



# **Red Hat Satellite 5.8 Getting Started Guide**

---

Basic setup and configuration with Red Hat Satellite

Red Hat Satellite Documentation Team



# Red Hat Satellite 5.8 Getting Started Guide

---

## Basic setup and configuration with Red Hat Satellite

Red Hat Satellite Documentation Team  
satellite-doc-list@redhat.com

## Legal Notice

Copyright © 2017 Red Hat.

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

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

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

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

## Abstract

This document contains information and instructions for users initially starting out with Red Hat Satellite. It explains basic concepts and configuration procedures. For more about Satellite basics, also see the Red Hat Satellite User Guide.

## Table of Contents

<b>Chapter 1. Channel Management</b>	<b>2</b>
1.1. Managing Red Hat Network Channels	2
1.2. Creating and Managing Custom Channels	6
1.3. Managing Configuration Channels	22
<b>Chapter 2. Organizations</b>	<b>26</b>
2.1. Creating an Organization	26
2.2. Managing Organization Entitlements	27
2.3. Managing Multiple Organizations	28
<b>Chapter 3. System Provisioning</b>	<b>36</b>
3.1. Provisioning Through Red Hat Satellite	36
3.2. Provisioning Through Red Hat Satellite Proxy	60
3.3. Provisioning Virtualized Guests	60
3.4. Kickstarting through Cloned or Custom Channels	61
3.5. Reprovisioning	62
3.6. Provisioning Rollbacks with Snapshots	63
<b>Chapter 4. System Management</b>	<b>66</b>
4.1. Registering Systems to Satellite	66
4.2. Managing Systems with Satellite	80
4.3. Managing Virtualized Client Systems	91
<b>Chapter 5. Errata Management</b>	<b>103</b>
5.1. Creating and Editing Errata	103
5.2. Assigning Packages to Errata	104
5.3. Publishing Errata	104
5.4. Cloning Errata	104
<b>Appendix A. Boot Devices</b>	<b>106</b>
<b>Appendix B. Custom Package Management</b>	<b>111</b>
B.1. Building Packages for Red Hat Network	111
B.2. Digital Signatures for Red Hat Network Packages	113
<b>Appendix C. Revision History</b>	<b>117</b>

## Chapter 1. Channel Management

A Red Hat Satellite channel is a collection of software packages. Channels help you segregate packages by sensible rules: a channel may contain packages from a specific Red Hat distribution. A channel may contain packages for an application or family of applications. Users may also define channels for their own particular needs; a company may create a channel that contains packages for a specific architecture in the organization's network.

This chapter will discuss two basic channel types available in Red Hat Satellite:

1. Red Hat Channels - are official Red Hat repositories containing Red Hat released packages.
2. Custom Channels - are channels created by a Satellite administrator based on an organization or group's needs. These are managed by the organization and may contain third-party packages and repositories.

### 1.1. Managing Red Hat Network Channels

A Red Hat Satellite channel is a collection of software packages.

#### 1.1.1. Differentiating Base Channels and Child Channels

There are two types of channels: *base channels* and *child channels*. A base channel consists of packages based on a specific architecture and Red Hat Enterprise Linux release. A child channel is a channel associated with a base channel that contains extra packages.

A system must be subscribed to only one base channel. A system can be subscribed to multiple child channels of its base channel. A subscribed system can only install or update packages available through its Satellite channels.

When a system is registered with Satellite, it is assigned to the base channel that corresponds to the system's version of Red Hat Enterprise Linux. Once a system is registered, its default base channel can be changed to a private base channel on a per-system basis via the Satellite web interface or APIs. Alternatively, activation keys associated with a custom base channel can be used so that systems registering with those keys are automatically associated with the custom base channel.

On the Satellite web interface, the **Channels** page (located under the **Channels** tab on the top navigation bar) provides a list of all base channels and their child channels. Clicking on the name of a channel displays the **Channel Details** page, which provides a list of all of the packages in that channel, its errata, and any associated systems.

