authentication](#)
 configuring, [Configuring LDAP User Authentication](#)

authorization, [Roles and LDAP User Groups](#)
supported LDAP servers, [About LDAP Authentication and Account Creation](#)

L

LDAP

assigning groups to roles, [Access and Roles](#)
authorization, [Roles and LDAP User Groups](#)
for authentication, [Configuring LDAP User Authentication](#)
 configuring, [Configuring LDAP User Authentication](#)
 configuring SSL, [Configuring LDAP User Authentication](#)

supported servers, [About LDAP Authentication and Account Creation](#)
user groups and roles, [Associating LDAP User Groups to Roles](#)
 LDAP group object classes, [About Group Authorization](#)
 member attributes, [About Group Authorization](#)
 search parameters, [About Group Authorization](#)

verifying credentials, [About LDAP Authentication and Account Creation](#)

P

permissions, [Security in JBoss ON](#)

applying to groups, [Creating a New Role](#)
assigning to users, [Changing Role Assignments for Users](#)

list of, [Access Control and Permissions](#)

R

reports

inventory, [Viewing Inventory Summary Reports](#)

resources

access control, [Security in JBoss ON](#)

access permissions on, [Access Control and Permissions](#)

and managing inventory, [Managing the Resource Inventory](#)

and roles, [Access and Roles](#)

creating children, [Creating Child Resources](#)

deleting entries, [Deleting Entries](#)

discovery and imports, [Importing Resources from the Discovery Queue](#)

dynamic search, [Dynamic Searches for Resources and Groups](#)

groups, [Managing Groups](#)

ignoring discovered resources, [Ignoring Discovered Resources](#)

inventory reports, [Viewing Inventory Summary Reports](#)

managed, [Managed Resources: Platforms, Servers, and Services](#)

manual discovery, [Running Discovery Scans Manually](#)

running discovery, [Importing Resources from the Discovery Queue](#)

search context, [Property Searches](#)

uninventory, [Uninventorying and Deleting Resources](#)

roles, [Access and Roles](#)

adding or removing users, [Changing Role Assignments for Users](#)

and groups, [Access and Roles](#)

and LDAP user groups, [Associating LDAP User Groups to Roles](#)

LDAP group object classes, [About Group Authorization](#)

member attributes, [About Group Authorization](#)

searching for members, [About Group Authorization](#)

and users, [Access and Roles](#)

assigning groups, [Creating a New Role](#)

assigning users from LDAP groups, [Access and Roles](#)

creating, [Creating a New Role](#)

default, [Access and Roles](#)

setting permissions, [Creating a New Role](#)

types of members, [Access and Roles](#)

S

search

- groups context, [Property Searches](#)
- resource context, [Property Searches](#)
- string operators, [Property Searches](#)
- using quotation marks, [Basic String Searches](#)

secure connections

- using for LDAP authentication, [Configuring LDAP User Authentication](#)

security

- users, [Creating User Accounts](#)

self-registering

- and LDAP, [About LDAP Authentication and Account Creation](#)

server

- access control, [Security in JBoss ON](#)

 - global rights, [Access Control and Permissions](#)

 - resource-level rights, [Access Control and Permissions](#)

- and LDAP groups for roles, [Associating LDAP User Groups to Roles](#)

 - building LDAP search, [About Group Authorization](#)

 - LDAP group object classes, [About Group Authorization](#)

 - member attributes, [About Group Authorization](#)

- LDAP authentication, [Configuring LDAP User Authentication](#)

 - configuring, [Configuring LDAP User Authentication](#)

SSL

- using for LDAP authentication, [Configuring LDAP User Authentication](#)

U

UI

- A Tour of the UI, [Using the JBoss ON Web Interface](#)

- access control, [Security in JBoss ON](#)

- dashboard, [Dashboard](#)

- deleting entries, [Deleting Entries](#)

- details page, [Entry Details Pages](#)

- inventory, [Inventory Browsers and Summaries](#)

- left menu, [The Left Menu](#)

- logging in, [Logging into the JBoss ON Web UI](#)

- setting favorites, [Setting Favorites](#)

users

- and roles, [Access and Roles](#)

assigning LDAP user groups to roles, [Access and Roles](#)
authentication, [Configuring LDAP User Authentication](#)
changing access controls, [Changing Role Assignments for Users](#)
changing roles, [Changing Role Assignments for Users](#)
configuring LDAP authentication, [Configuring LDAP User Authentication](#)
creating new, [Creating a New User](#)
disabling accounts, [Disabling User Accounts](#)
editing details, [Editing User Entries](#)
security, [Creating User Accounts](#)
using LDAP to self-register, [About LDAP Authentication and Account Creation](#)

[1] Provisioning Ant bundles is implemented through an agent plug-in which performs the tasks on the platform and a server-side plug-in which manages the bundles in the server.

[2] The excluded process can still be discovered by another agent plug-in.