
JBoss Operations Network 3.0

Initial Setup: the Resource Inventory, Groups, and Users

for initially setting up resources and security in JBoss ON



Copyright © 2011 Red Hat, Inc..

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at <http://creativecommons.org/licenses/by-sa/3.0/>. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

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, MetaMatrix, 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.

All other trademarks are the property of their respective owners.

December 7, 2011

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.

1. Using the JBoss ON Web Interface	3
1.1. Logging into the JBoss ON Web UI	3
1.2. A High Level Walk-Through	4
1.2.1. The Top Menu	5
1.2.2. The Left Menu	6
1.2.3. Dashboard	8
1.2.4. Inventory Browsers and Summaries	8
1.2.5. Entry Details Pages	9
1.2.6. Shortcuts in the UI	12
1.3. Getting Notifications in the Message Center	13
1.4. Sorting and Changing Table Displays	13
1.5. Tagging Entries	15
1.5.1. Adding Tags	15
1.5.2. Viewing Tags	16
1.5.3. Deleting Tags	16
1.6. Customizing the Dashboard	17
1.6.1. Editing Portlets	17
1.6.2. Adding and Editing Dashboards	17
1.7. Setting Favorites	19
1.8. Deleting Entries	20
2. Dynamic Searches for Resources and Groups	20
2.1. About Search Suggestions	21
2.2. About the Dynamic Search Syntax	22
2.2.1. Basic String Searches	23
2.2.2. Property Searches	25
2.2.3. Complex AND and OR Searches	27
2.3. Saving, Reusing, and Deleting Dynamic Searches	28
3. Managing the Resource Inventory	29
3.1. About the Inventory: Resources	29
3.1.1. Managed Resources: Platforms, Servers, and Services	30
3.1.2. Resources in the Inventory Used by JBoss ON	31
3.2. Discovering Resources	31
3.2.1. Finding New Resources: Discovery	32
3.2.2. Running Discovery Scans Manually	32
3.2.3. Importing Resources from the Discovery Queue	34
3.2.4. Ignoring Discovered Resources	35
3.3. Importing New Resources Manually	36
3.4. Configuring Tomcat/EWS Servers for Discovery (Windows)	38
3.5. Creating Child Resources	38
3.6. Viewing and Editing Resource Information	40
3.7. Removing Resources from the Inventory	41
3.7.1. Uninventorying through the Parent Inventory	42
3.7.2. Uninventorying through a Group Inventory	43
3.8. Getting Inventory Reports	44
4. Managing Groups	45
4.1. About Groups	45
4.1.1. Dynamic and Static Groups	46
4.1.2. About Autogroups	46

4.1.3. Comparing Compatible and Mixed Groups	47
4.2. Creating Dynamic Groups	49
4.2.1. About Dynamic Groups Syntax	49
4.2.2. Creating Dynamic Groups	54
4.2.3. Recalculating Group Members	56
4.3. Creating Groups	57
4.4. Changing Group Membership	59
4.5. Editing Compatible Group Connection Properties	61
5. Creating User Accounts	63
5.1. Managing the rhqadmin Account	63
5.2. Creating a New User	64
5.3. Editing User Entries	66
5.4. Disabling User Accounts	66
5.5. Changing Role Assignments for Users	67
6. Managing Roles and Access Control	68
6.1. About Security in JBoss ON: Roles and Access Control	68
6.2. Creating a New Role	71
6.3. Editing Roles	74
7. Integrating LDAP Services for Authentication and Authorization	75
7.1. Supported Directory Services	75
7.2. How JBoss ON Uses LDAP for Authentication	76
7.3. Configuring LDAP User Authentication	78
7.4. How JBoss ON Roles Work with LDAP User Groups	81
7.5. Associating LDAP User Groups to Roles in JBoss ON	82
8. Document Information	84
8.1. Giving Feedback	84
8.2. Document History	84
Index	84

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. 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**.

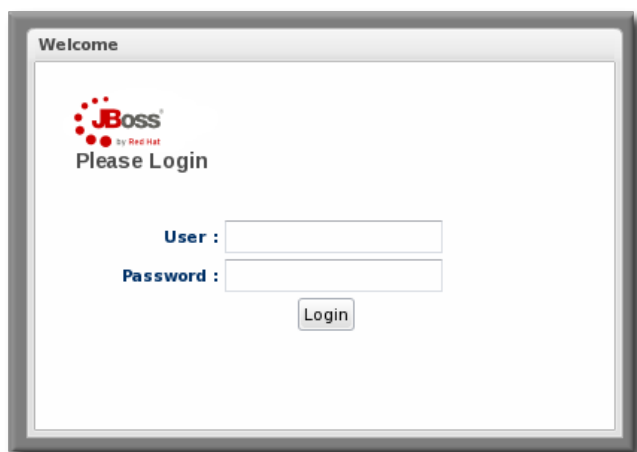


Figure 1. Logging into JBoss ON



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.

1.2. 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.

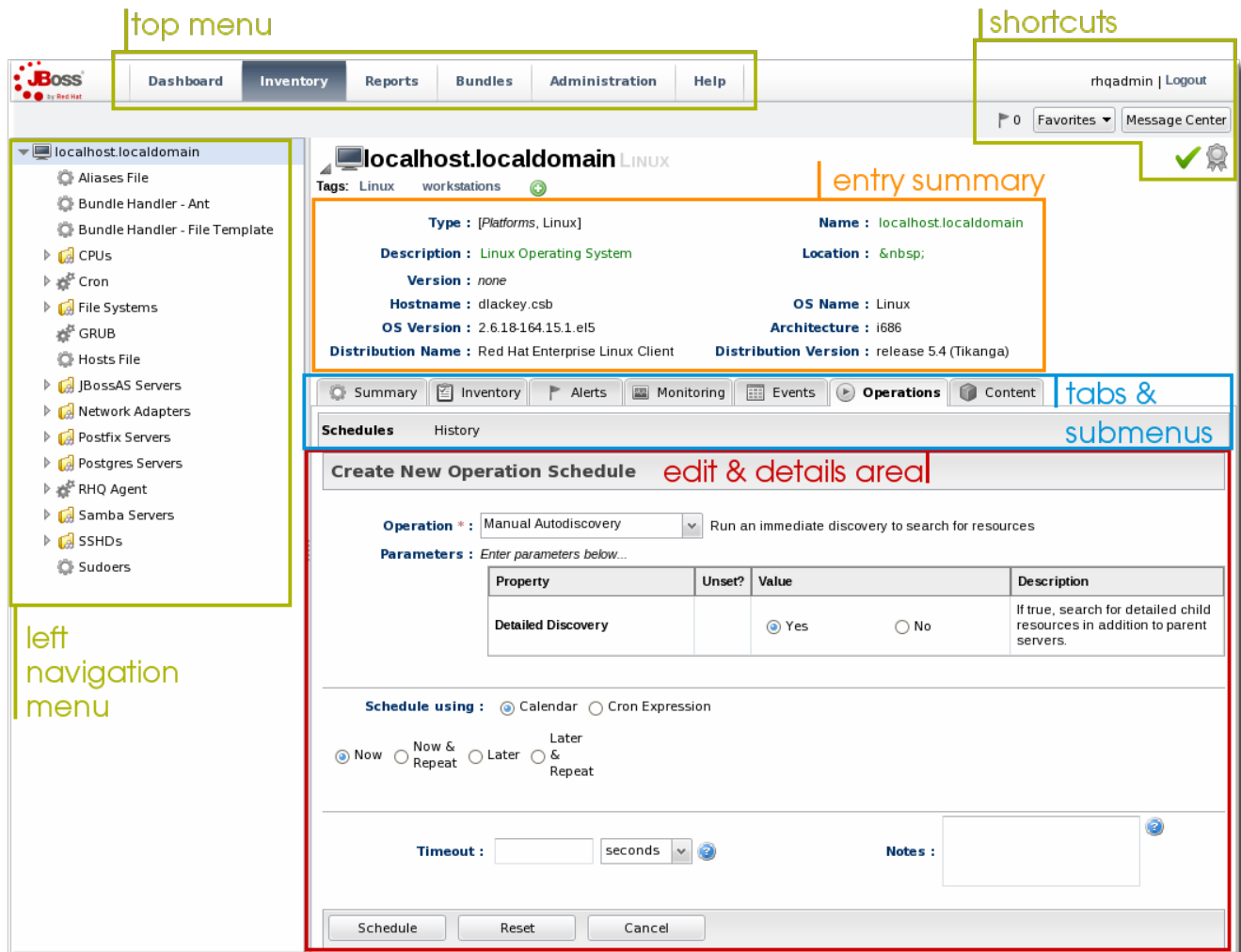


Figure 2. UI Elements All Together

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.

1.2.1. 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.

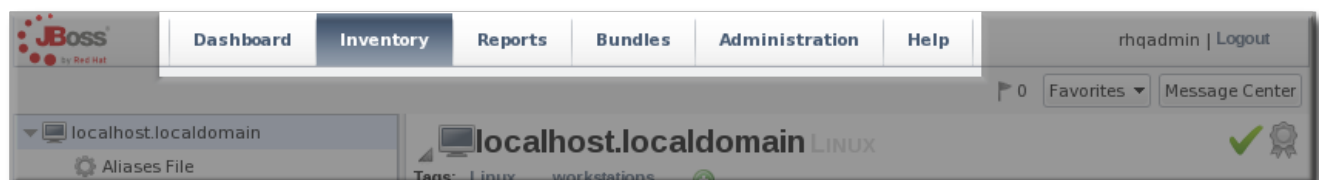


Figure 3. The Top Menu

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.

1.2.2. 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”](#).

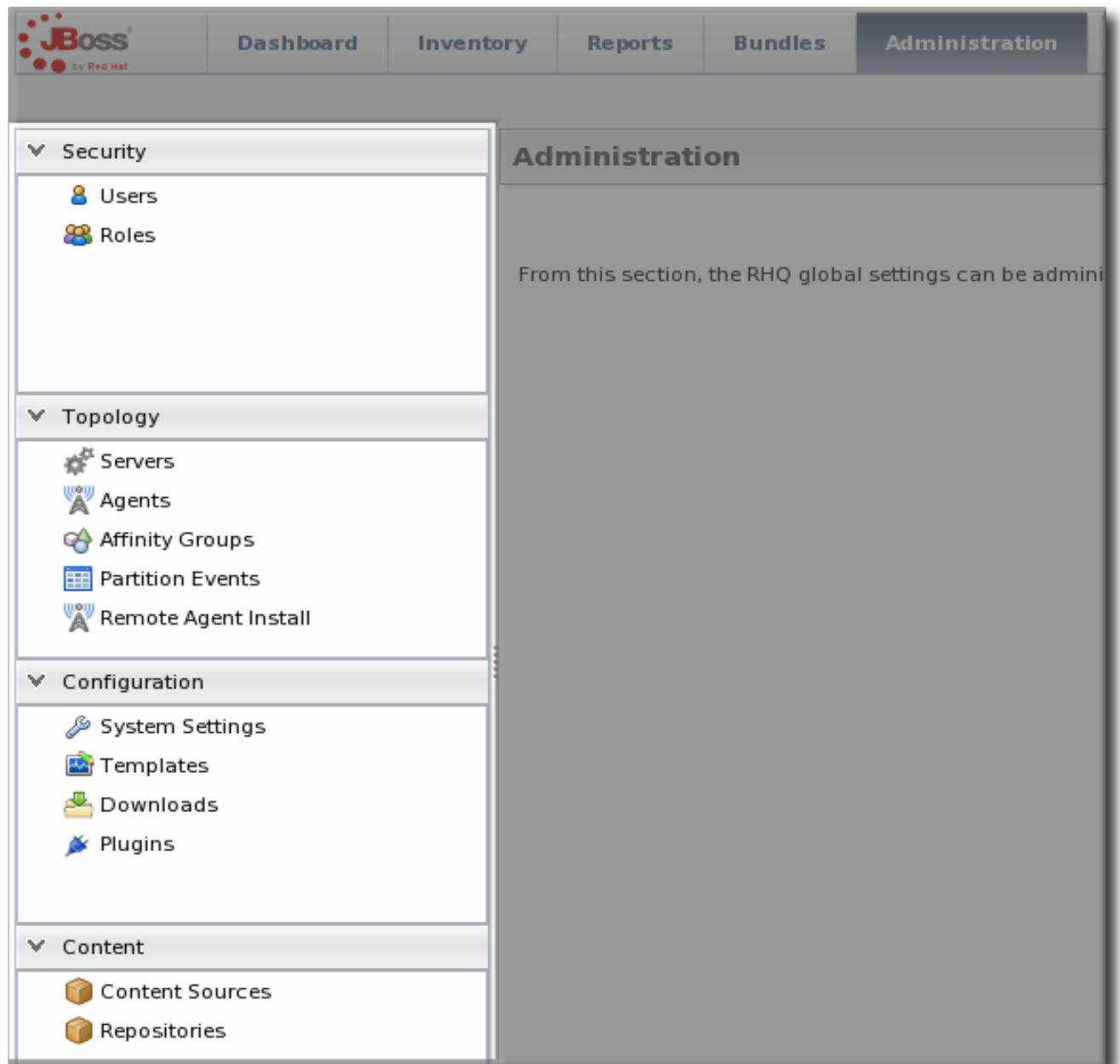


Figure 4. The Left Menu

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.

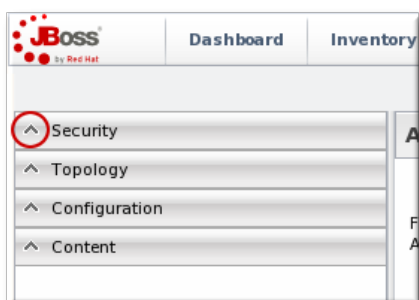


Figure 5. Collapsing the Left Menu

1.2.3. 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 screenshot displays the JBoss ON Dashboard interface. At the top, there is a navigation bar with tabs for Dashboard, Inventory, Reports, Bundles, Administration, and Help. The user 'rhqadmin' is logged in. The dashboard is divided into several portlets:

- Inventory Summary:** Displays various totals: Platform Total: 1, Server Total: 12, Service Total: 509, Compatible Group Total: 0, Mixed Group Total: 0, Group Definition Total: 0, and Average Metrics per Minute: 194.
- Recent Alerts:** A table with columns: Creation Time, Name, Condition Text, Priority, Status, Resource, and Ancestry. It shows 'No results found using specified criteria.'
- Discovery Queue:** A table with columns: Resource Name, Resource Key, Type, and Inventory Status. It lists resources like 'localhost.localdomain' and 'redhat0' with their respective paths and statuses.
- Alerted or Unavailable Resources:** A table with columns: Resource, Ancestry, Alerts, and Availability. It shows 'redhat0' with 0 alerts.
- Recent Operations:** A table with columns: Date Submitted, Operation, Request, Status, Resource, and Ancestry. It shows '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.

1.2.4. 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.

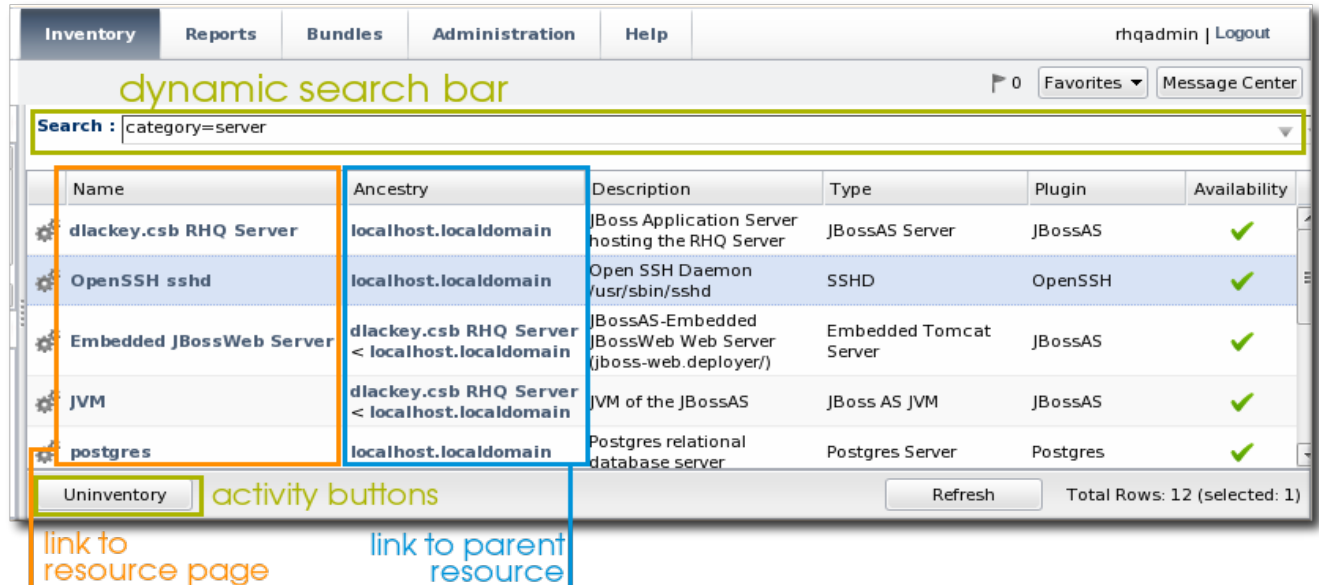


Figure 7. Inventory Browser

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.

1.2.5. 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.

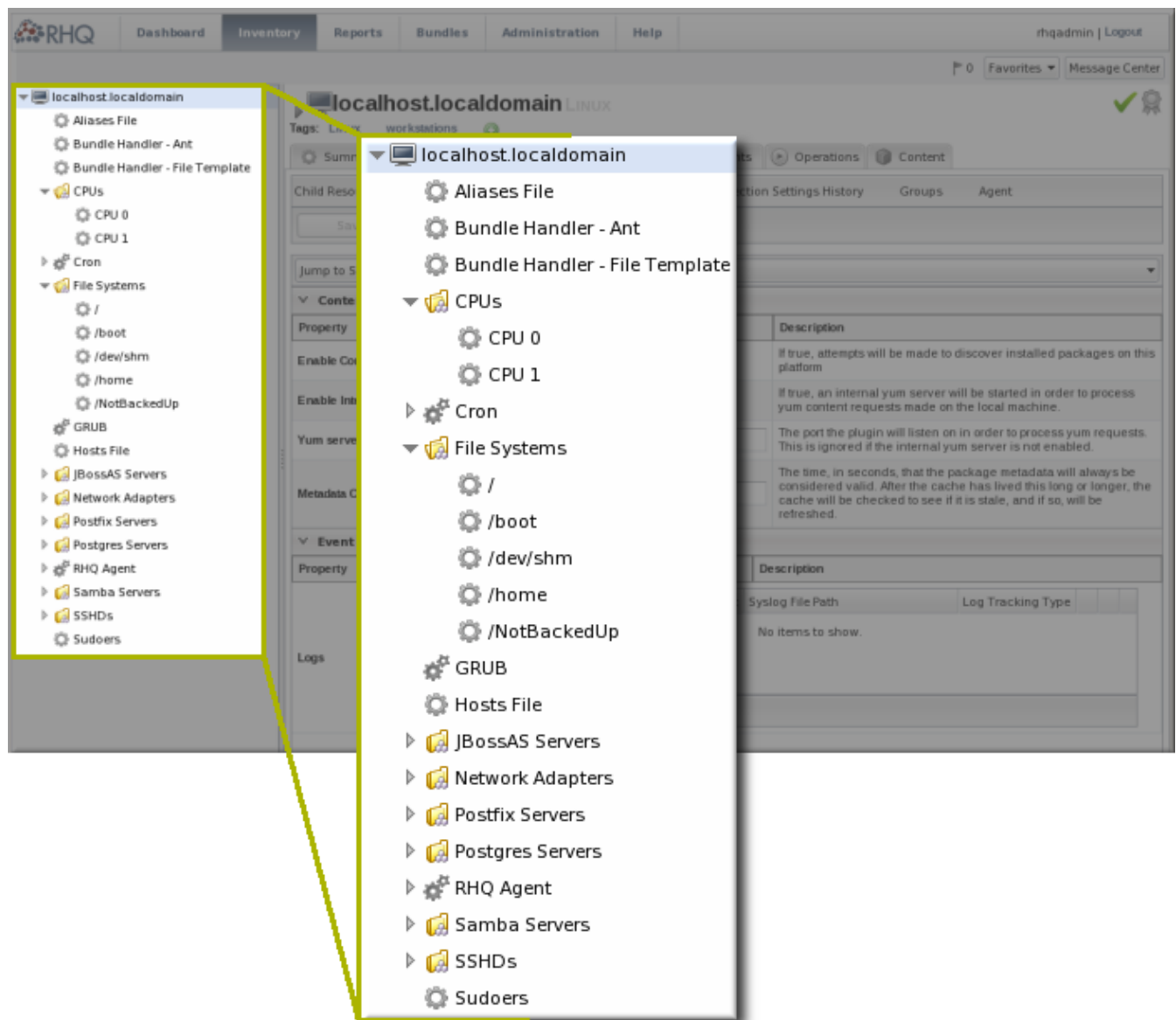


Figure 8. Resource Tree

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).