#### 1.1.2. Subscribing a System to the Red Hat Satellite

Subscribe systems to channels in the following ways:

- ✳ Registration through activation keys - Because of the simplicity and speed of activation keys, this is the preferred method for registering systems as clients of either Red Hat Satellite Proxy Server or Red Hat Satellite Server. Systems registered using an activation key are subscribed to all channels associated with that activation key. For more information about activation keys, consult the *Red Hat Satellite Client Configuration Guide* and the *Red Hat Satellite Reference Guide*.
- ✳ Install registration - After initial registration through either the **Red Hat Update Agent** or the **Red Hat Network Registration Client**, it is automatically assigned to the base channel that corresponds to the version of Red Hat Enterprise Linux on the system. Once a system is registered, its default base channel can be changed to a private base channel on a per-system basis using Red Hat Satellite. For more

information about using these applications, see the respective chapter of the *Red Hat Satellite Reference Guide* for your entitlement level (Management or Provisioning).

- ✳ Website subscription - Various specific child channels are available for subscription, depending on the system's base channel. The system may be subscribed to the child channel through the Satellite web interface. If the organization has created custom base channels, the systems may also be reassigned to these custom channels through the website. For more information about subscribing to channels online, see the Satellite web interface chapter of the *Red Hat Satellite Reference Guide*.
- ✳ Using the **spacewalk-channel** command-line tool (CLI) - the **spacewalk-channel** allows you to subscribe systems to specific channels via the command line without logging on to the Red Hat Network website.

For example, to subscribe to two channels:

```
# spacewalk-channel --add -c rhn-tools-rhel-x86_64-server-6 -c \
  rhel-x86_64-server-6 --user username --password password
```

To unsubscribe from the channels:

```
# spacewalk-channel --remove -c rhn-tools-rhel-x86_64-server-6 -c \
  rhel-x86_64-server-6 --user username --password password
```

To list subscribed channels:

```
# spacewalk-channel --list
```

### 1.1.3. Deleting a Red Hat Base Channel from Red Hat Satellite

There are several situations that would require administrators to remove Red Hat base channels. Some examples are:

- ✳ The specific architecture is not supported by the organization and should not be available.
- ✳ The subscription for the specific architecture has expired and updates are unavailable.
- ✳ The product channel has reached end of life.
- ✳ The Red Hat Satellite server needs disk space.

#### Prerequisites:

The *spacewalk-backend-tools*, version 0.5.28.49 or newer, is required to run the commands below.

To delete a Red Hat base channel from the Red Hat Satellite:

1. Log in as root on the Red Hat Satellite server.
2. List all subscribed channels the Satellite is subscribed to:

```
# spacewalk-remove-channel --list
```

Take note of the channel to be removed. This is called the **channel\_label** and will be used in the next step.

3. Remove the channel from the Satellite:

```
# /usr/bin/spacewalk-remove-channel -c channel_label --unsubscribe
```

Where:

- ✳ **--unsubscribe** will automatically unsubscribe all systems subscribed to the channel being removed.



## Note

This procedure is only applicable to Red Hat Satellite 5.3 and newer. For Red Hat Satellite 5.2 or older, a script can be used to remove the Red Hat provided channels from Red Hat Satellite. See the Knowledgebase article [How do I delete a Red Hat Base channel from my Red Hat Satellite?](#)

### 1.1.4. Managing Repositories

A repository contains a set of packages much like a channel. However, while a channel can house several repositories, a repository contains only package sets. In Red Hat Satellite, most repositories are created using **yum** tools. These repositories are added to the Red Hat Satellite's custom channels in order to provide packages for specific applications, usually third-party.

#### 1.1.4.1. Adding Repositories

Repositories can be added to allow additional packages from different sources to synchronize with your custom channels. The repository URL must point to the root directory of a yum repository.

1. Log in as a Channel Administrator or Organizational Administrator.
2. Click on **Channels** → **Manage Software Channels** → **Manage Repositories**.
3. On the top right-hand corner of the page, click **Create New Repository**.
4. Fill in the following:
  - ✳ **Repository Label** - An identifying name for the repository
  - ✳ **Repository URL** - A valid URL where the repository resides
  - ✳ **SSL CA Certificate** - The SSL Certificate Authority required for accessing repositories over HTTPS
  - ✳ **SSL Client Certificate** - The SSL Certificate required for accessing repositories over HTTPS
  - ✳ **SSL Client Key** - The SSL Key required for accessing repositories over HTTPS



## Note

Create new SSL certificates and manage existing ones by navigating to **Systems** → **Kickstart** → **GPG and SSL Keys**.



- ✳ **Filters** - Defines a package filter to apply to the repository. Use plus (+) to include packages, or minus (-) to exclude packages. The following example filters out the **kernel** package and any packages beginning with **zsh** except for **zsh-html**:

```
-zsh*,kernel +zsh-html
```

5. Click **Create Repository**.

#### 1.1.4.2. Adding Repositories to a Channel

In order to use repositories on the Red Hat Satellite, repositories need to be added to a channel. A repository can be added to as many channels as needed.

To add a repository to a channel:

1. Click **Channels** → **Manage Software Channels**.
2. Choose the specific channel you wish the repository to be a part of.
3. Click on the subtab **Repositories** and select the repositories you wish to add to the channel.
4. Select the repositories to be added to the channel.
5. Click **Update Repositories**.

#### 1.1.4.3. Scheduling Repository Synchronization

Source repositories may change or update based on upgrades, security and bug fixes. The Red Hat Satellite's custom repositories need to be synchronized with these source repositories to make sure that the packages being imported are at their most updated versions. Additionally, if new packages have been added to the source repositories, these packages will also be downloaded to the Red Hat Satellite custom repositories.

To schedule a repository synchronization:

1. Click **Channels** → **Manage Software Channels**.
2. Choose the channel the repository is a member of.
3. Click on the subtab **Repositories** → **Sync**.
4. Choose when you wish to schedule the synchronization. Click **Sync Now** to schedule a sync immediately or select a schedule with the options:
  - ✳ **Disable Schedule** - disables any schedule currently in place.
  - ✳ **Daily** - schedules a daily synchronization with source repositories at a specified time.
  - ✳ **Weekly** - schedules a weekly synchronization with source repositories at a specified day and time.
  - ✳ **Monthly** - schedules a monthly synchronization on a specified day of the month and time.
  - ✳ **Custom Quartz Format** - Defines a custom schedule for synchronization.
5. Click **Schedule** to save your changes and schedule the synchronization.

**Note**

The packages are assigned to the channel after the synchronizing is complete.

#### 1.1.4.4. Deleting Repositories

1. Log in as a Channel Administrator or Organizational Administrator.
2. Click on **Channels** → **Manage Software Packages** → **Manage Repositories**.
3. Choose the repository you wish to delete.
4. On the top right-hand corner of the page, click **delete repository**.

## 1.2. Creating and Managing Custom Channels

There are many channels in Red Hat Satellite. Some are available to all users, some are available to users in a particular organization, and some are available only if subscriptions have been purchased for access to specific channels. Channels fall into these main categories:

- ✦ **Paid Service Channels** - These channels are available if the organization has purchased access to them either directly or in conjunction with a particular Red Hat solution. Red Hat Enterprise Linux is an example of a paid service channel.
- ✦ **Custom Channels** - These channels are created by the organizational administrator to manage custom packages. These channels, also known as *private channels*, by default, appear only to the organization who creates them; they can never be accessed by anyone else. However, private channels can be shared across organizations by setting up Organizational Trusts and sharing the channel.

Custom channels allow administrators to use the Red Hat Satellite infrastructure to deploy packages built and maintained by their organizations. All channel and package management activities take place in the **Channels** tab of the Red Hat Satellite web interface.

**Note**

Because of the potential problems that may arise from deploying untested packages throughout the production environment, Red Hat strongly recommends creating beta channels covering select systems that can be used for staging.

For example, if the organization has a system group of Web servers that receives a set of custom packages, create temporary channels to install the packages on a non-critical subset of representative systems first. These might be development or staging servers, *not* live production systems. These temporary channels are then deleted using the steps described in [Section 1.2.11, “Deleting Software Channels”](#).