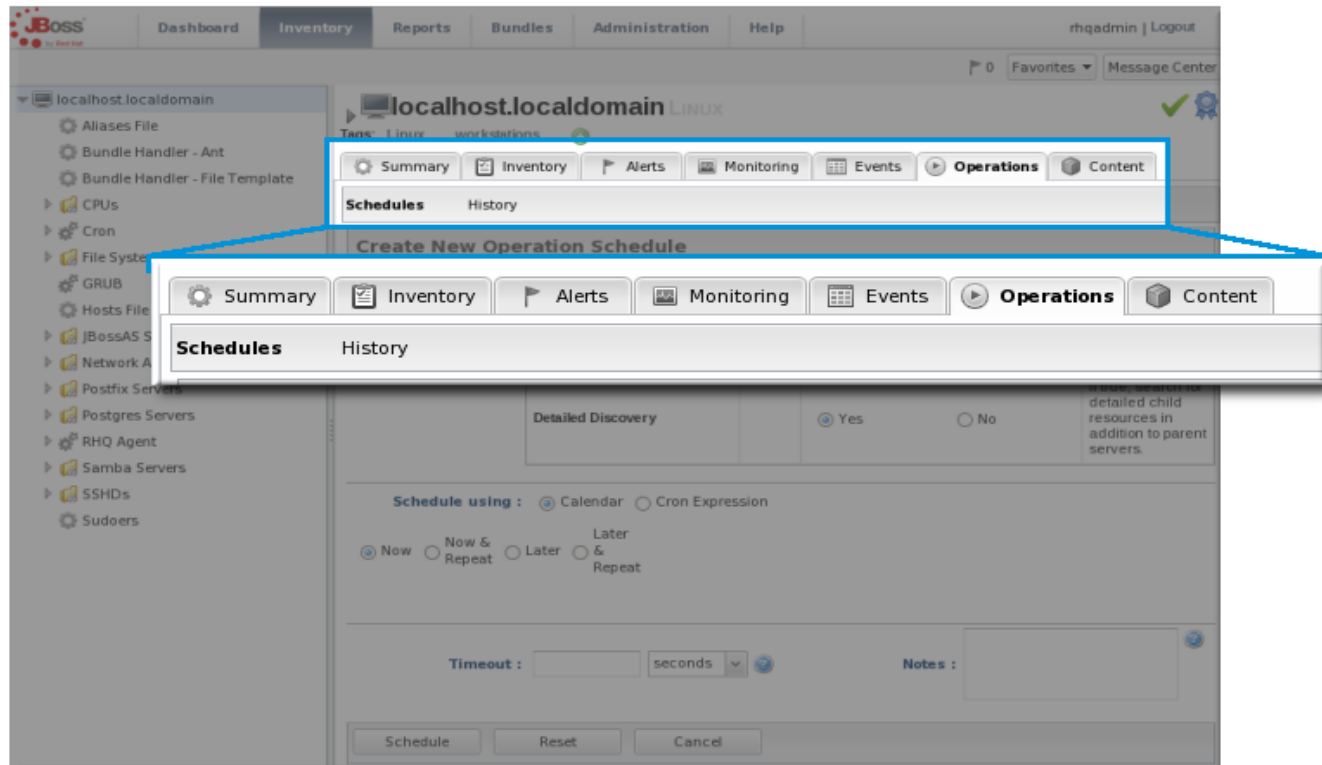


Figure 9. Tabs for a Resource Entry

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.

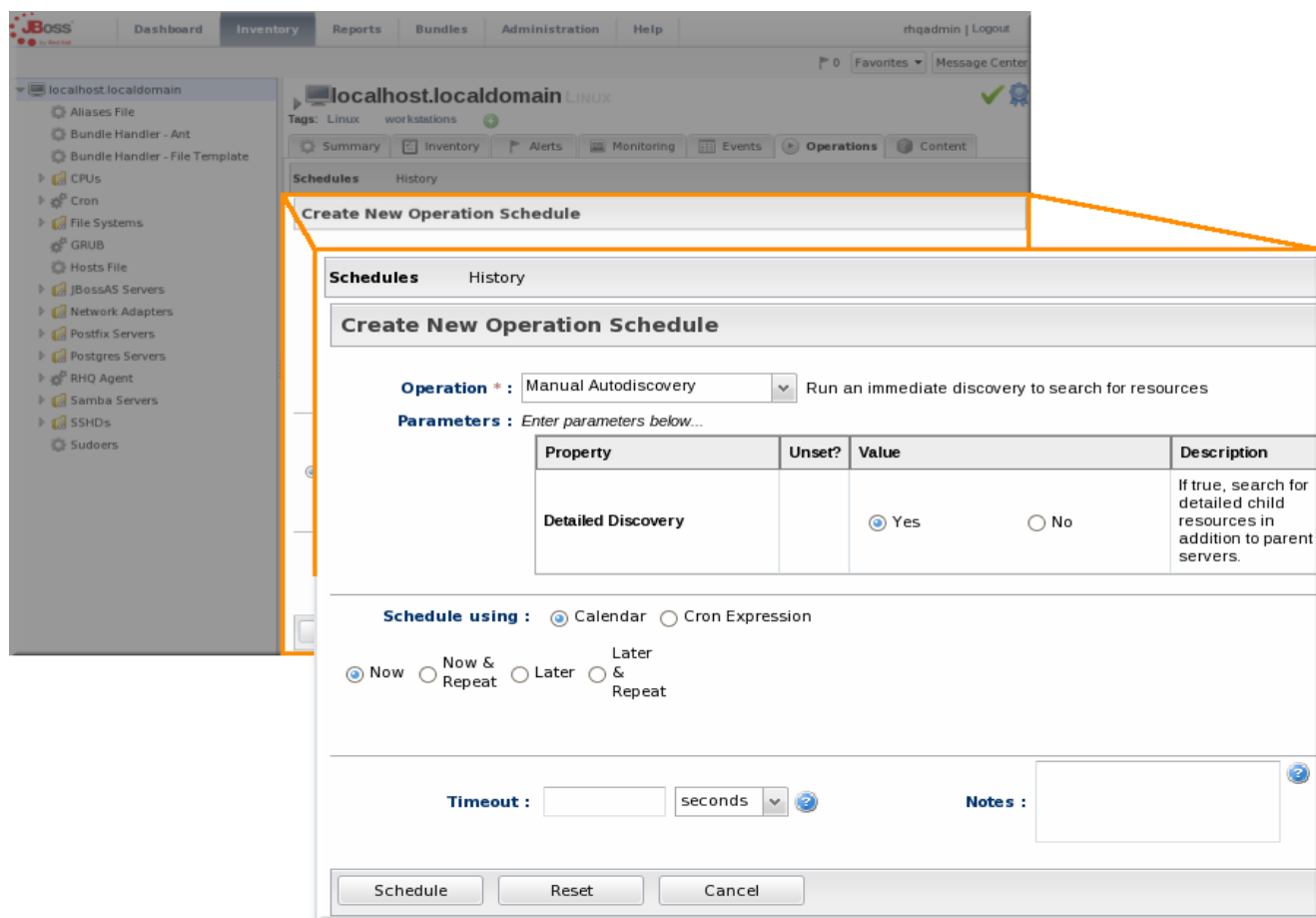


Figure 10. Editable Areas for a Resource Entry

1.2.6. 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.

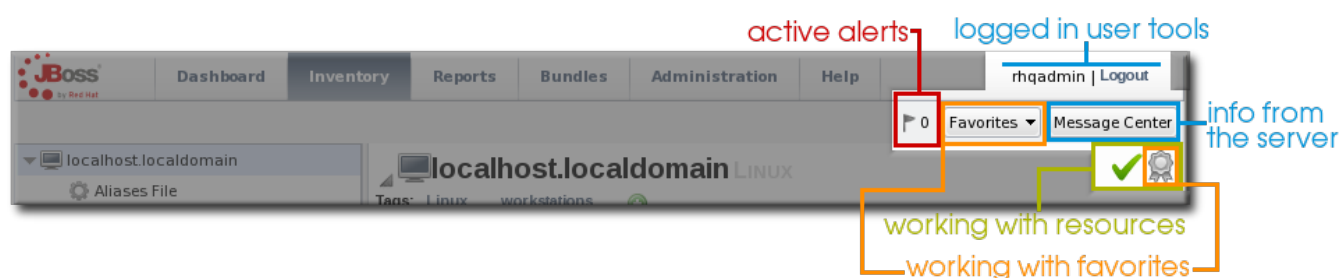


Figure 11. Shortcuts

1.3. 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.

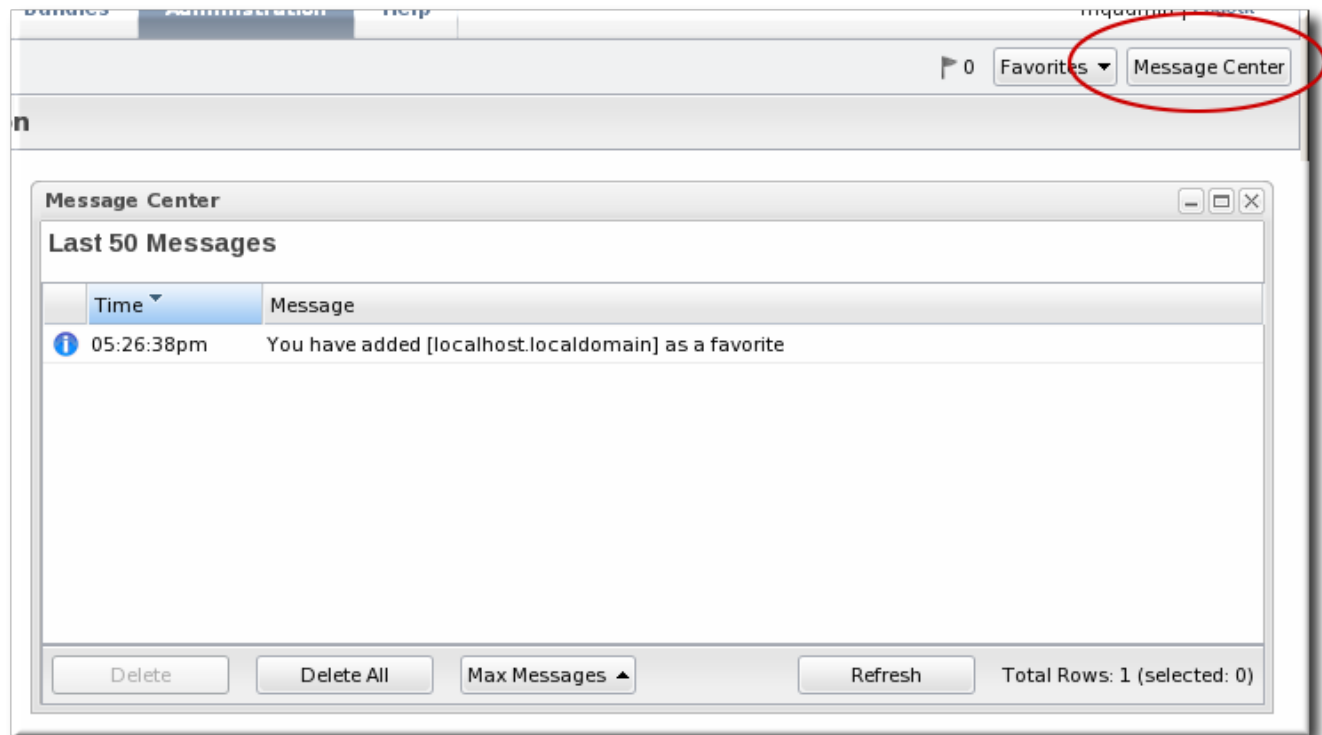


Figure 12. Message Center

1.4. 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.

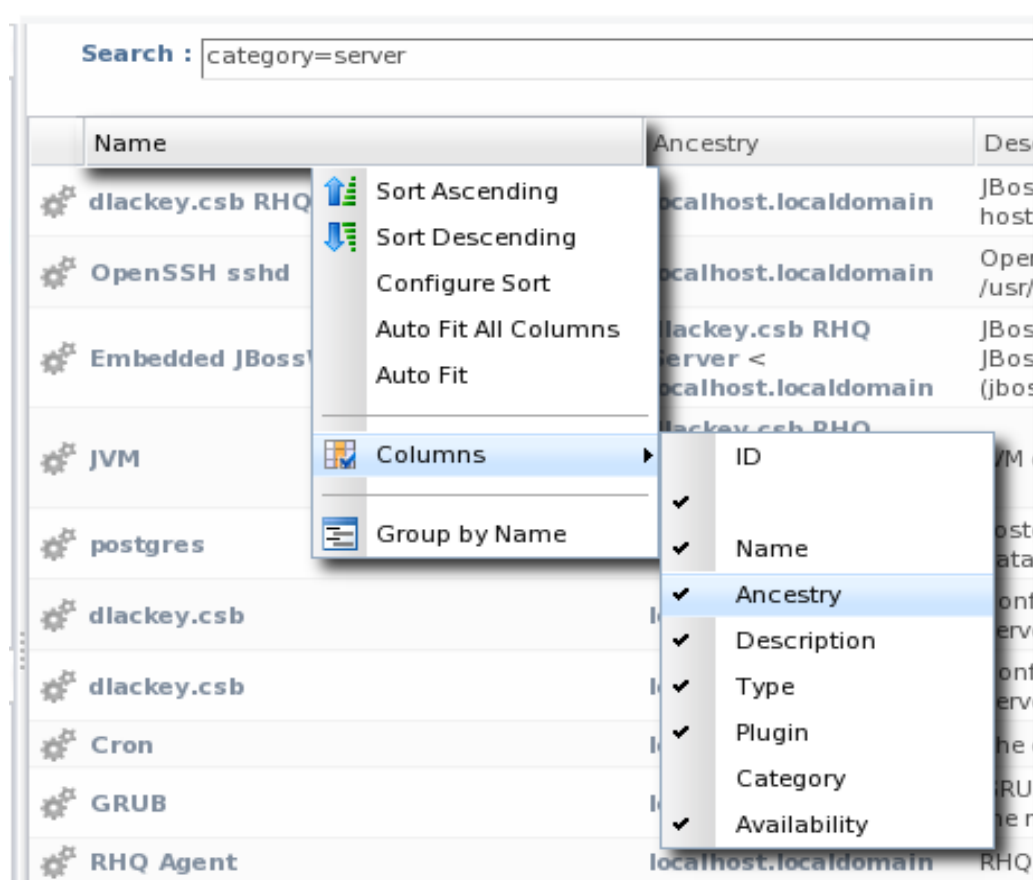


Figure 15. Advanced Table Sorting on the Server Resources List

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.

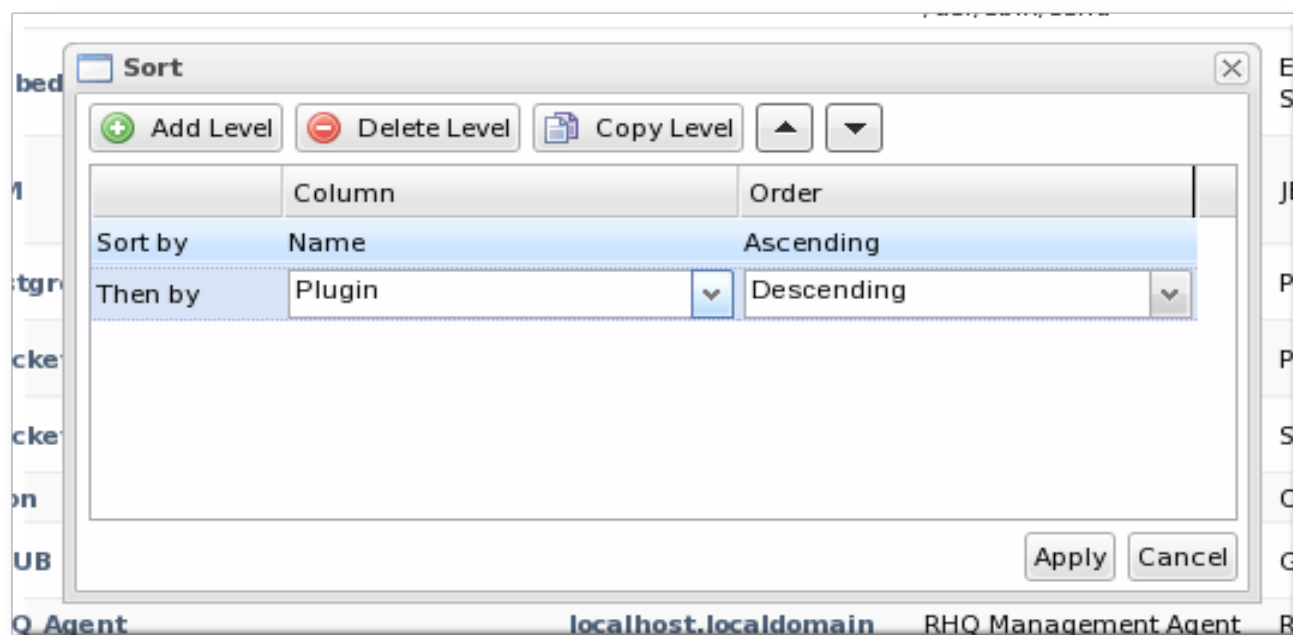


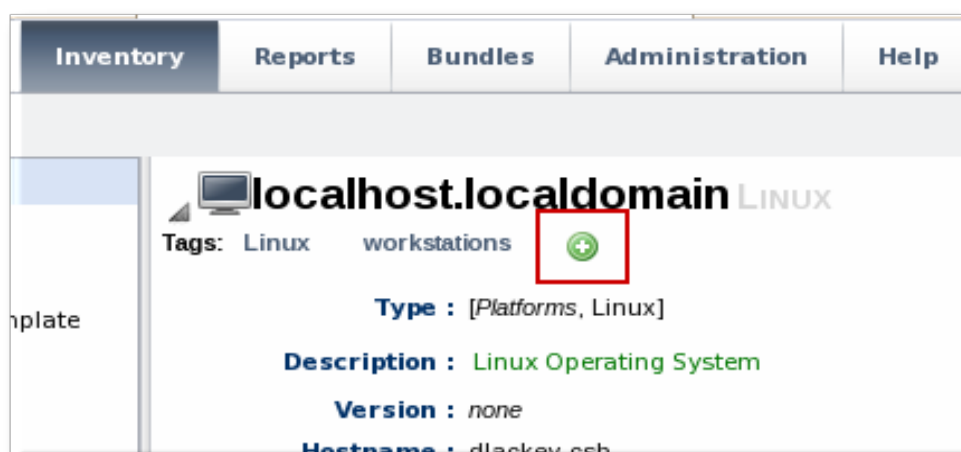
Figure 16. Changing the Sort Method

1.5. Tagging Entries

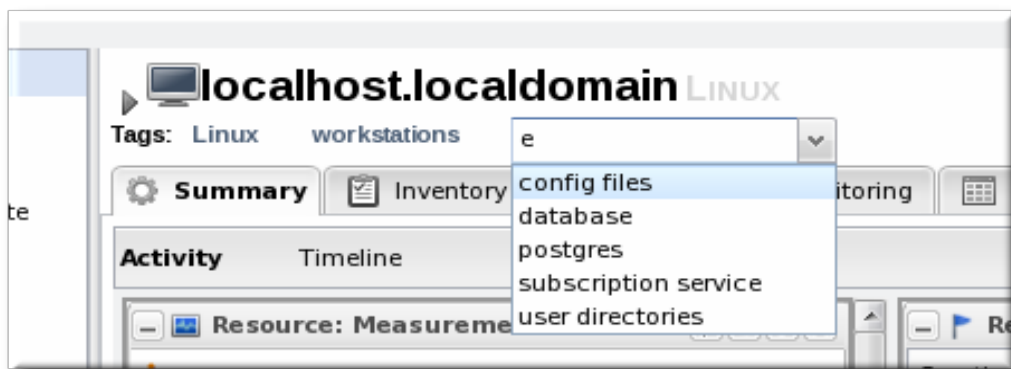
Many entries in JBoss ON can be *tagged*, much like tagging a blog post. Tagging entries has the same purpose as tagging blog posts — it provides an ad hoc and flexible way to group entries by any user-defined terms, including version numbers, keywords, and application names.

1.5.1. Adding Tags

1. Open the entry to tag.
2. Click the cross icon near the **Tags** label.



3. Select a tag from the drop-down menu or type in the text for a new label. The tag only requires a name, but a good format for tags is *namespace:semantic=name*, such as **Dev:env=Main Dev Branch**. This provides more context and makes it easier to search through tags.



4. Hit the **Enter** key to apply the new tag.

1.5.2. Viewing Tags

The tags for a single entry are listed at the top of the entry's display page, in the **Tags** area.

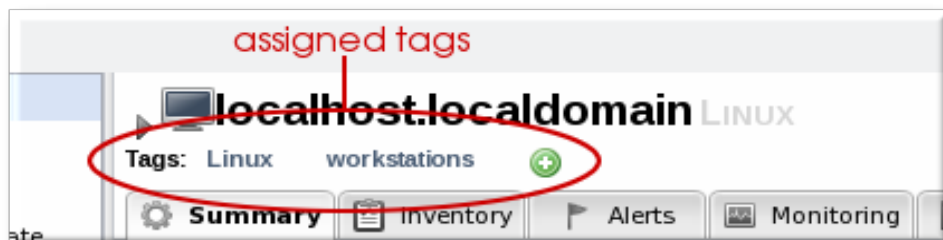


Figure 17. Tags on a Bundle Entry

Clicking on a tag opens a tag area that shows all of the entries which use that particular tag, organized by entry type such as resources and content bundles. There's a cloud at the top of all configured tags in JBoss ON with the different sizes representing the frequency that the tags are used. Clicking on any of the tags at the top opens the tag cloud for that tag.