### 1.2.1. Tools, Repositories, and Practices

Before creating and managing channels, note the differences between the various tools and repositories at your disposal. This is especially important when deploying both a Red Hat Satellite Server and Red Hat Satellite Proxy Server, as this increases the utilities and storage locations available. Furthermore, a Proxy-Satellite combination offers certain best practices for optimal performance.

The following package management tools are used in Red Hat Satellite:

- **Red Hat Network Package Manager** - Use this to push custom packages into custom channels on your Red Hat Satellite Proxy Server.
- **Red Hat Network Push** - Use this to push custom packages into custom channels on your Red Hat Satellite Server.
- **Red Hat Satellite Synchronization Tool** - Use this to import and synchronize standard packages from Red Hat Network Classic to the Red Hat Satellite server at your location. This is done via the Internet or CD/DVD ISO images.

Each of these tools has a corresponding package repository. Both **Red Hat Network Package Manager** and **Red Hat Network Push** require the creation of a temporary staging directory for placement of custom packages that are uploaded to the Proxy or Satellite. Delete these staging directories after use.



### Note

Red Hat recommends archiving custom packages externally from Red Hat Satellite.

When using both Red Hat Satellite Proxy Server and Red Hat Satellite, use only **Red Hat Network Push** and **Red Hat Satellite Synchronization Tool**. The Proxy-Satellite combination requires custom packages and channels be uploaded to the *Satellite only*. From there, the Proxy obtains the packages and distributes them to client systems.

## 1.2.2. Creating a Software Channel

Before uploading packages to the server, a custom channel can be created to house them. See [Section 1.2.12, “Uploading and Maintaining Custom Packages”](#) for instructions. Once uploaded, packages may be reassigned through the website, as described in [Section 1.2.3, “Assigning Packages to Software Channels”](#).

Channels are created on the Red Hat Satellite website as follows:

1. Log in to the Red Hat Satellite website as a Channel Administrator.
2. On the top navigation bar, click the **Channels** tab and then click the **Manage Software Channels** button on the left navigation bar.
3. On the **Software Channel Management** page, click **create new software channel** at the top-right corner. Red Hat Satellite Server administrators are presented with the option to **clone channel**. See [Section 1.2.8, “Cloning Software Channels”](#) for instructions.
4. On the **New Channel** page, define the details of the channel following the instructions on the page. For most channel management actions, the **Channel Label** is used to identify the channel, so select a meaningful label. View the details of existing channels for ideas.

The **GPG key URL** must be the location of the key on the server, as defined during the client configuration process. See the *Red Hat Satellite Client Configuration Guide*. The GPG key ID is the unique identifier, such as "DB42A60E", while the GPG key fingerprint is similar to "CA20 8686 2BD6 9DFC 65F6 ECC4 2191 80CD DB42 A60E". Notice that the key ID is the same as the last pair of quartets in the key fingerprint.

5. When finished, click **Create Channel** at the bottom of the page.

## 1.2.3. Assigning Packages to Software Channels

When packages are initially uploaded, they can be assigned to a custom channel, multiple custom channels, or no channel at all. See [Section 1.2.12, “Uploading and Maintaining Custom Packages”](#) for instructions. Once uploaded, packages may be reassigned between custom channels and the No Channels repository.

These functions can be made available by following these steps:

1. Click on the **Channels** tab in the top navigation bar and then **Manage Software Channels** on the left navigation bar.
2. In the **Software Channel Management** page, click the name of the channel to receive packages.
3. In the **Basic Channel Details** page, click the **Packages** tab and then the **Add** subtab. To associate packages with the channel being edited, select the option containing the packages from the **View** dropdown menu and click **View**.



### Note

Packages already associated with the channel being edited are not displayed. Packages not assigned to a specific channel are found in the **Packages in no channels** menu item. Selecting **All managed packages** presents all available packages.

4. Select the checkboxes of the packages to be assigned to the edited channel and click **Add Packages** at the bottom-right corner of the page. A confirmation page appears with the packages listed.
5. Click **Confirm** to associate the packages with the channel. The **List/Remove** subtab of the **Managed Software Channel Details** page then appears with the new packages listed.

Once packages are assigned to a channel, the errata cache is updated to reflect the changes. This update is delayed briefly so that users may finish editing a channel before all of the changes are made available. To initiate changes to the cache manually, click the **commit your changes immediately** link within the text at the top of the **List/Remove** subtab.

## 1.2.4. Managing Channel Management Privileges

In order to perform any channel management tasks, users must have obtained the proper permissions as a *Channel Administrator*. These permissions can be modified through the Red Hat Satellite website. Permissions are assigned to users by *Organization Administrators*, the highest level of administrator. Channel Administrator privileges are assigned as follows:

1. Log in to the Red Hat Satellite website as an Organization Administrator.
2. On the top navigation bar, click the **Users** tab and then click the name of the user who is performing channel management functions.
3. On the **User Details** page, scroll down to the **Roles** section and select the checkbox labeled **Channel Administrator**. Then click **Submit** at the bottom of the page. Note that Organization Administrators are automatically granted channel administration privileges.
4. Have the user log in to the Red Hat Satellite website, click the **Channels** tab on the top navigation bar, and ensure the **Manage Software Channels** button appears on the corresponding left navigation bar.

## 1.2.5. Changing Custom Channel Permissions

Custom channels can be restricted to specific organizations and systems. Situations where this would be useful would include:

- ✦ Restricting channel content to specific organizations and systems for qualifying purposes like testing different software configurations before production
- ✦ Controlled distribution of licensed or third-party packages

### 1.2.5.1. Modifying Custom Channel User Permissions

#### Prerequisites

It is assumed that there is an existing channel that needs permission changes.

1. Log in to the Satellite server as a channel or organization administrator.
2. Click **Channels** → **Manage Software Channels**.
3. Click the channel whose permissions need to be modified.
4. Scroll down to the **Channel Access Control** → **Per-User Subscription Restrictions** and **Only selected users within your organization may subscribe to this channel**.
5. Click **Update Channel** to save the changes.
6. Click the **Subscribers** subtab and select the users that should be able to subscribe to the channel.
7. Click **Update**.

### 1.2.5.2. Modifying Custom Organization Permissions

#### Prerequisites

It is assumed that there is an existing channel that needs permission changes.

1. Log in to the Satellite server as a channel or organization administrator.
2. Click **Channels** → **Manage Software Channels**.
3. Click the channel whose permissions need to be modified.
4. Scroll down to the **Channel Access Control** → **Organization Sharing**. Choose either of the following:
  - ✦ This channel is private and cannot be accessed by any other organization.
  - ✦ This channel is protected and may only be accessed by specific trusted organizations.
  - ✦ This channel is public and may be accessed by any of the trusted organizations trusted by this organization.
5. Click **Update Channel**.
6. (Optional) If you chose protected channel, the Satellite server will ask to confirm the channel sharing modification that was made. Since systems maybe subscribed to the channel that will be removed because of the change in channel permissions. Choose either to:
  - ✦ Click on **Deny Access and Confirm** to unsubscribe any systems previously subscribed from trusted organizations.

- Click on **Grant Access and Confirm** to keep the systems from trusted organizations subscribed.
- Click **Cancel** if you wish to review the systems and trusted organizations before taking action.

### 1.2.6. Manage Software Channels

In addition to the buttons and pages available to standard Red Hat Satellite users a user with Organization or Channel Administrator roles can click on **Channels** to allow Red Hat Satellite Server and Red Hat Satellite Proxy Server customers to access **Manage Software Channels** on the left navigation bar. This button opens the Software Channel Management page, where all custom software channel management work occurs.



#### Warning

When using both Red Hat Satellite Proxy Server and Red Hat Satellite Server, manage custom channels and packages *only* on the Satellite, since the Proxy servers receive updates directly from it. Manually managing packages and channels on a Proxy in this combined configuration risks putting the servers out-of-sync.

The different subtabs within the Software Channel Management list takes you to different tabs of the Basic Channel Details page.