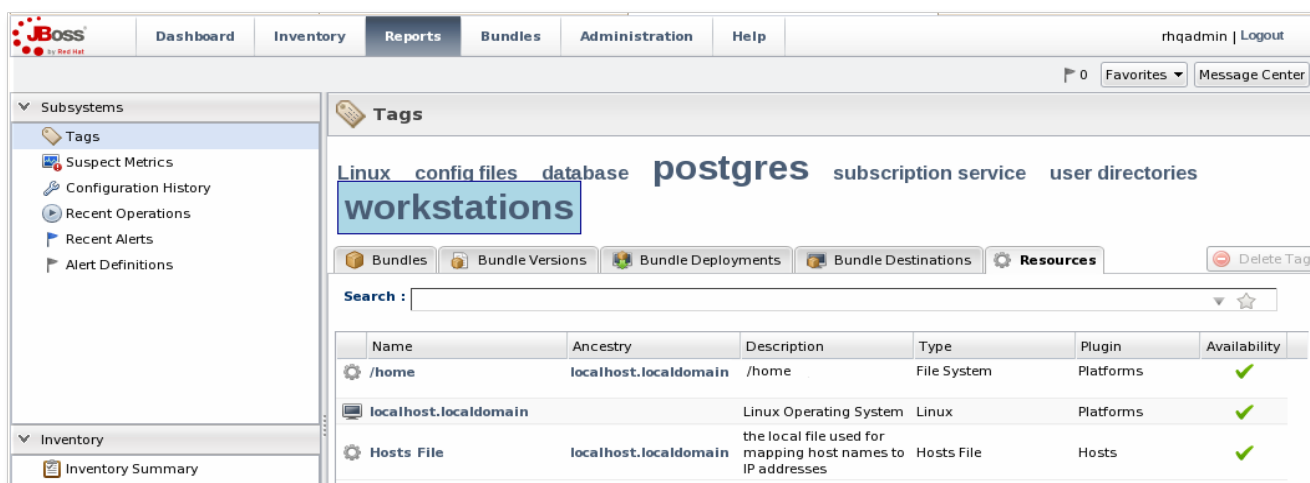


Figure 18. Viewing Every Entry with a Specific Tag

1.5.3. Deleting Tags

1. Open the entry from which to remove the tag.

2. Hover over the name of the tag to remove under the **Tags** label.



3. Click the delete icon that appears by the name of the tag. This immediately removes the tag from the entry.

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.

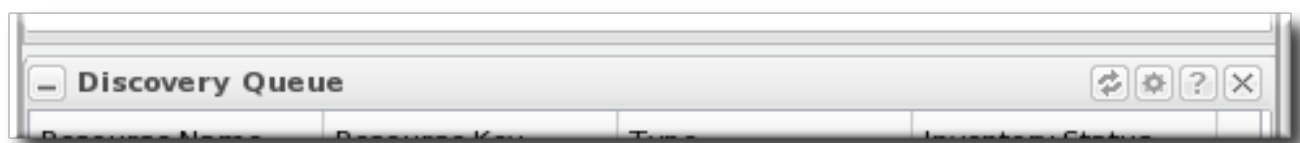


Figure 19. Portlet Icons

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.

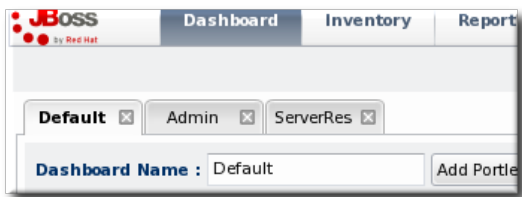
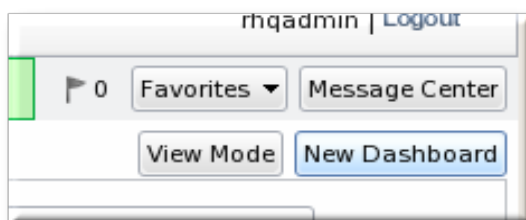


Figure 20. Tabbed Dashboards

To add a new Dashboard:

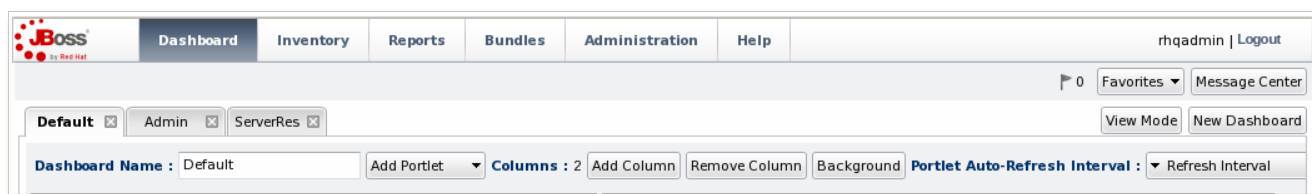
1. Click the **New Dashboard** button in the far right of the main Dashboard.



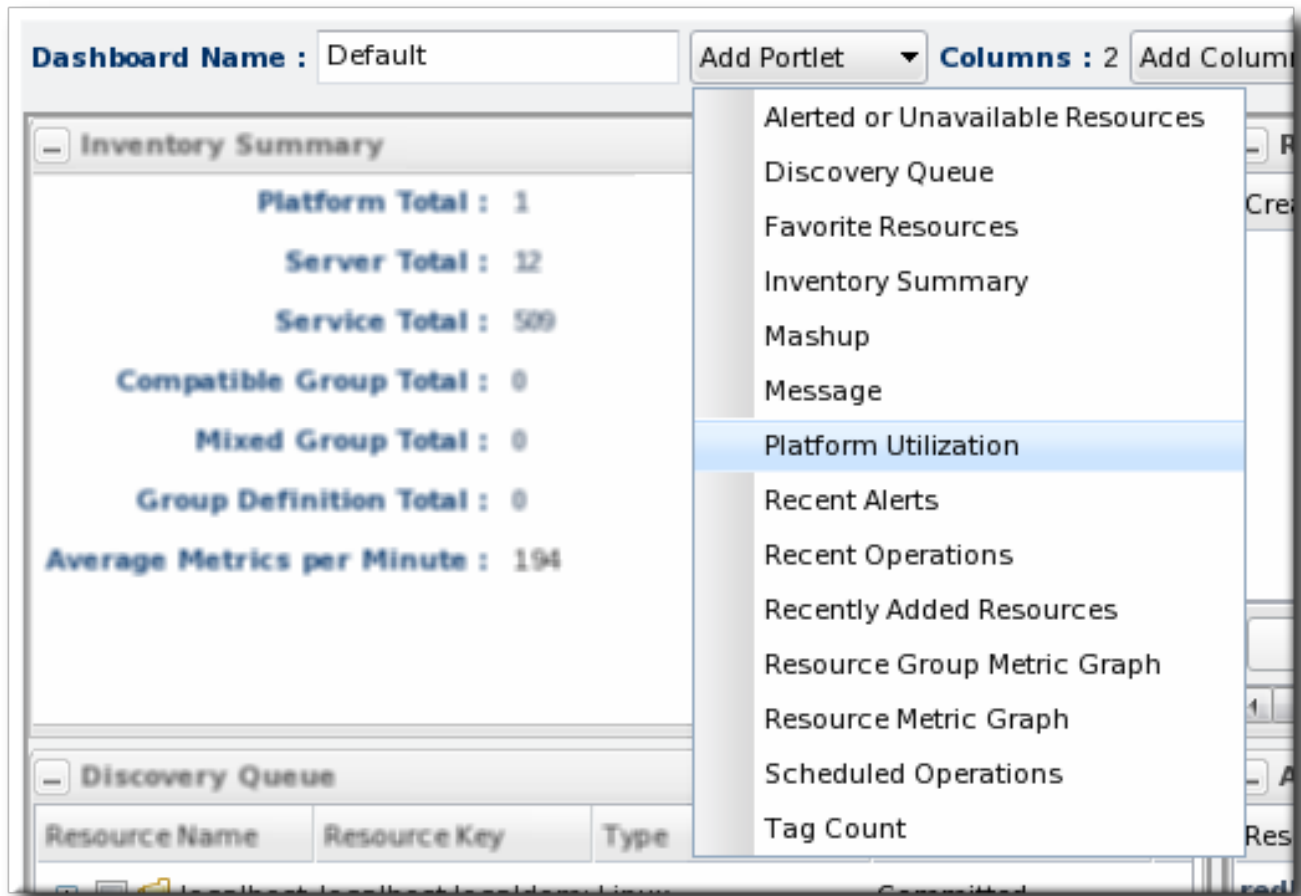
TIP

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.

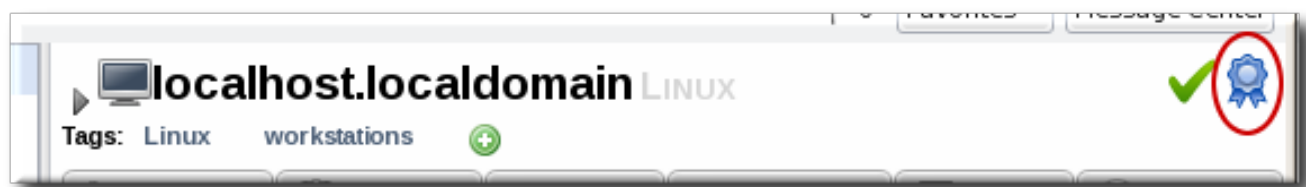


Figure 21. Favorites Icon

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.

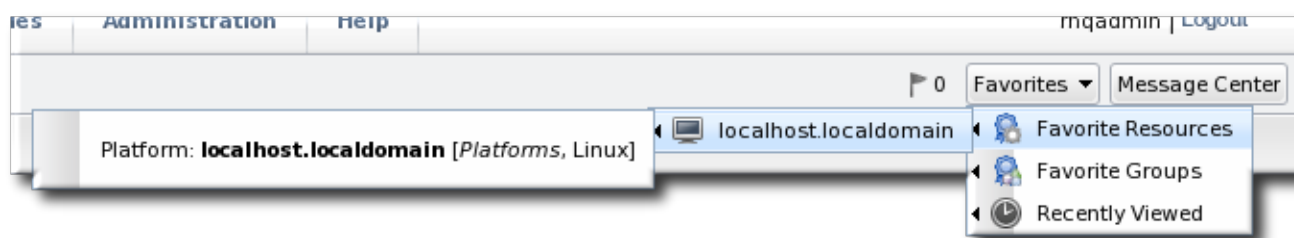


Figure 22. Favorites List

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.

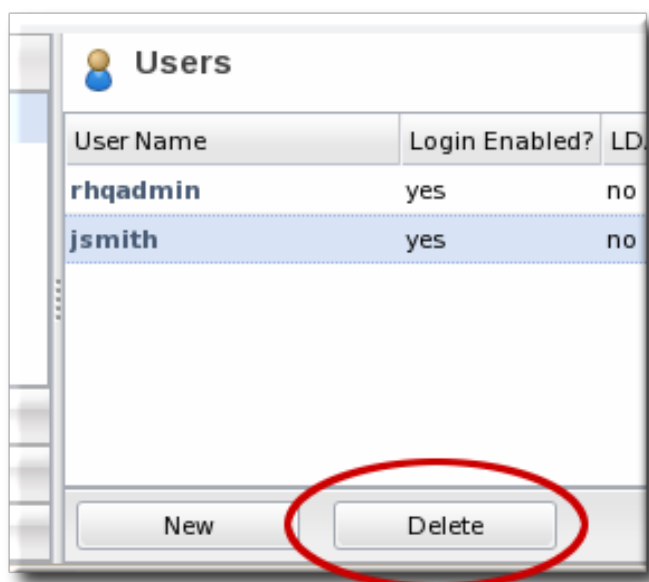


Figure 23. Delete Button in the Area Browser



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

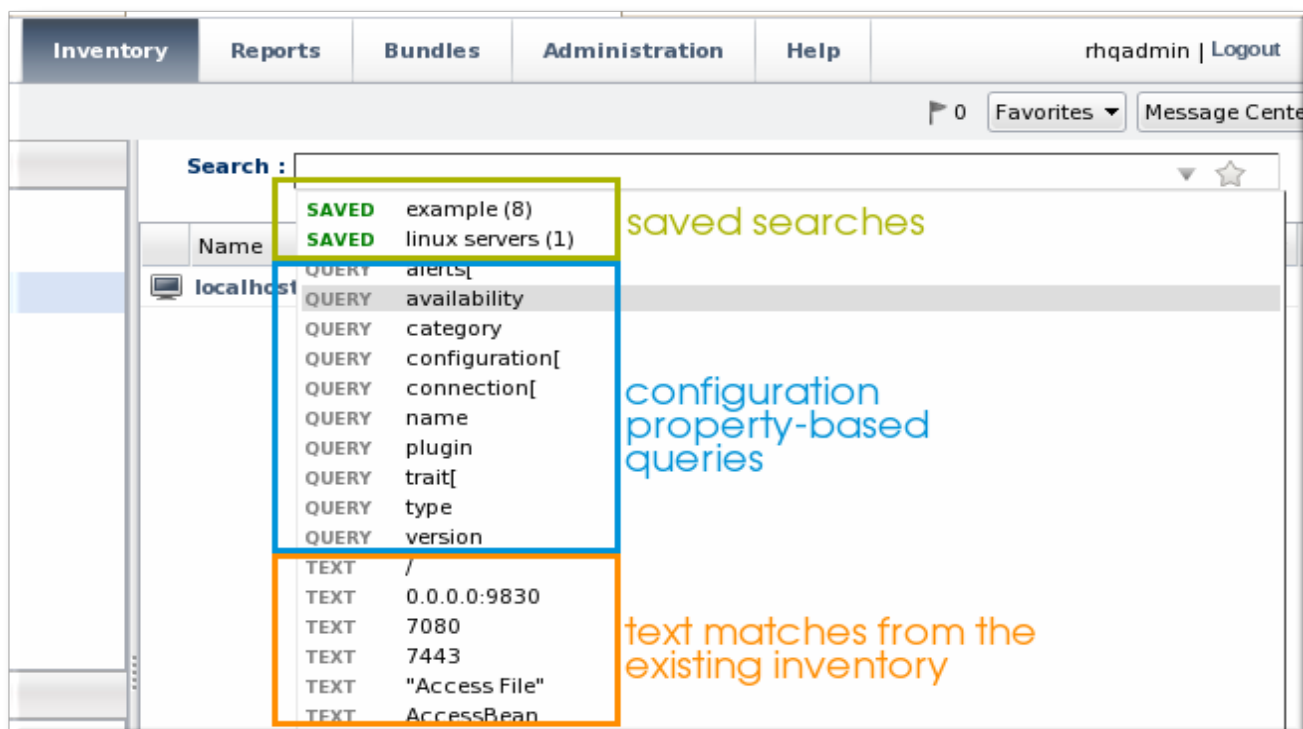


Figure 24. Types of Search Suggestions

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”](#)).

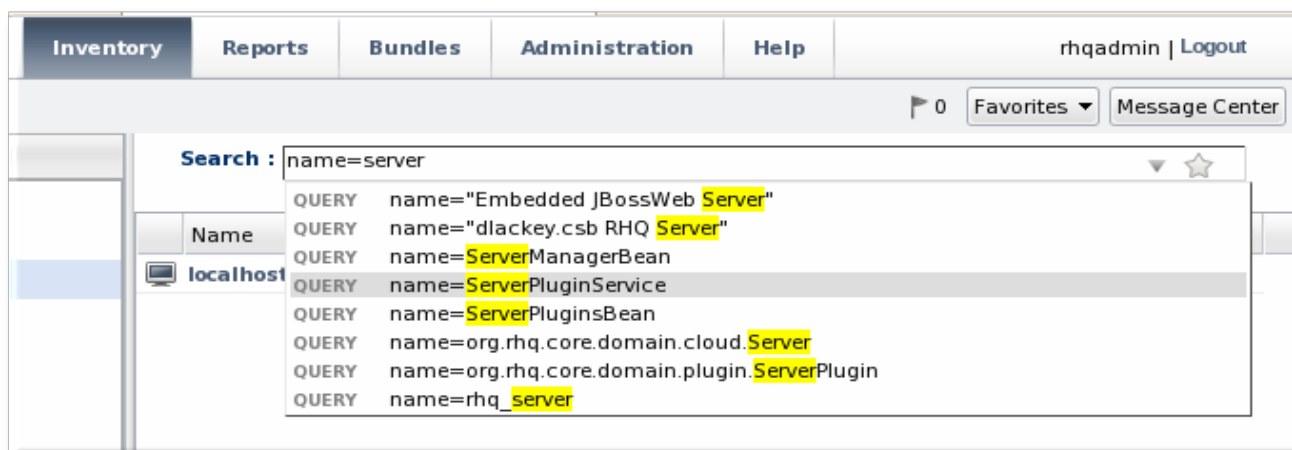


Figure 25. Highlighting Search Terms

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.

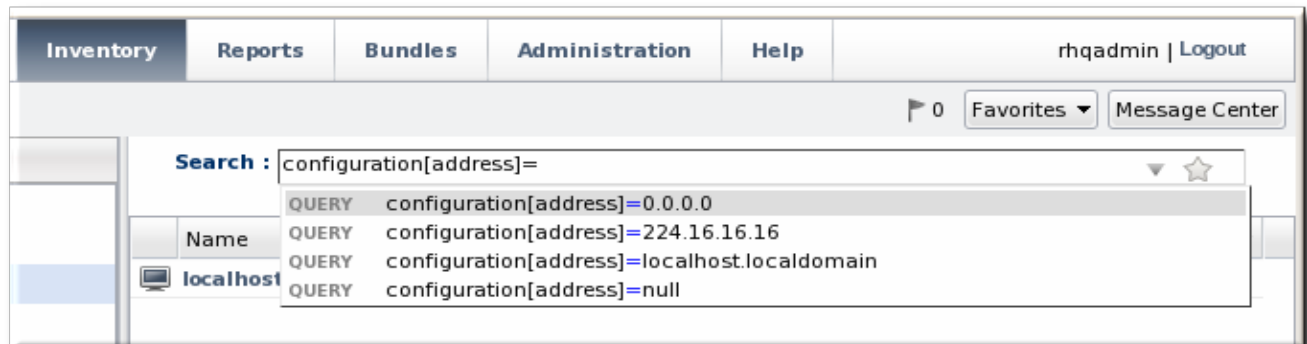


Figure 26. Searching by Resources Traits

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. For example, the

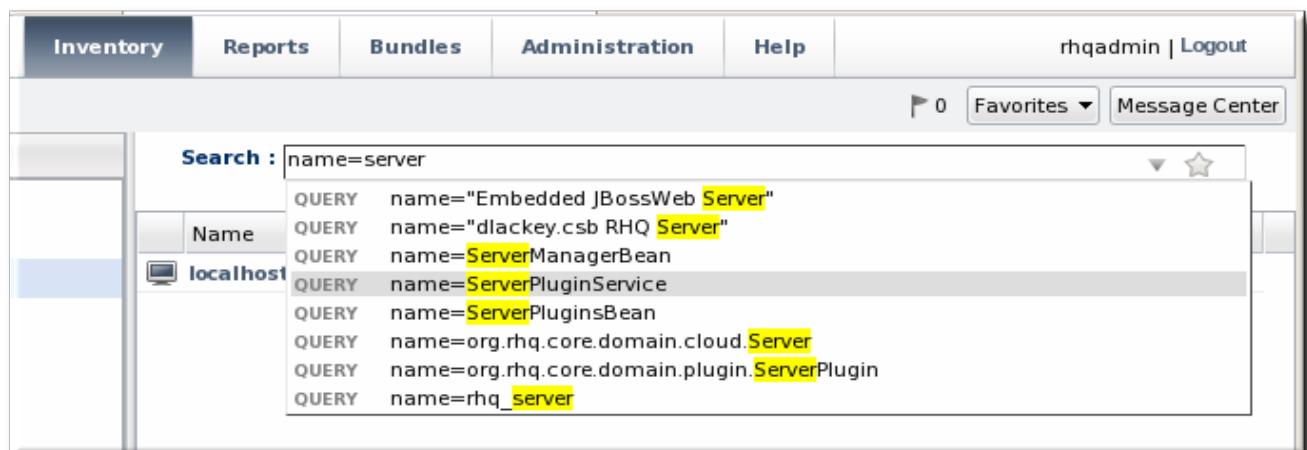


Figure 27. Matching the Search Term

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”](#).



TIP

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.
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*:

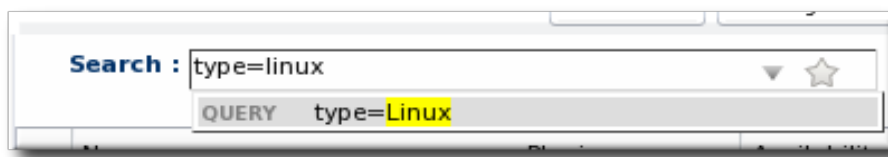
```
(a) (b | (c d))
```

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

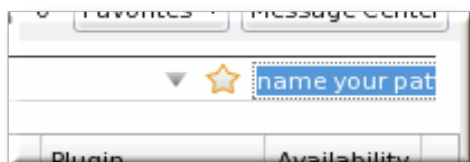
2.3. Saving, Reusing, and Deleting Dynamic Searches

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

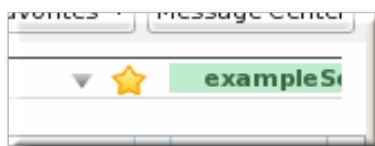
1. Run the search.



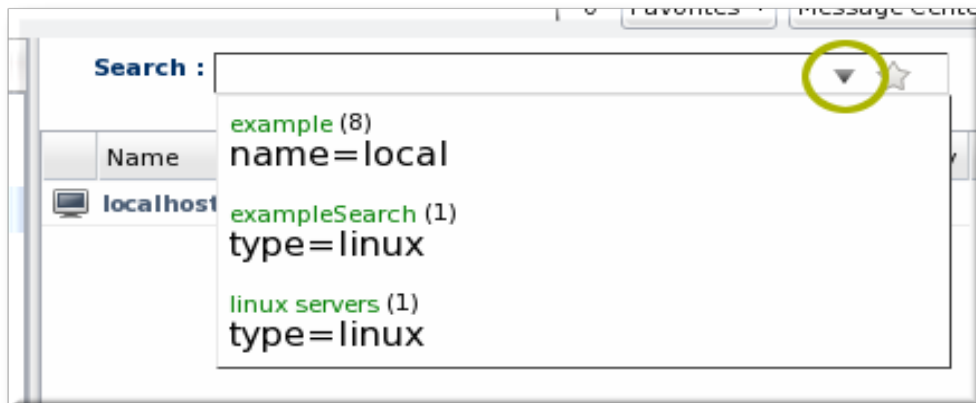
2. Click the star in the right of the search bar. When the field comes up, enter the name for the new search.



The search name is then displayed in green.

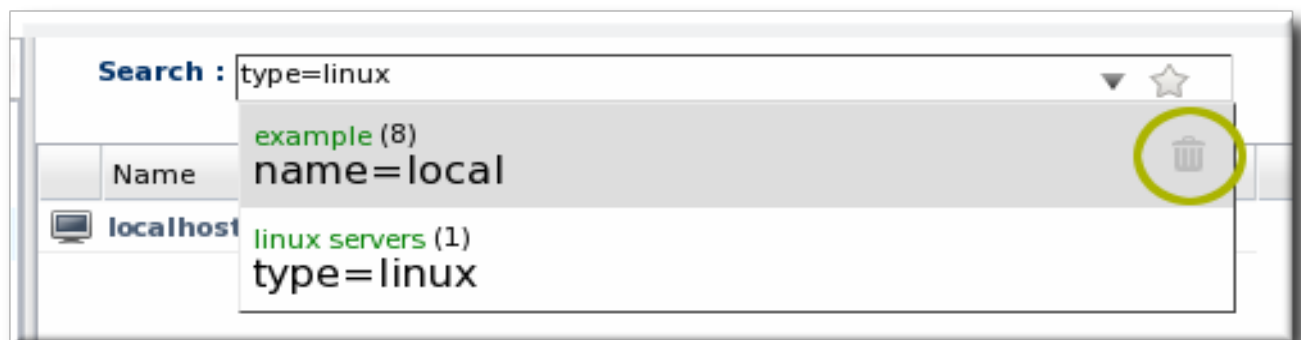


Saved searches are listed with other search results whenever the dynamic search criteria match any part of the saved search string and immediately whenever the dynamic search field is active, before search parameters are entered. The name of the search is shown in bright green. The drop-down option shows 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. 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.

3.1. About the Inventory: Resources

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

3.1.1. 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.

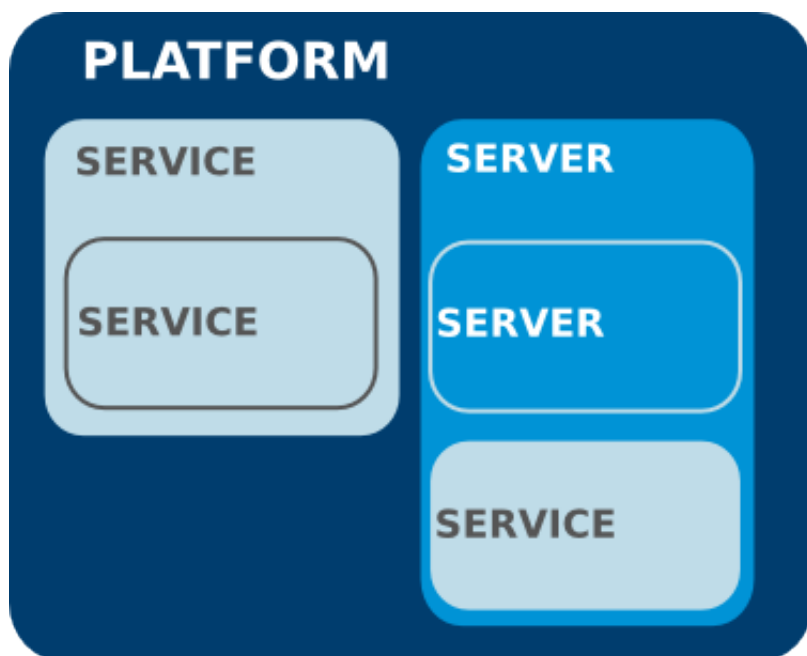


Figure 28. An Example Resource Hierarchy

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 5, "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 5. Default Resources Supported in JBoss ON

Managed Platform			
AIX Platform	FreeBSD	HP-UX Platform	Java Platform
Linux Platform	Mac OS X Platform	Solaris Platform	Windows Platform
Managed Services			
CPU Service	File System Service	Network Adapter Service	
Managed Servers			
Apache HTTP Server	APT Source Server	GRUB Server	Hibernate Statistics Service
Host Server	HTTPService	IIS Server	JBoss AS Server
JBossCacheSubsystem Server	JBossOSGi Server	JMX Server	MySQL Server
Oracle Server	Ping Service	Postgres Server	JBoss ON Agent
JBoss ON Server	Tomcat Server		

3.1.2. 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.

3.2. 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.

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.

3.2.1. 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.

3.2.2. 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. Start the agent terminal. Since the agent should already be running, using the **--nostart** option to start the agent terminal without attempting to start the agent process.

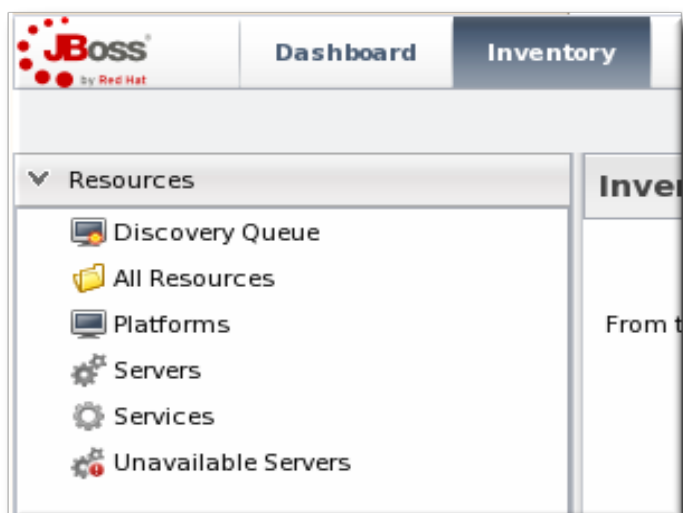
```
# agentRoot/rhq-agent/bin/rhq-agent.sh --nostart
```

2. In the agent terminal, run the **discovery** command. Using the **-f** option runs a full discovery, for servers and services. Running **discovery** alone checks for servers only.

```
> discovery -f
```

Alternatively, a discovery scan can be initiated by running an operation on the platform. (Operations are discussed in detail in *Setting up Monitoring, Alerts, and Operations*.)

1. Click the **Inventory** tab in the top menu.
2. Select the platform resource in the **Resources** menu table on the left.



3. Click the **Operations** tab.
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).

 A screenshot of the 'Create New Operation Schedule' form in the JBoss Inventory interface. The form is titled 'Linux Server 1 LINUX' and has tabs for 'Summary', 'Inventory', 'Alerts', 'Monitoring', 'Operations', 'Events', and 'Content'. The 'Operations' tab is selected, and the 'Schedules' subtab is active. The form contains a dropdown menu for 'Operation' set to 'Manual Autodiscovery' with a description 'Run an immediate discovery to search for resources'. Below this is a 'Parameters' section with a table for configuration.

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 :

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

3.2.3. Importing Resources from the Discovery Queue

- Click the **Dashboard** tab in the top menu.
- In the **Discovery Queue** portlet, select the checkbox by the name of the resources to import. Selecting a parent resource (such as a platform) gives the option to automatically import all of its children, too.

JBoss by Red Hat

Dashboard **Inventory** **Reports** **Bundles** **Administration**

Default **Admin** **ServerRes**

Discovery Queue

Resource Name	Resource Key	Type	Inventory Status
<input checked="" type="checkbox"/> Linux Server 1	localhost.localdomain	Linux	Committed
<input checked="" type="checkbox"/> 0.0.0.0: /etc/httpd	/etc/httpd	Apache HTTP Server	New
<input type="checkbox"/> Cobble	/etc/cobbler	Cobbler	New

Show :

**NOTE**

The **Discovery Queue** portlet may not be visible in the default Dashboard configuration. Add this portlet or create another Dashboard view to access this portlet.

Changing the Dashboard configuration is covered in [Section 1.6, “Customizing the Dashboard”](#).

3. Click the **Import** button.

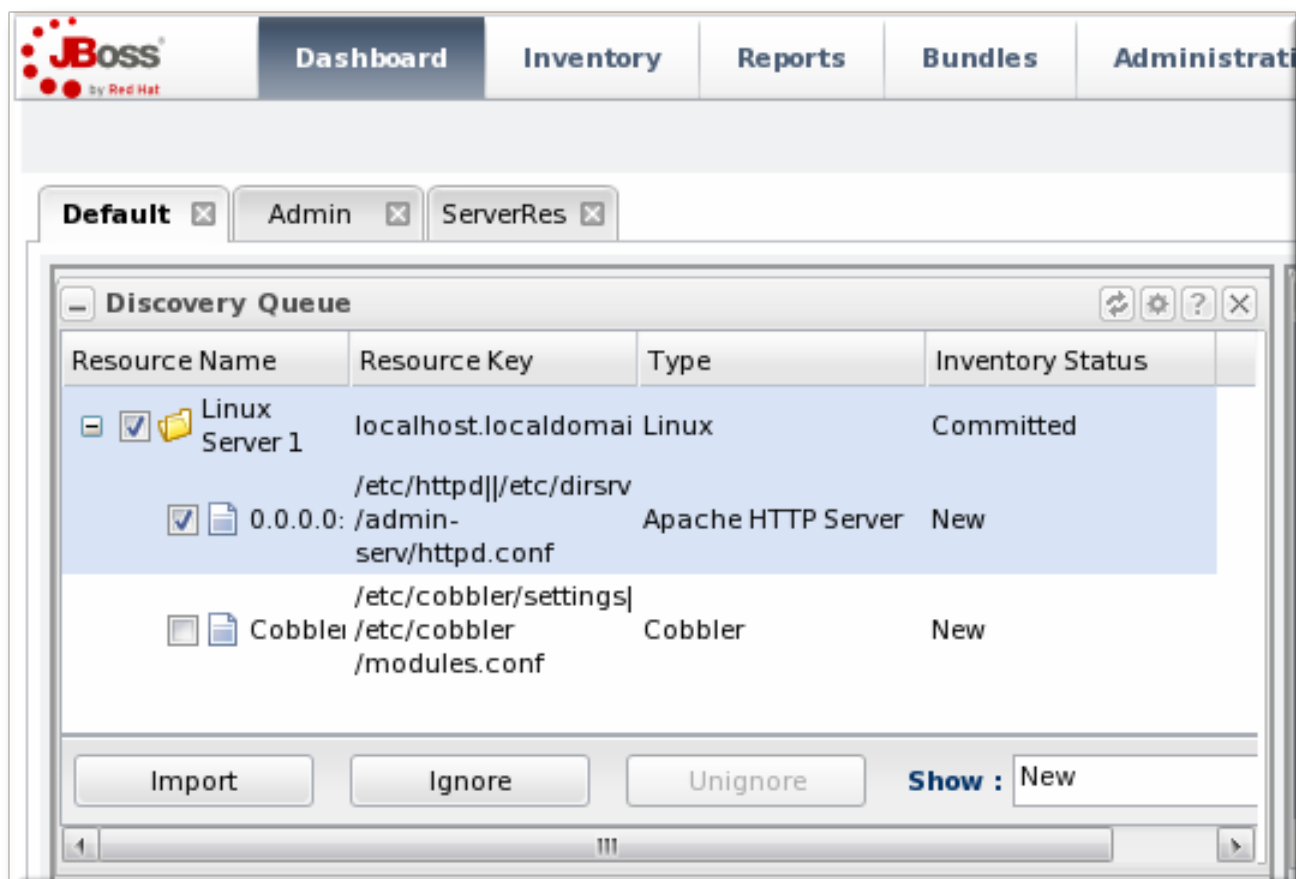
3.2.4. Ignoring Discovered Resources

When the agent discovers an application or service that will not be included in 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. Click the **Dashboard** tab in the top menu.
2. In the **Discovery Queue** portlet, select the checkbox by the name of the resources to ignore. Selecting a parent resource automatically ignores all of its children, too.



- Click the **Ignore** button.



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.

3.3. 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

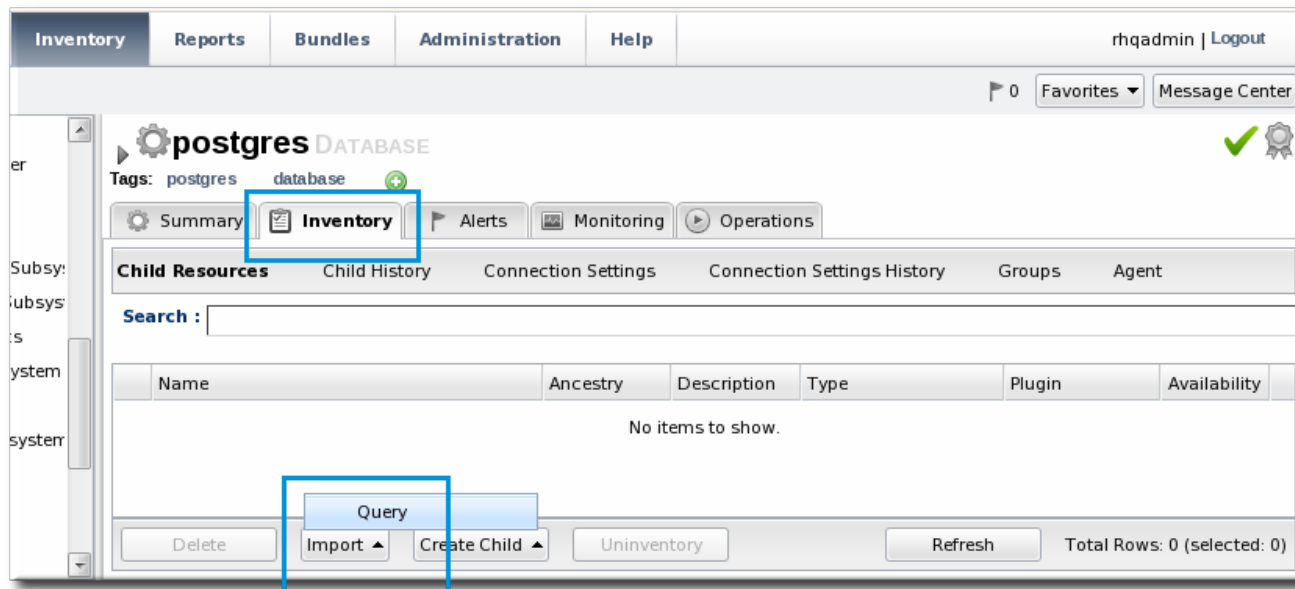
The parent resource must be in an available state in order to import a child resource.

- 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.

Resource Import Wizard

Import Resource of Type [Query] Step 1 of 1

Edit Configuration

Property	Unset?	Value	Description
Table		example	The table to discover
Name		MyExamplePostgresDB	
Description		An example postgres db table	
Metric Query			The query that will gather metric data

Cancel Previous Finish

3.4. 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\TomcatVer#\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.

3.5. 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.

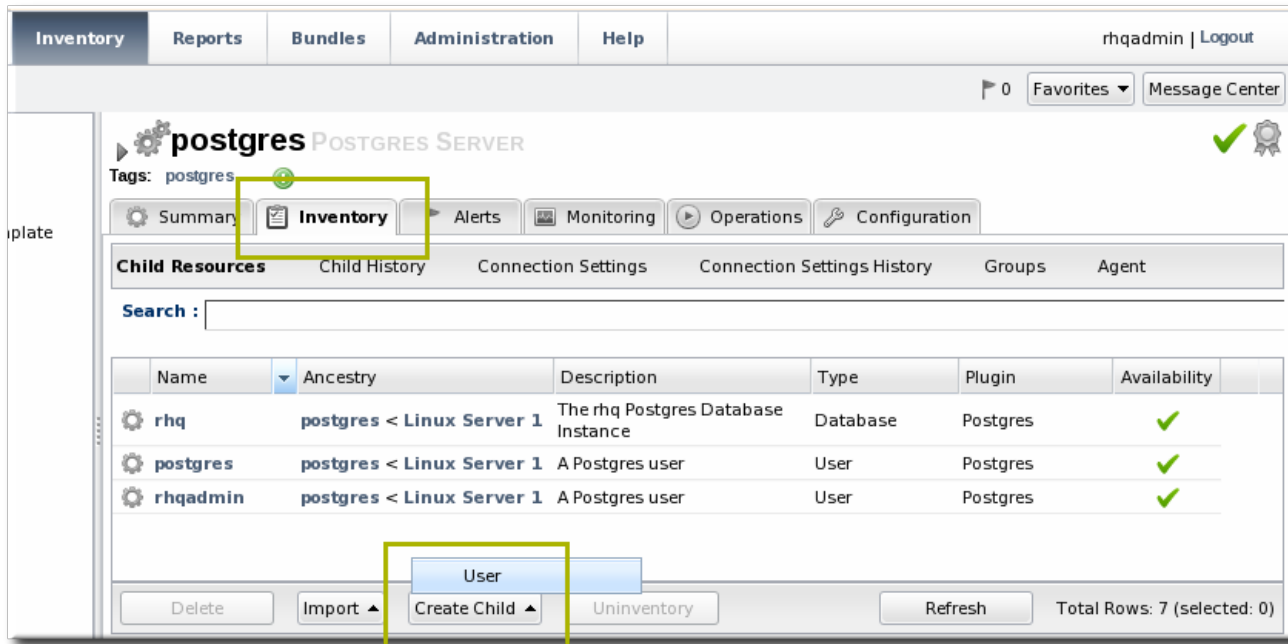


NOTE

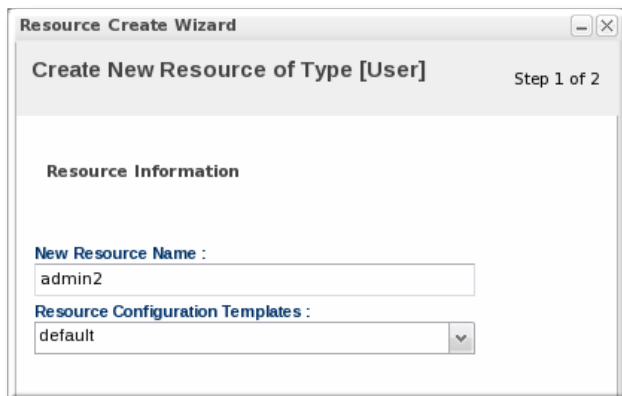
The parent resource must be available to add a child resource.

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.



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



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

Import Resource of Type [Query] Step 1 of 1

Edit Configuration

Property	Unset?	Value	Description
Table		example	The table to discover
Name		MyExamplePostgresDB	
Description		An example postgres db table	
Metric Query		❗	The query that will gather metric data

Cancel Previous Finish

3.6. 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.