### 1.2.7. Basic Channel Details

Virtually all custom channel management tasks are carried out within the **Basic Channel Details** page, accessed by clicking **Manage Software Channels** on the left navigation bar and then selecting the name of channel to be altered. This page offers several tabs:

- **Details** - Provides basic information about the channel, such as its parent channel, name, summary, and description. Some of this information is modifiable. In addition, a **Per-User Subscription Restrictions** combobox can be seen by Organization Administrators and Channel Administrators. This signifies the default behavior of every channel allowing any user to subscribe systems to it. Unchecking this box and clicking **Update Channel** causes the appearance of a **Subscribers** tab, which is used to grant certain users subscription permissions to the channel.
- **Organizations** - Provides a list of organizations in which the channel has granted access to view and consume content in the channel. These organizations are listed because of the organizational trust your Organization has with them. Access to this channel by organizations can be modified here. Select the checkbox and click on **Modify Access** to remove an organization's access. Note that Organization Administrators and Channel Administrators automatically have subscription access to all channels.
- **Managers** - Lists users who have management permissions to the custom channel. This tab appears for Organization Administrators and Channel Administrators. Select the checkboxes of the users to be allowed full administration of this channel and click **Update**. This status does not enable the user to create new channels. Note that Organization Administrators and Channel Administrators automatically have management access to all channels.
- **Errata** - Provides the errata associated with each of your custom channels. Just as Red Hat Network produces and delivers errata updates to Red Hat Enterprise Linux software, you deliver errata updates to your custom channels as part of updating your servers with the latest code. This tab contains subtabs that allow you to view, add, remove, and clone erratum: **List/Remove**, **Add** and **Clone**. Note that cloning errata can be done only via Red Hat Satellite Server.

- **List/Remove** - Displays all of the errata currently associated with the custom channel and provides a means to cancel that association. To remove errata from the channel, select their checkboxes and click **Remove Errata** on the bottom right-hand corner of the page. A confirmation page appears listing the errata to be removed. Click **Confirm** to complete the action.
- **Add** - Enables the addition of errata to the channel. All of the errata potentially applicable to the channel are listed. To add errata to the channel, select the appropriate checkboxes and click **Add Errata**. See [Chapter 5, Errata Management](#) for a discussion of errata management.
- **Clone** - Allows Satellite customers to replicate errata and associated packages for a cloned channel. This subtab immediately appears populated for channels that were cloned with either the original state or select errata option. The **Clone** tab also gains errata whenever one is issued for the target (that is, originating) channel. This makes it useful for channels cloned with the current state option, as well. See [Section 1.2.8, "Cloning Software Channels"](#) for a discussion of cloning options.

To include errata from the target channel in the cloned channel, select either **Merge** or **Clone** from each advisory's dropdown menu. The **Merge** option exists only if the erratum has been previously cloned. Use it to associate the erratum across channels and avoid duplicate entries. Use the **Clone** option to create a new entry, such as when modifying it from the previous clone.

By default, cloned errata inherit the label of the original Red Hat advisory with the "RH?" prefix replaced with "CL". For example, RHSA-2003:324 becomes CLA-2003:324. Subsequent clones of the same advisory have their second letters sequenced to denote their order, such as "CM" and "CN". These labels can be altered through the Errata Management page. See [Chapter 5, Errata Management](#) for instructions.

In addition to the **Merge** option, previously cloned errata contain values within the **Owned Errata** column. The erratum label is linked to its details page. The **pub** and **mod** flags within parentheses identify whether the cloned erratum has been published or modified from the original advisory. A plus sign + before the flag indicates affirmative, the cloned errata has been published. A minus sign - before the flag denotes negative. For example, **(-mod)** may mean a package has been deleted. To find out more about publishing and editing custom errata, See [Chapter 5, Errata Management](#).

To exclude errata from the cloned channel, select **Do Nothing** from the dropdown menus. When satisfied with the changes, click **Clone Errata**. Review the impending changes on the confirmation page and click **Update Errata**.