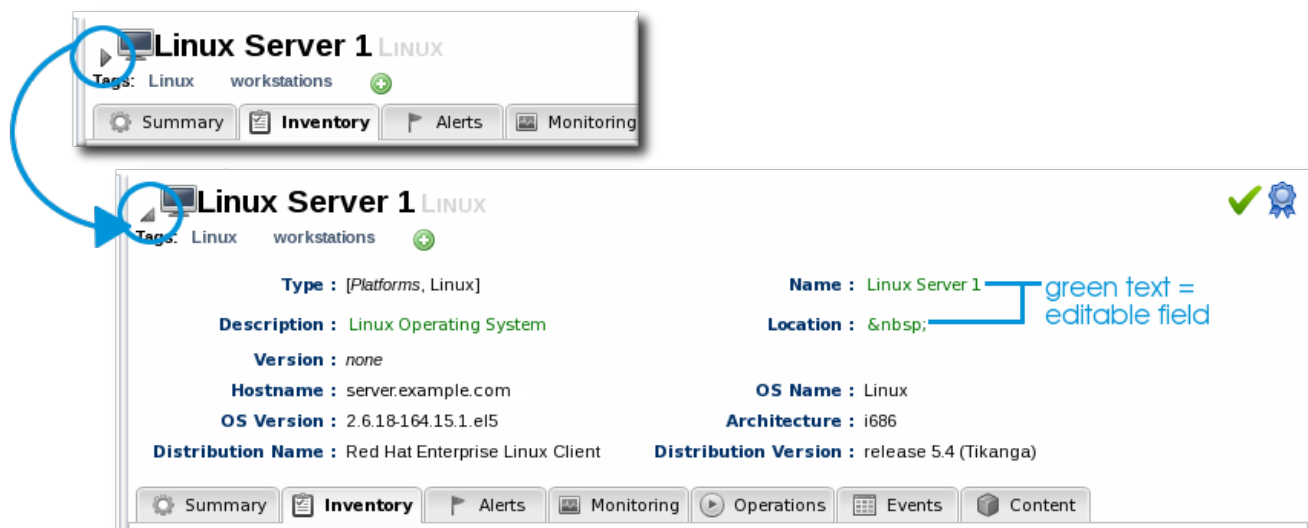
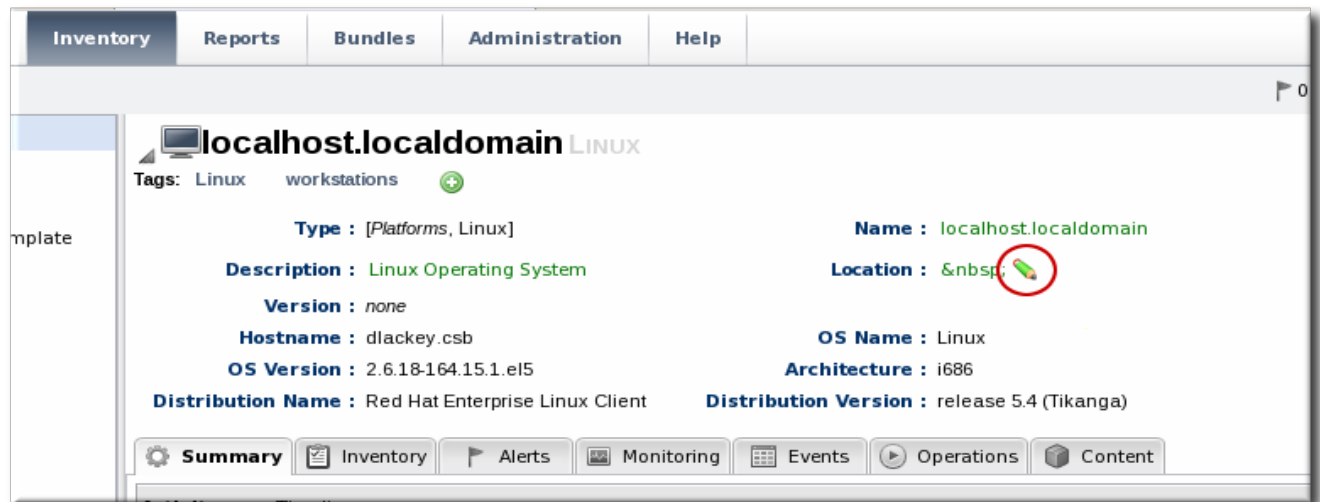


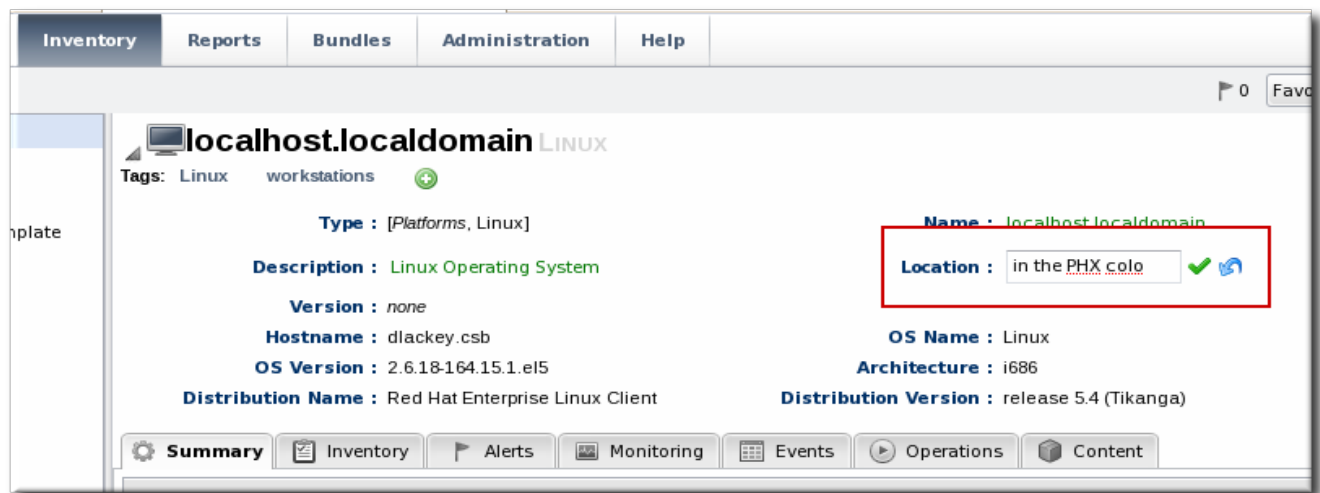
Figure 29. Expanding Resource Entry Details

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.



3.7. Removing Resources from the Inventory

A resource can be removed from the JBoss ON inventory after it has been imported.

NOTE

Removing a resource from the inventory does not block the resource from future discovery scans. If the resource still exists on the platform and should not be included in the JBoss ON inventory, then ignore the resource after it is removed from the inventory. See [Section 3.2.4, “Ignoring Discovered Resources”](#).

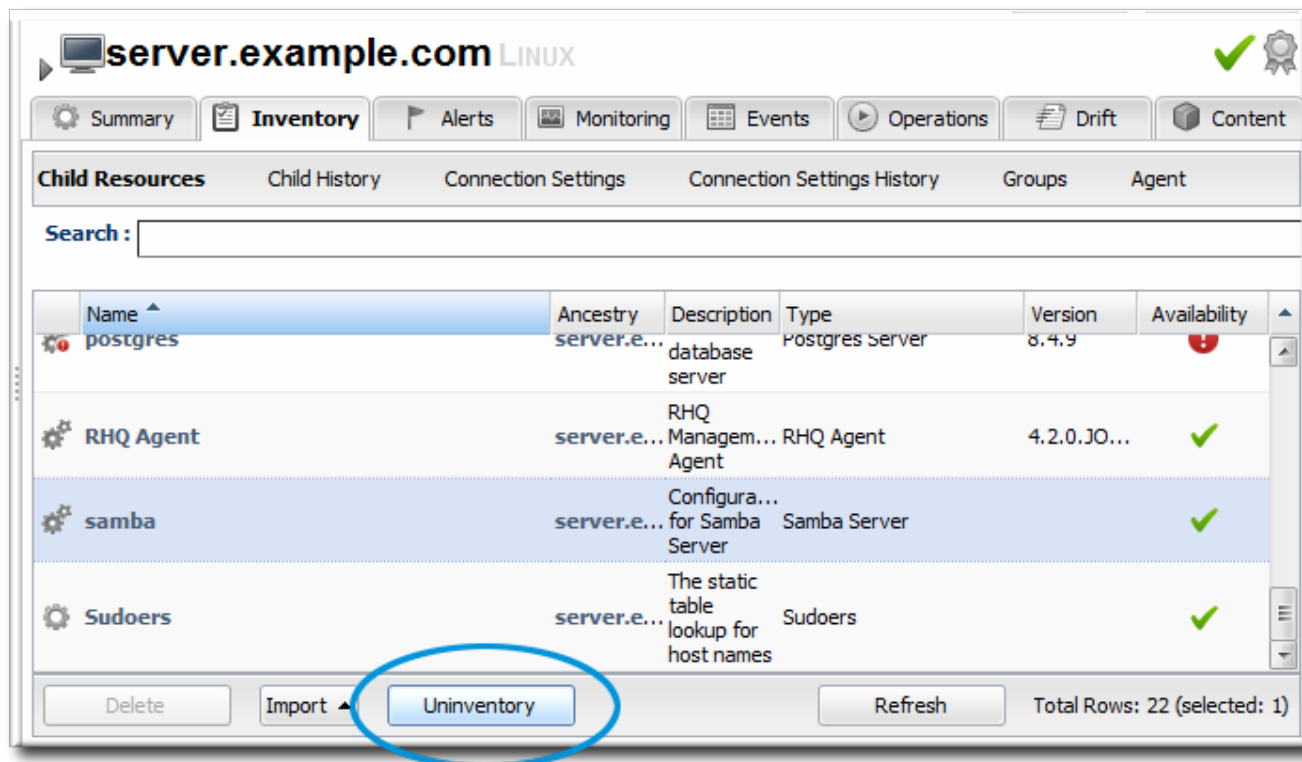
A resource is removed from the inventory by editing the inventory settings in a parent entry. This can be the inventory for the resource's parent entry or a group to which the resource belongs.

3.7.1. Uninventorying through the Parent Inventory

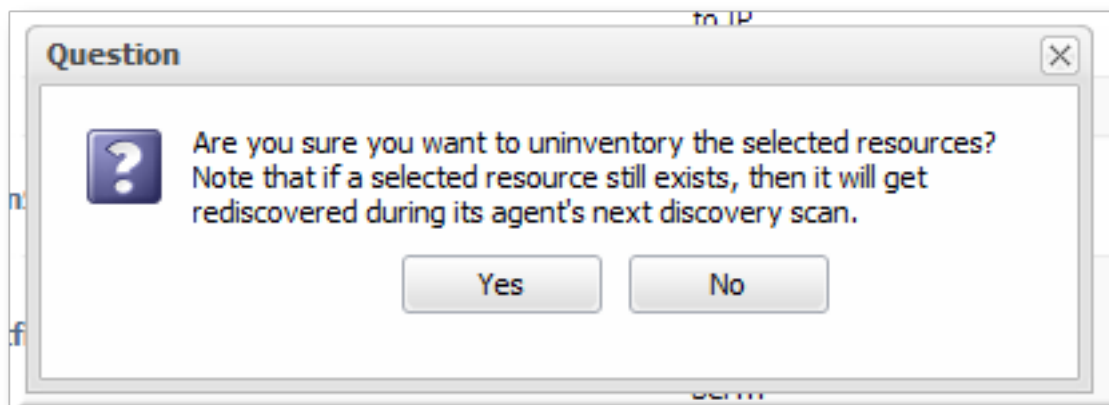
1. Click the **Inventory** tab in the top menu.
2. Search for the parent resource of the new resource.

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

3. Click the **Inventory** tab for the parent resource.
4. Click on the line of the child resource to uninventory. To select multiple entries, use the **Ctrl** key.



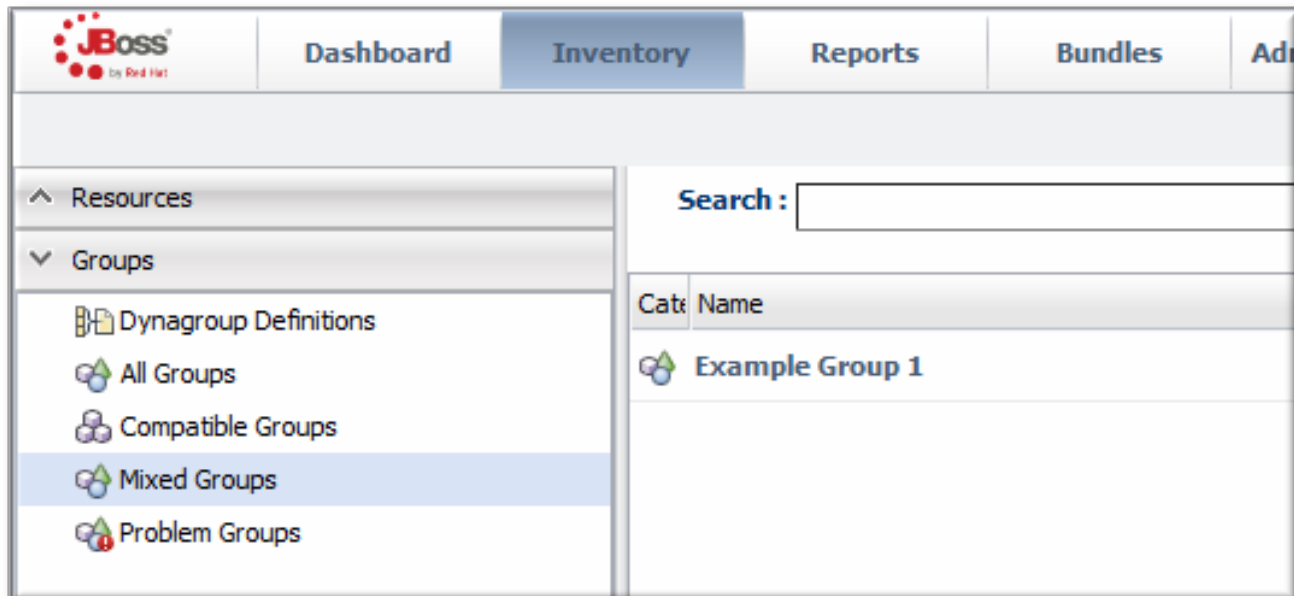
5. Click the **Uninventory** button.
6. When prompted, confirm that the resource should be uninventoried.



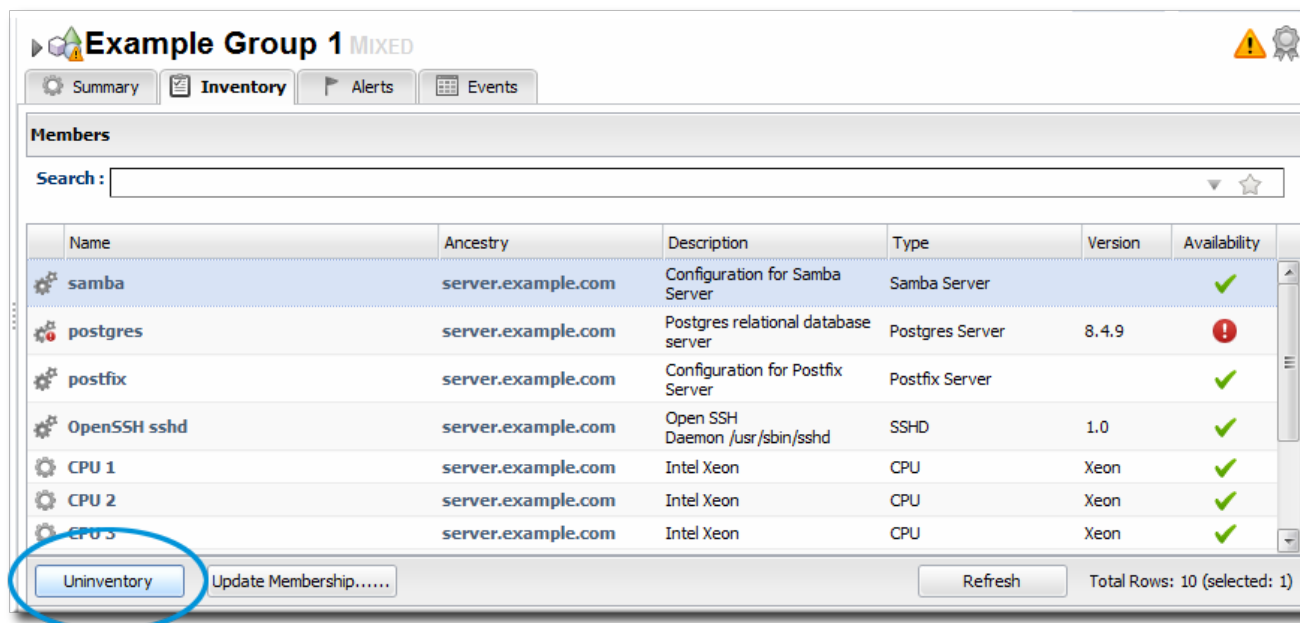
3.7.2. Uninventorying through a Group Inventory

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

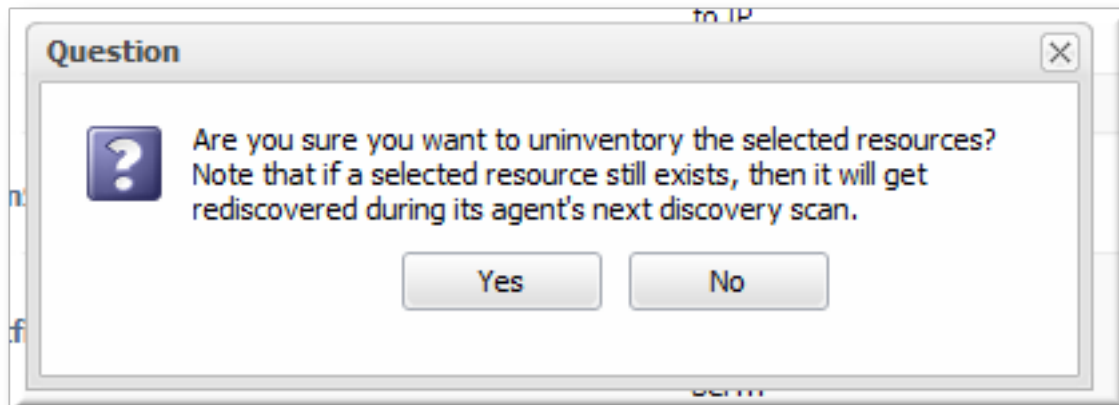
1. In the **Inventory** tab in the top menu, select the compatible or mixed groups item in the **Groups** menu on the left.



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 uninventoried. To select multiple entries, use the **Ctrl** key.



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



3.8. Getting Inventory 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 of the resource type; for example, for the Linux resource type, there can be a Linux 2.6.18-164.15.1.el5 version
- 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.

A screenshot of the JBoss ON web interface. The top navigation bar includes 'Dashboard', 'Inventory', 'Reports' (selected), 'Bundles', 'Administration', and 'Help'. The user 'rhqadmin' is logged in. On the left, a sidebar shows 'Subsystems' and 'Inventory' with 'Inventory Summary' selected. The main content area displays the 'Inventory Summary' report as a table.

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

**TIP**

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

4. 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.

4.1. 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 6, “Types of Groups”](#), which allows an administrator to manage resources in different, flexible ways.

Table 6. 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
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	Contains all the descendant, or child, resources of resources within the group. Recursive groups show both the primary member availability and the child resource availability.	Static (members) and dynamic (children)

Type	Description	Static or Dynamic
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

4.1.1. Dynamic and Static Groups

Groups are a way of organizing resources. The different types of groups are covered in [Table 6, “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.

4.1.2. 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 4.2, “Creating 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 30, “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.

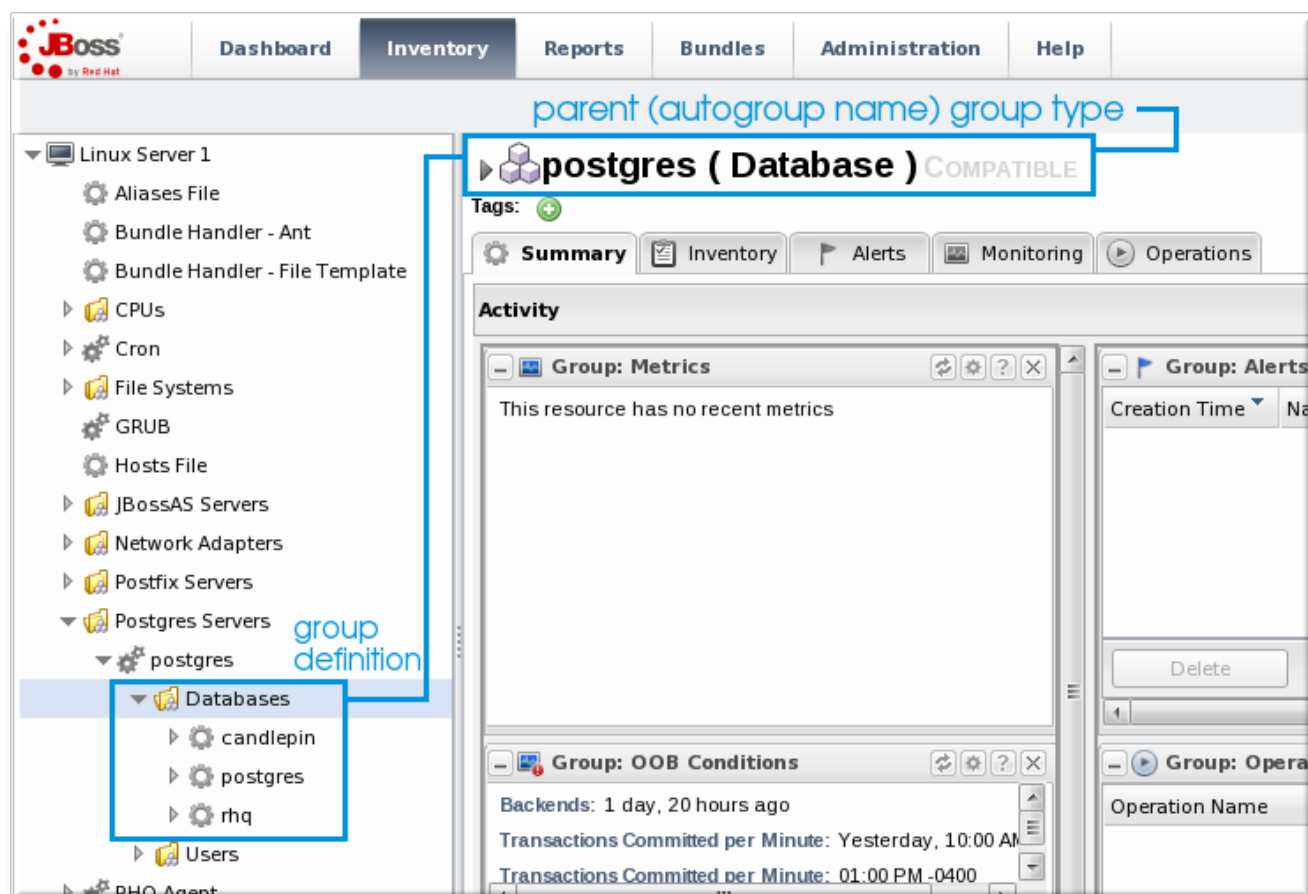


Figure 30. PostgreSQL Autogroup

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.

4.1.3. 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.

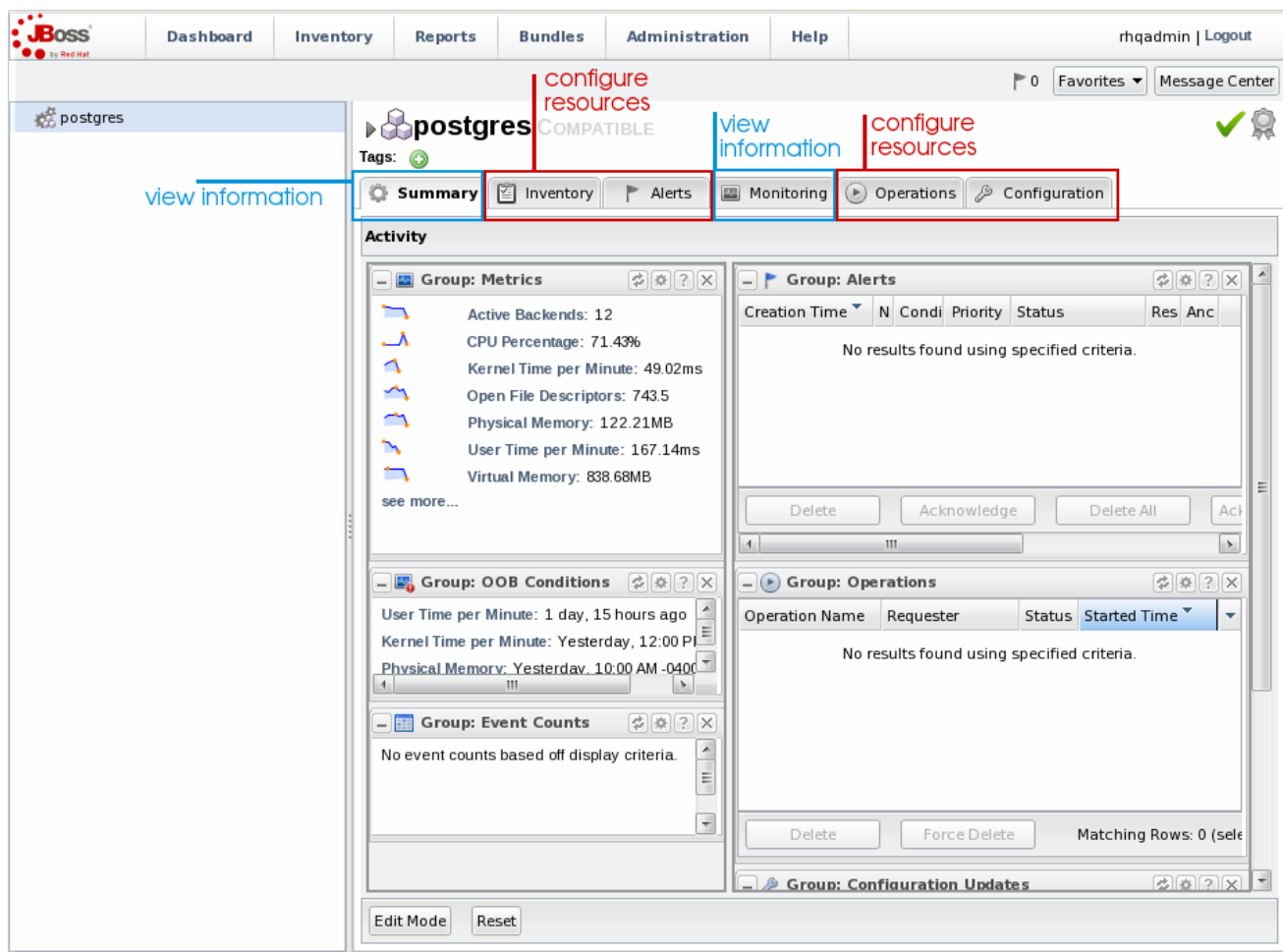


Figure 31. Compatible Group Entry

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.

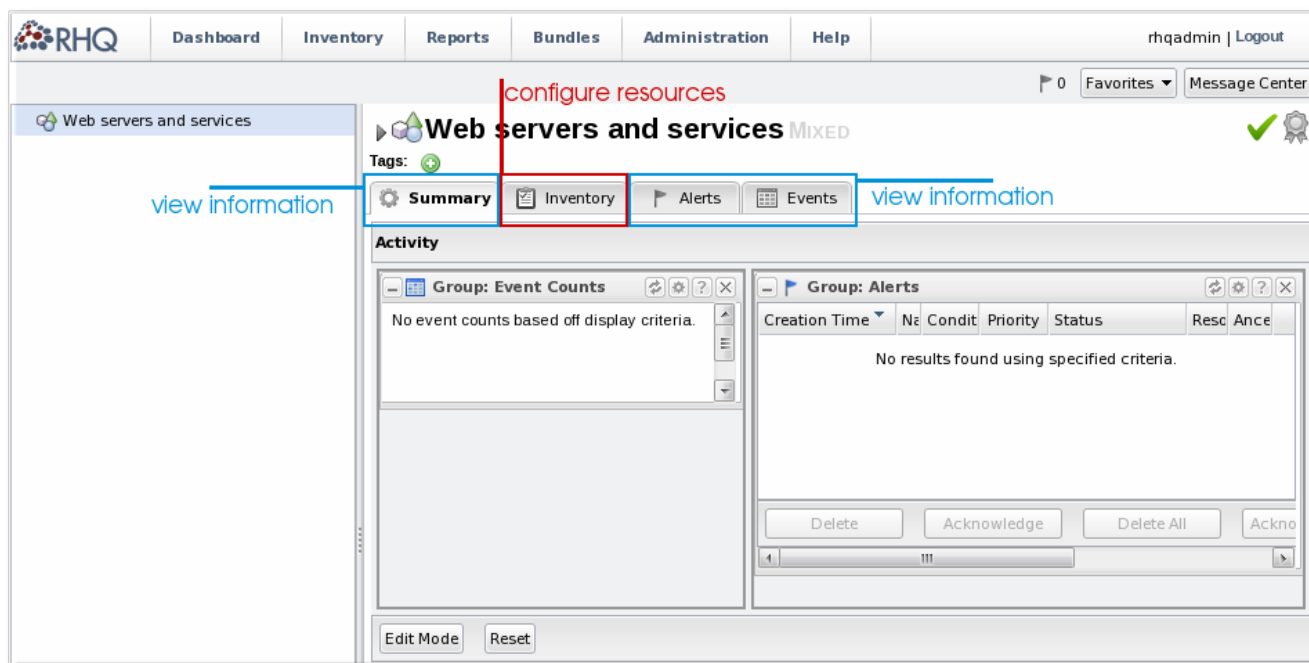


Figure 32. Mixed Group Entry

4.2. Creating Dynamic Groups

A dynamic group 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.

4.2.1. 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"](#)).

4.2.1.1. 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 7. Dynamic Group Properties

Type	Supported Attributes
Related to the resource itself	
resource	id name version parent grandparent children
Related to the resource type	
resourceType	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 7, “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.

4.2.1.2. 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]
```

4.2.1.3. 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 33, “Resources and Parents”](#), the pivot expression creates groups for the three parents within the resource hierarchy: ResourceParentA, ResourceParentB, and ChildA2.

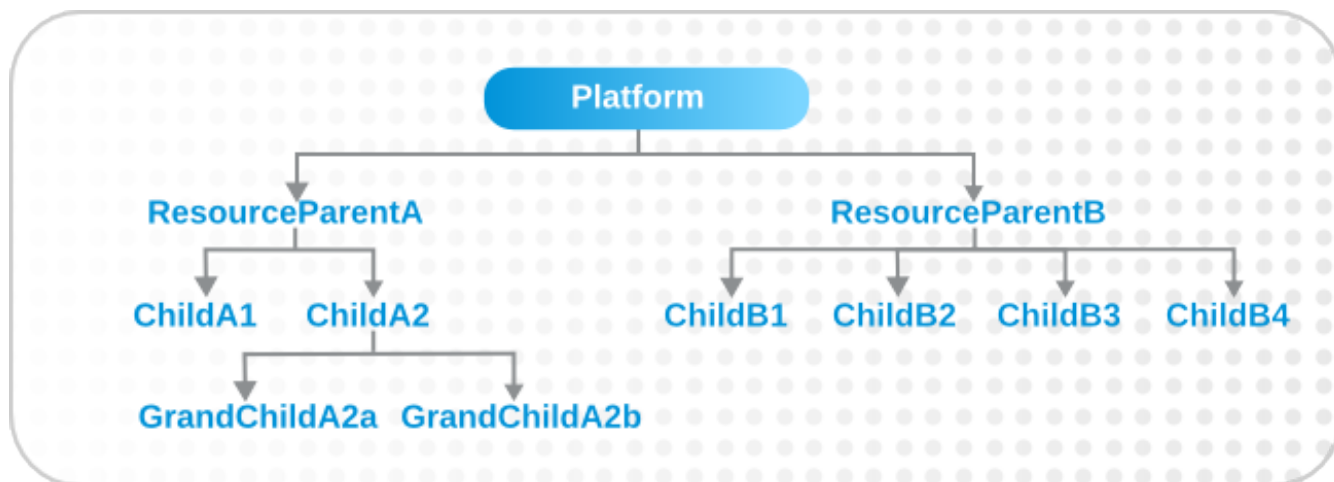


Figure 33. Resources and Parents

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

4.2.1.4. 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]
```

4.2.1.5. Dynagroup Expression Examples

[Table 8, “Dynagroup Examples”](#) contains some dynagroup expressions to create commonly-used groups. 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.

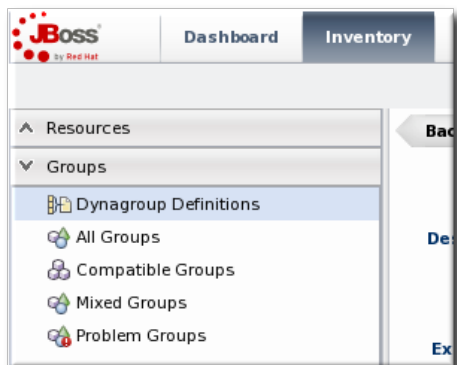
Table 8. Dynagroup Examples

Description	Expression
JBoss clusters	<pre>resource.type.plugin = JBossAS resource.type.name = JBossAS Server groupby resource.traits[partitionName]</pre>
Each operating system type	<pre>resource.type.plugin = Platforms resource.type.category = PLATFORM groupby resource.type.name</pre>
All autogroups  NOTE This could create a large number of groups in large inventories.	<pre>groupby resource.type.plugin groupby resource.type.name groupby resource.parent.name</pre>

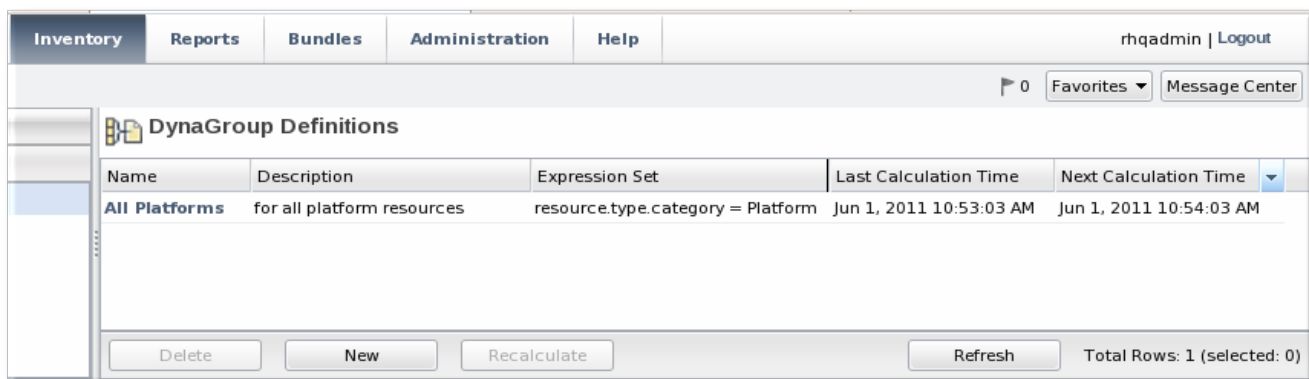
Description	Expression
Just raw measurement tables	<pre>resource.type.plugin= Postgres resource.type.name = Table resource.parent.name = rhq Database resource.name.contains = rhq_meas_data_num_</pre>
Just agents configured with multi-cast server detection	<pre>resource.type.plugin= RHQAgent resource.type.name = RHQ Agent resource.resourceConfiguration[rhq.communications.multicast-detector.enabled] = true</pre>
Just Windows platforms with event tracking	<pre>resource.type.plugin= Platforms resource.type.name = Windows resource.pluginConfiguration[eventTrackingEnabled] = true</pre>
JBoss AS servers grouped by machine	<pre>groupby resource.parent.trait[Trait.hostname] resource.type.plugin = JBossAS resource.type.name = JBossAS Server</pre>

4.2.2. 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.





- 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) :

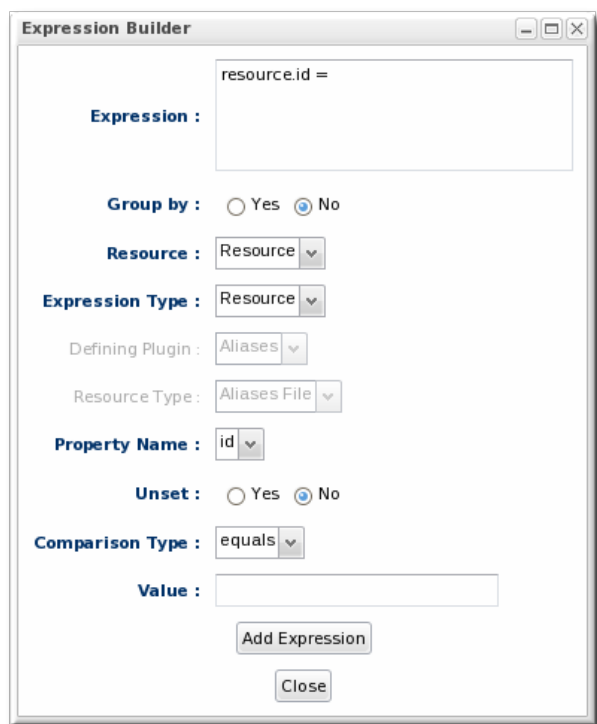
DynaGroup Children

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.



Expression Builder

Expression : resource.id =

Group by : ☐ Yes ☒ No

Resource : Resource

Expression Type : Resource

Defining Plugin : Aliases

Resource Type : Aliases File

Property Name : id

Unset : ☐ Yes ☒ No

Comparison Type : equals

Value :

Add Expression

Close

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

- After entering the expressions, set whether the dynamic group is recursive.
- 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.



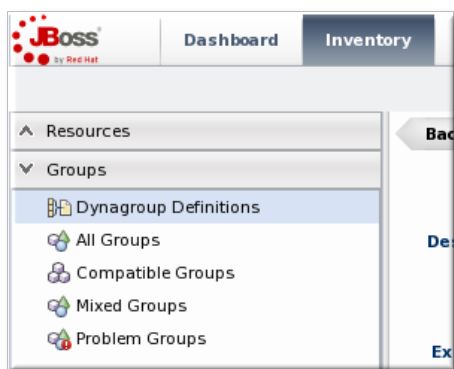
NOTE

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.

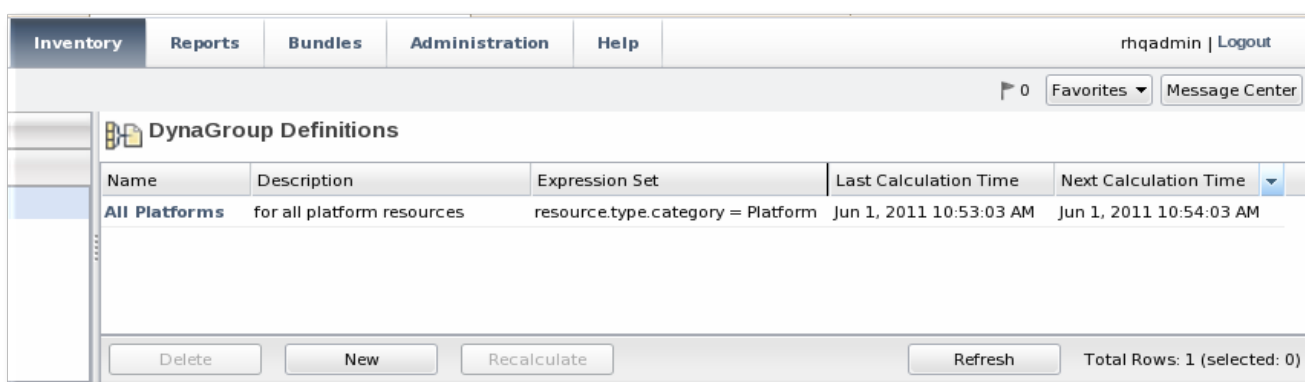
4.2.3. Recalculating Group Members

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.

- 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.



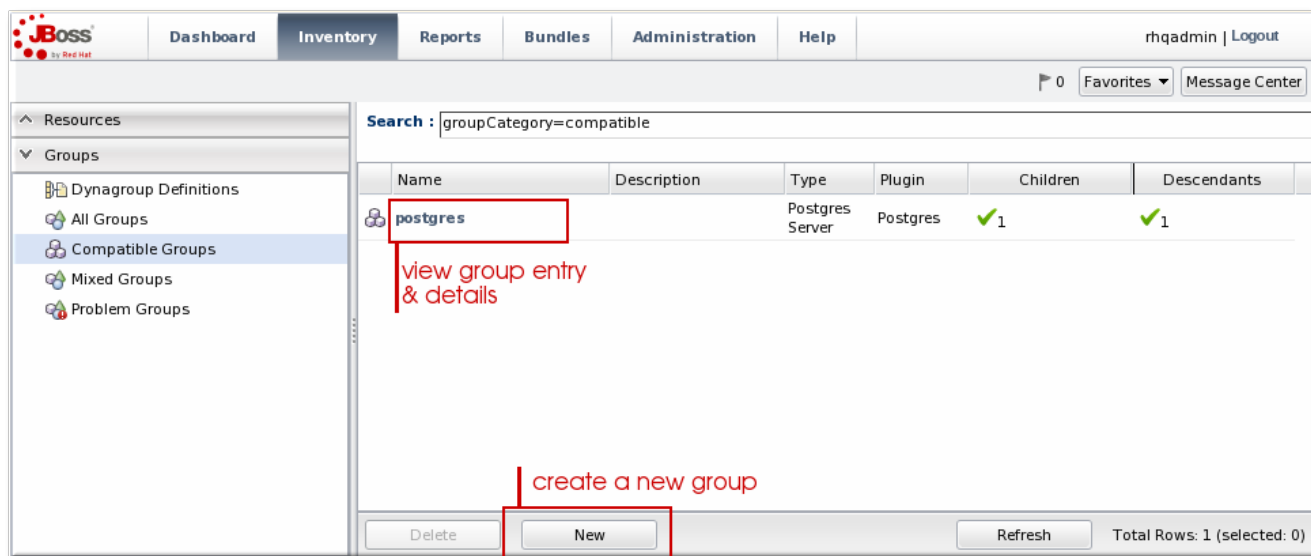
4. Click the **Recalculate** button at the bottom of the table.

4.3. 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 4.1.3, “Comparing Compatible and Mixed Groups”](#).



3. Enter a name and description for the group.

The 'Create Group' dialog box is shown, indicating 'Step 1 of 2'. Under 'Group Settings', the 'Name' field contains 'My Example Group' and the 'Description' field contains 'For a lot of resources.' The 'Recursive' checkbox is checked. At the bottom are 'Cancel', 'Previous', and 'Next' buttons.

Create Group Step 1 of 2

Group Settings

Name : My Example Group

Description : For a lot of resources.

☒ Recursive

Cancel Previous Next

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



NOTE

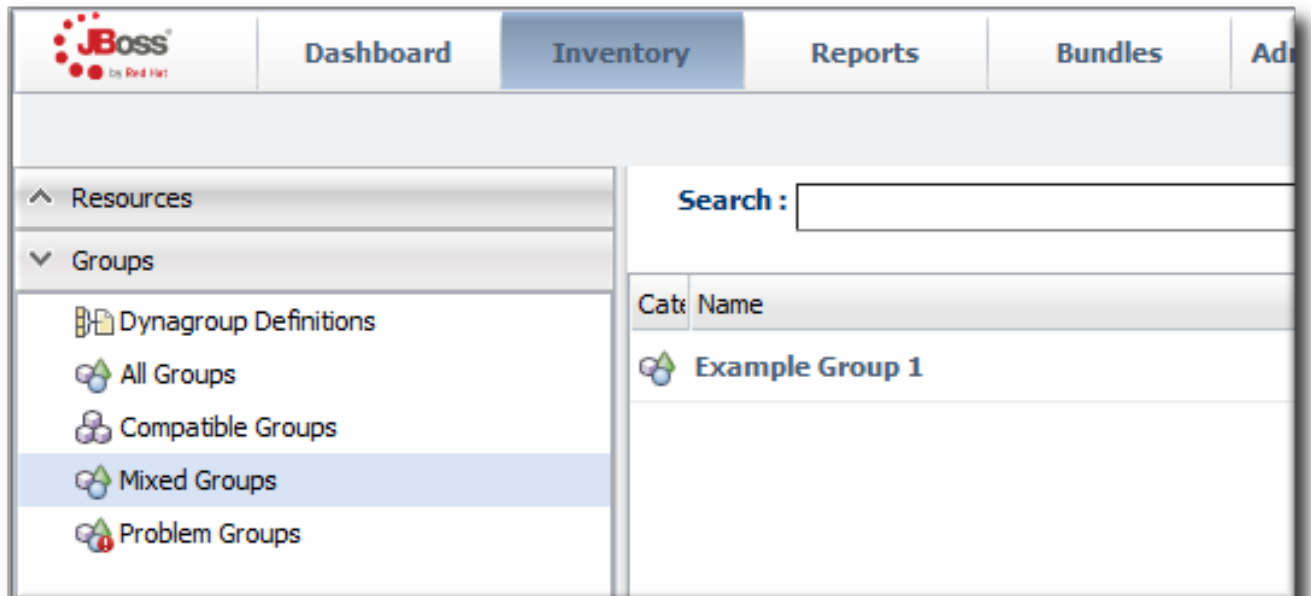
Be very cautious before marking a compatible group as a recursive group. Recursive groups automatically import all children of the members, and if the children are a different resource type than the parent, then the compatible group becomes a mixed group. This limits the management options for the group because group operations, metrics, and configuration changes are only available for compatible groups.

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

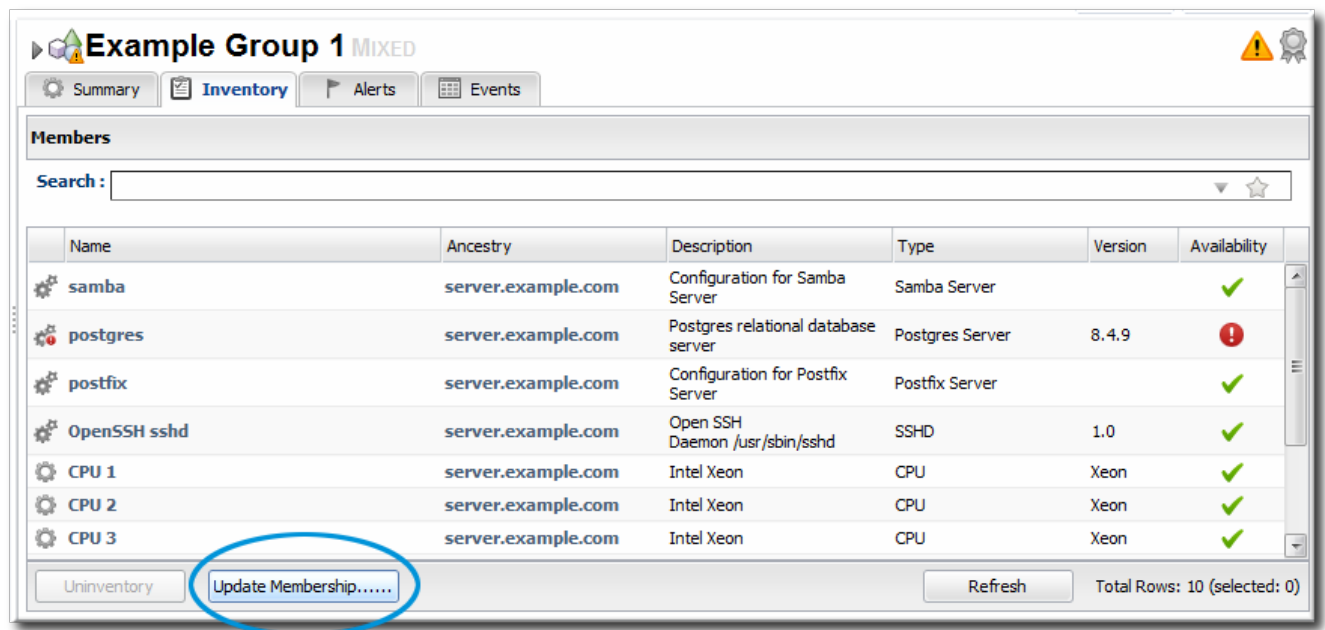
4.4. Changing Group Membership

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.

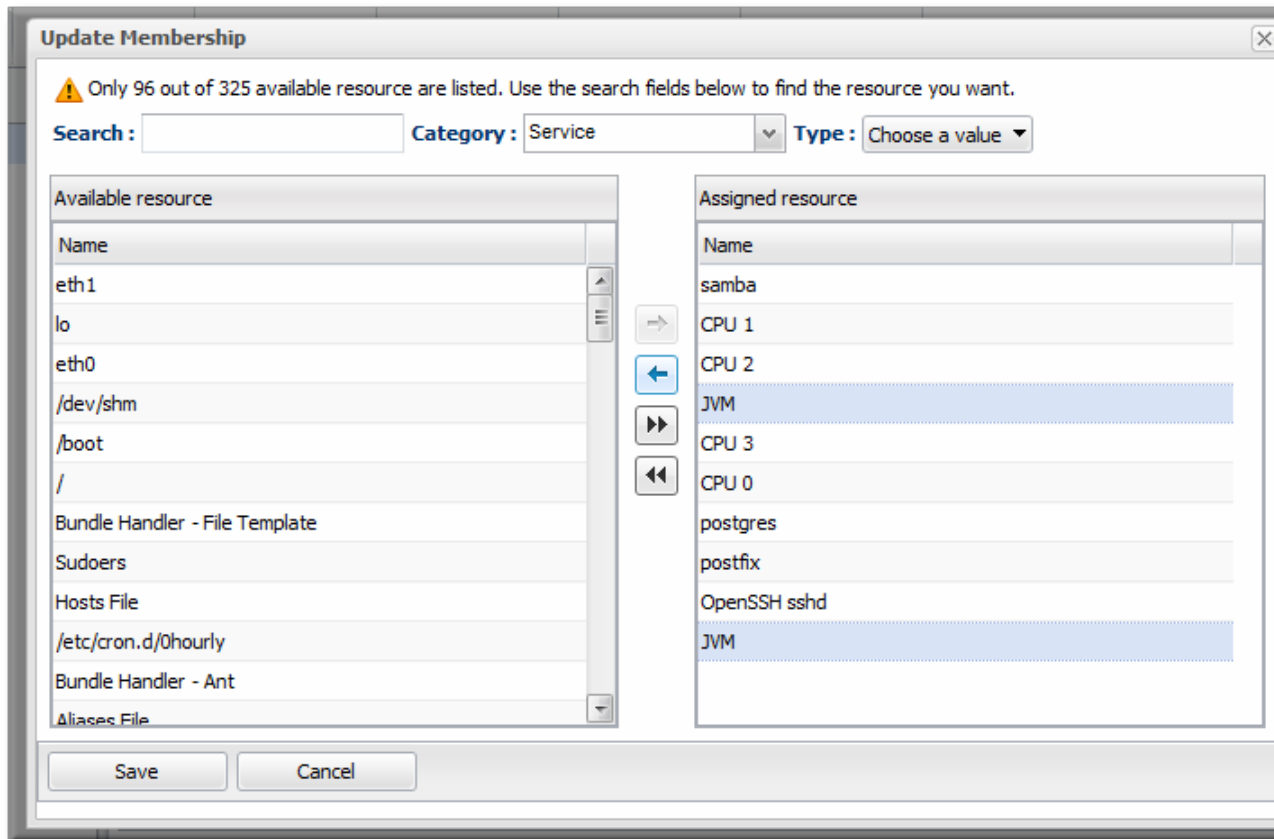
1. In the **Inventory** tab in the top menu, select the compatible or mixed groups item in the **Groups** menu on the left.



2. Click the name of the group.
3. Open the **Inventory** tab for the group, and open the **Members** submenu.
4. Click the **Update Membership** button at the bottom of the page.



5. 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.



6. Click the **Save** button.

4.5. 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.

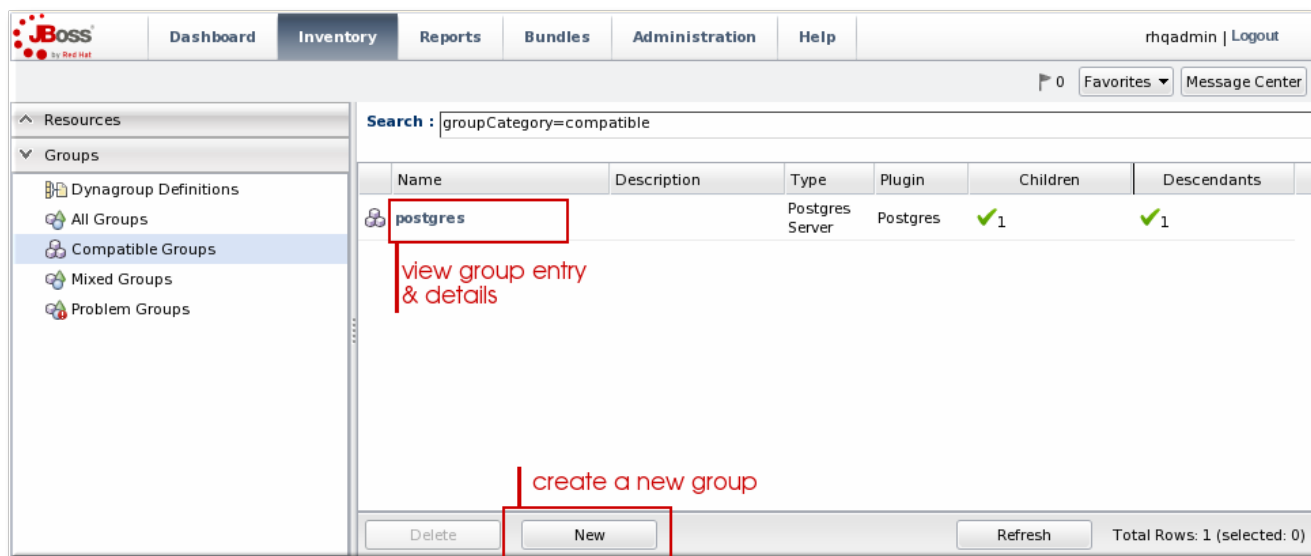
The rules for the values of compatible group connections are simple:

- 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 ~*.

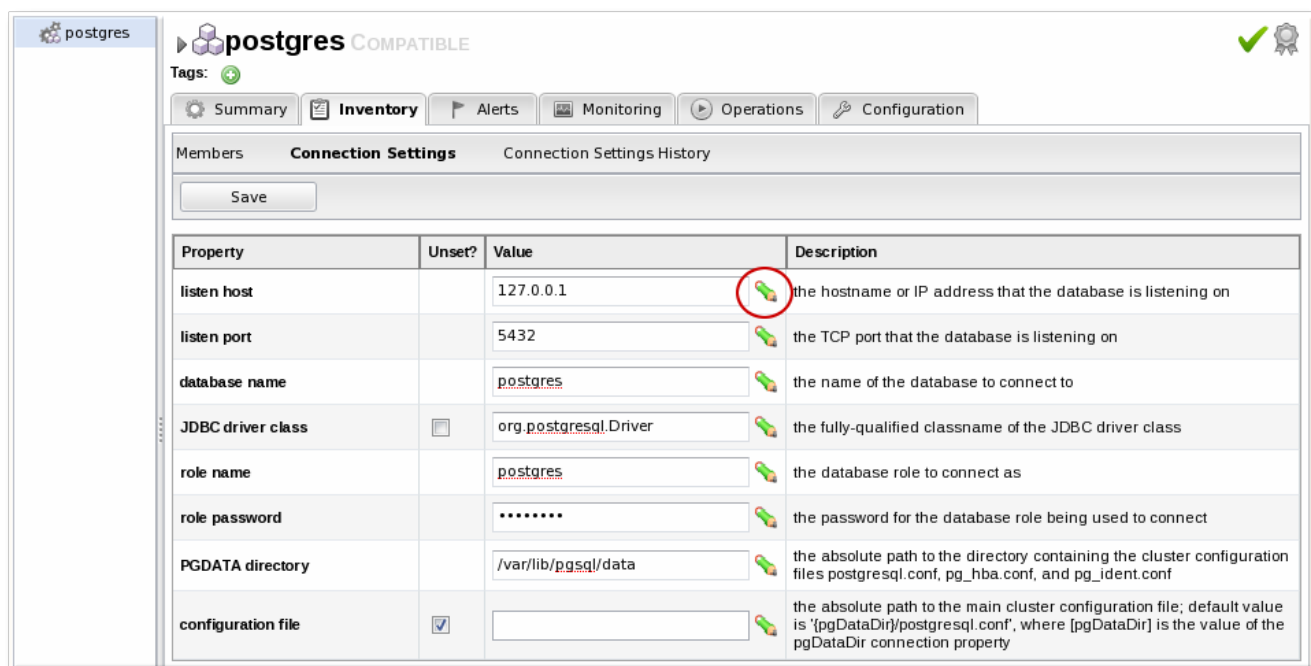
To edit the connection properties:

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

Initial Setup: the Resource Inventory, Groups, and Users



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.

5. 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.

5.1. 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.

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**.

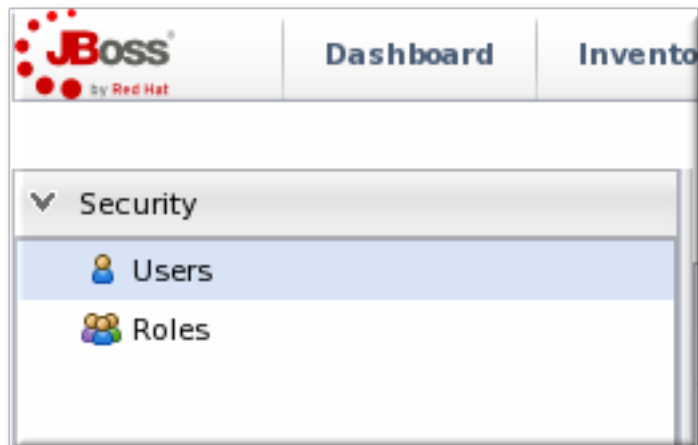


3. Click the name of **rhqadmin**.
4. In the edit user form, change the password to a new, complex value.

A screenshot of the 'Edit User [rhqadmin]' form in the JBoss web application. The form is titled 'Edit User [rhqadmin]' and contains several input fields and checkboxes. The fields are arranged in two columns. The left column contains: 'User Name * : rhqadmin', 'Password * : [masked]', 'First Name * : RHQ', 'Email Address * : rhqadmin@localhost', and 'Department : [empty]'. The right column contains: 'LDAP Login? : no', 'Verify Password * : [masked]', 'Last Name * : Administrator', 'Phone Number : [empty]', and 'Login Enabled? * : yes'. Below the input fields is a section titled 'Assigned Roles' which contains a table with one row: 'Name' and 'Super User Role'.

5.2. 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 has a header 'Create New User' with a person icon. It contains several input fields:

- User Name ***: bjensen
- Password ***: masked with dots
- First Name ***: Barbara
- Email Address ***: bjensen@example.com
- Department**: empty field
- LDAP Login?**: no
- Verify Password ***: masked with dots
- Last Name ***: Jensen
- Phone Number**: empty field
- Login Enabled? ***: radio buttons for 'yes' (selected) and 'no'.

 Below the input fields are two panels:

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

 Between these two panels are four arrow buttons: a single right arrow, a single left arrow, a double right arrow, and a double left 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.

5.3. 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.

The screenshot shows the JBoss ON Administration interface. The top navigation bar includes 'Dashboard', 'Inventory', 'Reports', 'Bundles', 'Administration' (selected), and 'Help'. The user 'rhqadmin' is logged in. On the left, the 'Security' menu is expanded, showing 'Users' and 'Roles'. The main content area is titled 'Edit User [bjensen]'. It contains several form fields: 'User Name' (bjensen), 'Password' (masked), 'First Name' (Barbara), 'Email Address' (bjensen@example.com), 'Department' (empty), 'LDAP Login?' (no), 'Verify Password' (masked), 'Last Name' (Jensen), 'Phone Number' (empty), and 'Login Enabled?' (radio buttons for 'yes' and 'no', with 'yes' selected). Below these fields are two tables: 'Available Roles' and 'Assigned Roles'. 'Available Roles' lists 'Super User Role' and 'All Resources Role'. 'Assigned Roles' lists 'Example Role'.

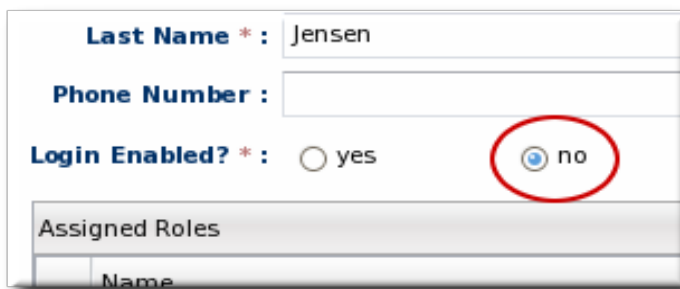
5.4. Disabling User Accounts

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**.



The screenshot shows a user edit form. The 'Last Name' field contains 'Jensen'. The 'Phone Number' field is empty. The 'Login Enabled?' field has two radio buttons: 'yes' and 'no'. The 'no' radio button is selected and circled in red. Below this field is a section titled 'Assigned Roles' with a table header 'Name'.

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

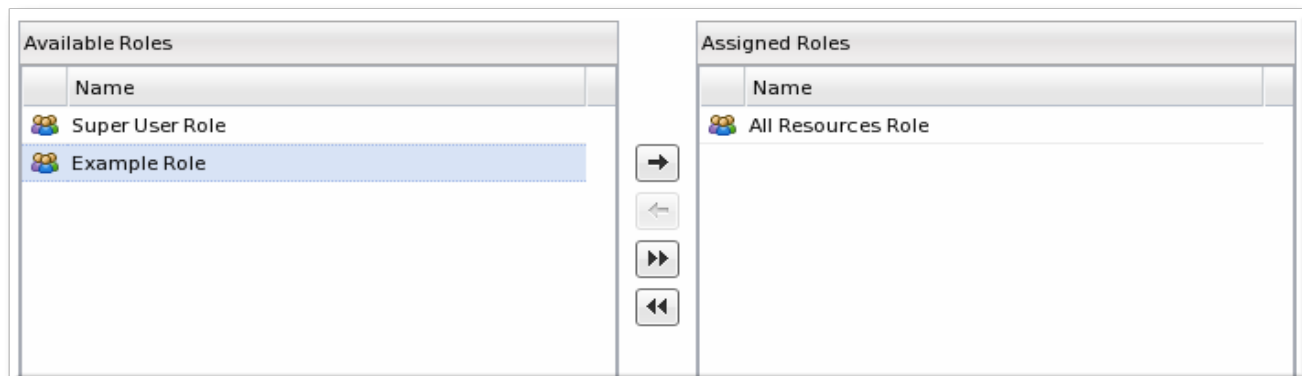
The user account can be re-enabled at any time by changing the **Enable Login** value back to **Yes**.

5.5. 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.

6. 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.

6.1. About Security in JBoss ON: Roles and Access Control

In JBoss Operations Network, security is achieved by establishing 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.

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 categories where access control is given:

- *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 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 9, “JBoss ON Access Control Definitions”](#).

For resource-level rights, all of the rights granted are explicit modify rights. This means that giving the right allows users to do something on the resource. For the most part, the ability to view that area (read rights) are granted implicitly. 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. The read access is implied. There is one exception to

that rule: resource configuration. Resource configuration can be considered a security risk, and some administrators may want to restrict read access to configuration. If write access is given to resource configuration, then read access is automatically granted. However, read-only access must be explicitly granted to a role; otherwise, read access is denied by default.







Resource Permissions			
Name	Read?	Write?	Description
 Inventory	✓	✓	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	✓	✓	Read: (IMPLIED) view metric data and collection schedules Write: update metric collection schedules
 Manage Alerts	✓	✓	Read: (IMPLIED) view alert definitions and alert history Write: create, update, and delete alert definitions; acknowledge and delete alert history items
 Configure	☐	✓	Read: view Resource configuration and Resource configuration revision history Write: update Resource configuration; delete Resource configuration revision history items
 Control	✓	✓	Read: (IMPLIED) view available operations and operation execution history Write: execute operations; delete operation execution history items
 Manage Events	✓	✓	Read: (IMPLIED) view events


Figure 34. 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 9. JBoss ON Access Control Definitions

Access Control Type	Description
Global - Security	<p>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.</p> <div>  WARNING </div> <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>

Access Control Type	Description
Global - Inventory	Allows any operation to be performed on any JBoss ON resource, including importing new resources.
Global - 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.
Global - Bundles	Allows a user to upload and manage bundles (packages) used for provisioning resources.
Global - 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.
Resource - Modify	Allows a user to change the resource definition entry in JBoss ON. This does not grant rights to edit the resource configuration.
Resource - Delete	Allows the user to delete the resource from the inventory.
Resource - Create Child	Allows the user to manually assign a child resource to another resource.
Resource - 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.
Resource - Measurements	Allows the user to configure monitoring settings for the resource.
Resource - Content	Allows the user to manage content providers and repositories that are available to resources.
Resource - Control	Allows a user to run operations (which are also called <i>control actions</i>) on a resource.
Resource - Configure	<p>Allows users to change the configuration settings on the resource through JBoss ON.</p> <div data-bbox="764 1523 858 1619"> </div> <div data-bbox="858 1547 967 1585"> NOTE </div> <div data-bbox="785 1632 1275 1736"> <p>The user still must have adequate permissions on the resource to allow the configuration changes to be made.</p> </div> <p>This access area has two options:</p> <ul style="list-style-type: none"> • Read, which grants read-only access to the resource configuration

Access Control Type	Description
	<ul style="list-style-type: none"> • 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>

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 4.3, “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 7.5, “Associating LDAP User Groups to Roles in JBoss ON”](#)). 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 7.3, “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.

6.2. Creating a New Role



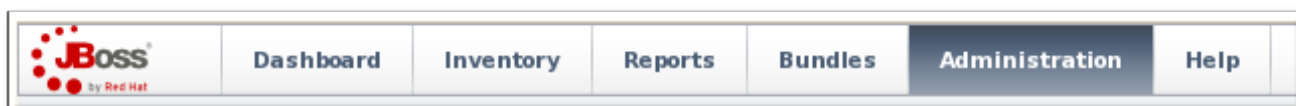
NOTE

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 4.3, “Creating Groups”](#).

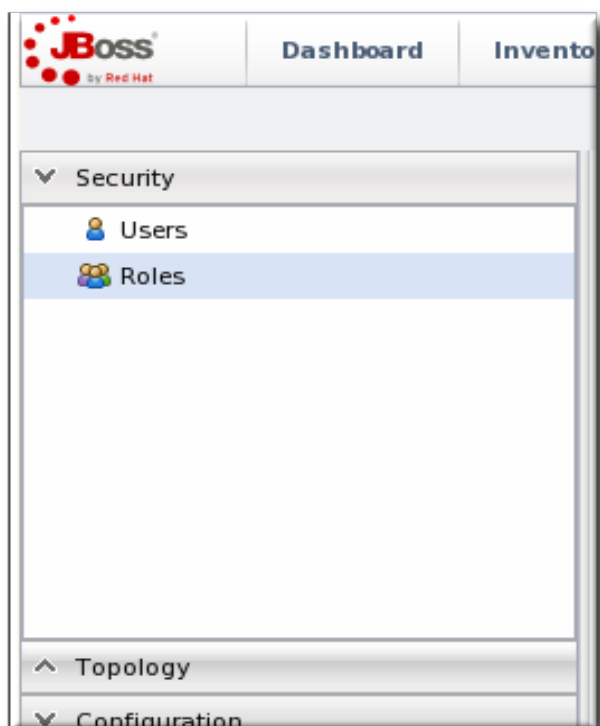
1. Create any resources groups which will be associated with the role. Creating groups is described in [Section 4.3, “Creating Groups”](#).

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 7.5, “Associating LDAP User Groups to Roles in JBoss ON”](#).

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:
 - *Global permissions* grant permissions to areas of the JBoss ON server and configuration.
 - *Resource permissions* grant permissions for managing resources.



Create New Role


Name * :

Description :







Permissions


 Resource Groups



 Users


 LDAP Groups

Global Permissions

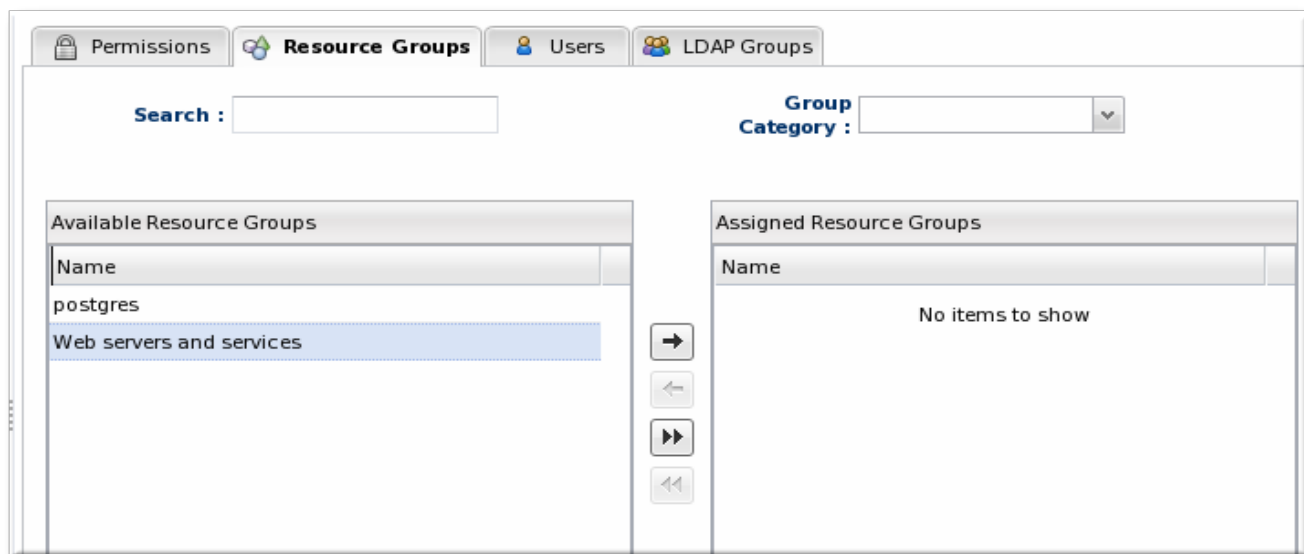
	Name	Authorized	Description
	Manage Security	<input type="checkbox"/>	can create, update, or delete users and roles (viewing is implied for everyone)
	Manage Inventory	<input checked="" type="checkbox"/>	has all Resource permissions, as described below, for all Resources; can create, update, and delete groups; and can import auto-discovered or manually discovered Resources
	Manage Settings	<input type="checkbox"/>	can modify the RHQ Server configuration and perform any Server-related functionality
	Manage Bundles	<input checked="" type="checkbox"/>	can create, update, or delete provisioning bundles (viewing is implied for everyone)
	Manage Repositories	<input checked="" type="checkbox"/>	can create, update, or delete repositories of any user (everyone can create their own repositories), can associate content sources to repositories.

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 Read: (IMPLIED) view metric data and collection schedules

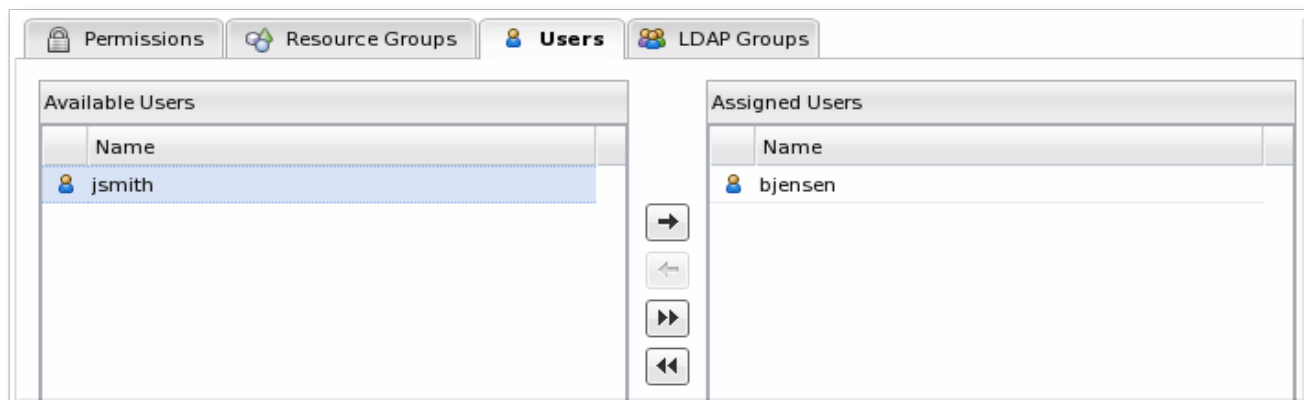
The specific access permissions are described in [Table 9, “JBoss ON Access Control Definitions”](#).

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.

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

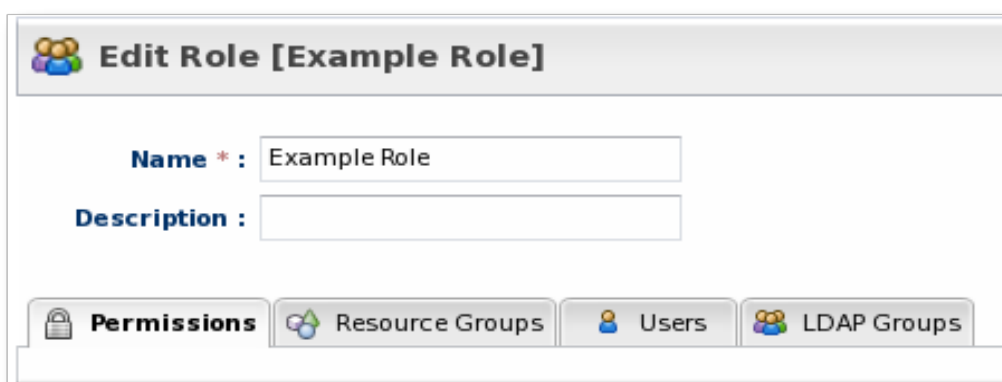
6.3. Editing Roles

All of the role configuration — including the access rights set for the roles and the groups and users assigned to the role — can be changed after the role is created. The edit options for a role are exactly the same as the creation options in [Section 6.2, “Creating a New Role”](#), even to changing the name of the role.

1. In the **Security** menu table on the left, select the **Roles** item, and click the name of the role to edit.



- Go through the role's tabs and change the configuration as desired.



7. 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 [Issues Related to Using LDAP for a User Store](#).

7.1. Supported Directory Services

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

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

7.2. How JBoss ON Uses LDAP for Authentication

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.



TIP

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.

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.

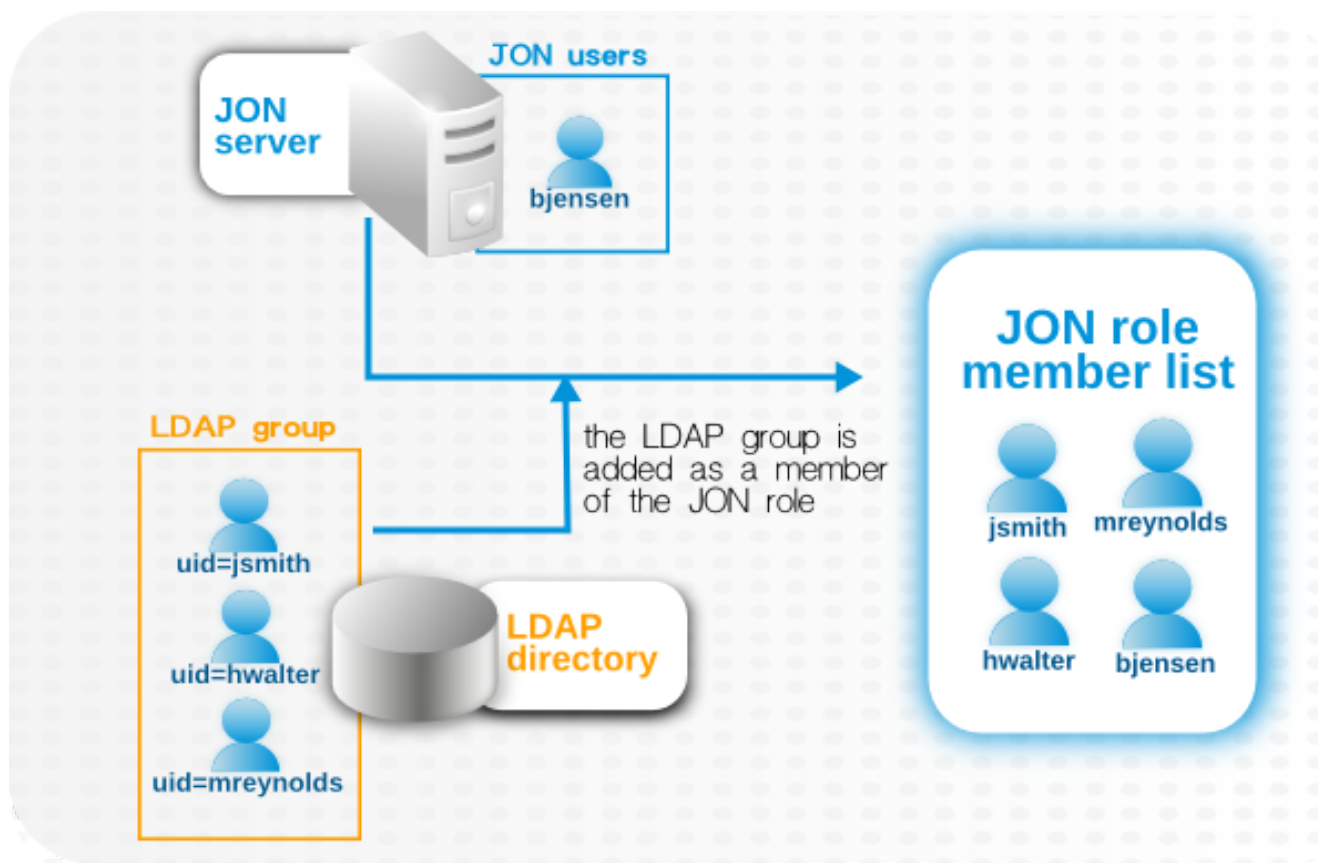


Figure 35. LDAP Groups, JBoss ON Roles, and Role Members

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.

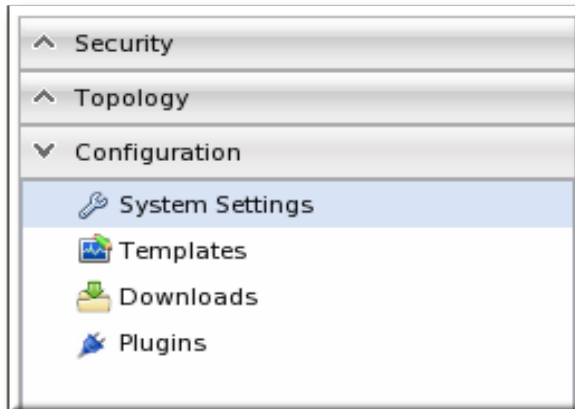
7.3. Configuring LDAP User Authentication

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.

A screenshot of the 'LDAP Configuration Properties' window. The 'Jump to Section' dropdown is set to 'LDAP Configuration Properties'. The table below shows the configuration:

Property	Unset?	Value	Description
Enable LDAP		<input checked="" type="radio"/> Yes <input type="radio"/> No	Should LDAP be used to determine user identity?
Search Base	<input type="checkbox"/>	dc=csb	The base of the directory tree to search for usernames and passwords while authenticating users, e.g. ou=People,dc=redhat,dc=com

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

A screenshot of the 'LDAP Configuration Properties' window. The 'Jump to Section' dropdown is set to 'LDAP Configuration Properties'. The table below shows the configuration:

Property	Unset?	Value	Description
Enable LDAP		<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

Save

- 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).

- 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
```

- 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.

- 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 7.5, "Associating LDAP User Groups to Roles in JBoss ON"](#).

7.4. How JBoss ON Roles Work with LDAP User Groups

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.

Edit Role [Example Role]

Name * :

Description :

Permissions Resource Groups Users **LDAP Groups**

Search :

Available LDAP Groups		Assigned LDAP Groups	
Name		Name	
cn=Jboss Admins		No items to show	
cn=Postres Admins			
cn=Dev Dept			

Navigation buttons: → ← ⇨ ⇩

Figure 36. 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.



TIP

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

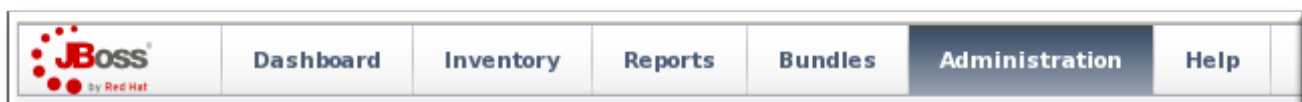
7.5. Associating LDAP User Groups to Roles in JBoss ON

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.

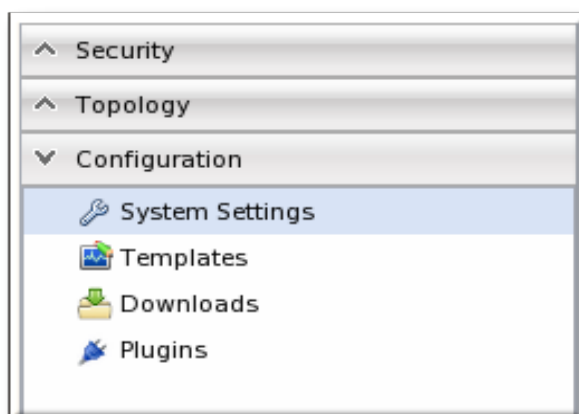
[Section 7.3, “Configuring LDAP User Authentication”](#) describes how an LDAP directory server can be used to authenticate users to JBoss ON. Any time a user attempts to log in, that request — with the username and password — is simply forwarded to the specified LDAP directory server to see if the credentials 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 7.3, “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))
```

- 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)
```

- 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.

8. 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.

8.1. Giving Feedback

If there is any error in this *Basic Admin Guide* 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 **Other** products group.
2. Select **RHQ Project** from the list.
3. Set the component to **Documentation**.
4. Set the version number to 3.0.
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. You are welcome to contact Red Hat Content Services directly at docs@redhat.com².

8.2. Document History

Revision 3.0-0 December 7, 2011

Ella Deon Lackey dlackey@redhat.com

Initial release of JBoss O 3.0.

Index

A

- access controls, 68
 - about read rights, 68
 - applying to resources, 73
 - assigning to users, 67
 - changing on roles, 74

² <mailto:docs@redhat.com>

- list of permissions, 69
- users, 63
- authentication
 - and LDAP, 78
 - configuring LDAP, 78
 - storing credentials, 76
- authorization
 - for LDAP groups and roles, 81

B

- boundary characters
 - in searches, 25

D

- discovery
 - ignoring resources, 35
 - importing, 34
 - manual, 32
 - resources, 34
- dynamic search
 - groups and resources, 20
 - null, 25
 - saving, reusing, and deleting, 28
 - syntax, 22
 - using quotation marks, 25

E

- entries
 - tagging, 15

G

- groups
 - and roles, 71
 - assigning to roles, 73
 - changing role assignments, 74
 - dynamic search, 20
 - empty groups, 51
 - overview, 45
 - search context, 26

I

- importing
 - discovery, 34
- inventory
 - importing, 29
 - overview, 29
 - reports, 44

J

- JBoss ON
 - access control, 68
 - and LDAP, 78
 - configuring, 78

- authorization, 81
- supported LDAP servers, 76

L

LDAP

- assigning groups to roles, 71
- authorization, 81
- for authentication, 78
 - configuring, 78
 - configuring SSL, 80
- supported servers, 76
- user groups and roles, 82
 - LDAP group object classes, 82
 - member attributes, 82
 - search parameters, 81
- verifying credentials, 76

P

- permissions, 68
 - applying to groups, 73
 - assigning to users, 67
 - changing on roles, 74
 - list of, 69

R

reports

- inventory, 44

resources

- access control, 68
- access permissions on, 69
- and managing inventory, 29
- and roles, 71
- creating children, 38
- deleting entries, 20
- discovery and imports, 34
- dynamic search, 20
- groups, 45
- ignoring discovered resources, 35
- inventory reports, 44
- managed, 31
- manual discovery, 32
- running discovery, 34
- search context, 26
- uninventory, 41

roles, 71

- adding or removing users, 67
- and groups, 71
- and LDAP user groups, 82
 - LDAP group object classes, 82
 - member attributes, 82
 - searching for members, 81
- and users, 71
- assigning groups, 73

- assigning users from LDAP groups, 71
- changing member groups, 74
- changing permissions, 74
- creating, 71
- default, 71
- setting permissions, 73
- types of members, 71

S

- search
 - groups context, 26
 - resource context, 26
 - string operators, 27
 - using quotation marks, 25
- secure connections
 - using for LDAP authentication, 80
- security
 - users, 63
- self-registering
 - and LDAP, 76
- server
 - access control, 68
 - global rights, 69
 - resource-level rights, 69
 - and LDAP groups for roles, 82
 - building LDAP search, 81
 - LDAP group object classes, 82
 - member attributes, 82
 - LDAP authentication, 78
 - configuring, 78
- SSL
 - using for LDAP authentication, 80

U

- UI
 - A Tour of the UI, 3
 - access control, 68
 - dashboard, 8
 - deleting entries, 20
 - details page, 9
 - inventory, 8
 - left menu, 6
 - logging in, 3
 - setting favorites, 19
- users
 - and roles, 71
 - assigning LDAP user groups to roles, 71
 - authentication, 78
 - changing access controls, 67
 - changing roles, 67
 - configuring LDAP authentication, 78
 - creating new, 64
 - disabling accounts, 66

editing details, 66

security, 63

using LDAP to self-register, 76