- **Sync** - Displays the errata packages that were not included in the initial channel cloning but have since been updated. This page is where cloned channels can be synchronized with current errata by marking the desired checkbox and clicking **Sync Errata**.
- ✱ **Packages** - Provides the packages associated with each of the custom channels. This tab contains the following subtabs that allow organizations to view, add, and remove packages: **List/Remove**, **Add**, and **Compare**.
  - **List/Remove** - Displays all of the packages currently associated with the custom channel and provides a means to cancel that association. To remove packages from the channel, select their checkboxes and click **Remove Packages** on the bottom right-hand corner of the page. A confirmation page appears with the packages to be removed listed. Click **Confirm** to complete the action.





### Important

The list displayed on this page differs from the standard package list available in the **Software Channel Details** page. It displays all versions of a package remaining in the database rather than just the latest. It is possible to revert to a previous version of a package simply by removing the latest version.

- **Add** - Enables the addition of packages to the channel. To see available packages, select an option from the **View** dropdown menu and click **View**. To add packages to the channel, select the appropriate checkboxes and click **Add Packages**. See [Section 1.2.3, “Assigning Packages to Software Channels”](#) for more information about this process.
- **Compare** - Enables the comparison of package lists between different channels. To see the differences, select another channel from the **Compare to:** dropdown menu and click **Compare**. A list appears showing all packages not contained by both channels and indicating the existing channel location of each.
- ✦ **Repositories** - Select **Manage Repositories** to assign **yum** repositories to the channel and synchronize repository content.
  - **Add / Remove** — Lists configured repositories. Repositories can be added and removed by selecting the checkbox next to the repository name and clicking **Update Repositories**.
  - **Sync** — Lists configured repositories. The synchronization schedule can be set using the dropdown boxes, or an immediate synchronization can be performed by clicking **Sync Now**.

## 1.2.8. Cloning Software Channels

Red Hat Satellite Server Channel Administrators also have the ability to clone software channels for easy package association. Cloning offers a complete replica of another channel, enabling immediate association of appropriate packages and errata with a custom software channel.

To access this functionality:

1. Click the **Channels** tab on the top navigation bar then the **Manage Software Channels** on the left navigation bar. This takes you to the **Software Channel Management** page.
2. Click **clone channel** at the top-right corner to begin cloning.

Three cloning options are presented: current state of the channel, original state of the channel, or select errata. These options are described fully on the webpage itself but are summarized as:

- ✦ **Current state of the channel** - All of the errata and all of the latest packages now in the target channel.
  - ✦ **Original state of the channel** - All of the original packages from the target channel but none of the errata or associated update packages.
  - ✦ **Select Errata** - All of the original packages from the target channel with the ability to exclude certain errata and associated update packages.
3. Select the option desired using the radio buttons within the **Clone** field, identify the target channel using the **Clone From** dropdown menu, and click **Create Channel**.



4. On the **New Software Channel** page, complete the fields as described in [Section 1.2.2, “Creating a Software Channel”](#). The default values will suffice.
5. Click **Create Channel**. If either the original or current option is selected, the **Details** tab of **Managed Software Channel Details** page will appear. Alter the settings for the new channel. See [Section 1.2.7, “Basic Channel Details”](#) for instructions.

If the select errata option to clone the channel is selected, it will redirect to the **Clone** subtab of **Managed Software Channel Details** page instead. Errata and associated packages for cloning may have to be individually selected for cloning and inclusion in the new channel. See [Section 1.2.7, “Basic Channel Details”](#) for specific instructions.



### Note

There is a command that clones all errata, within a given channels based on date to ensure a consistent reproducible package sets. This command is called **spacewalk-clone-by-date**.

## 1.2.9. Creating Custom Channels From Specific Update Levels

The following situations may require custom channels with specific update levels:

- A controlled environment with systems that require only minor release updates instead of the latest updates
- A test environment with a specific package set
- Systems with applications that require specific versions to function

### Prerequisites

To implement the solution below, the Satellite Server must be subscribed to the Red Hat Network Tools channel and the **spacewalk-remote-utils** must be installed on the Satellite Server. The package is included in the Red Hat Network Tools channel.

1. Log in as root to the Satellite server.
2. Create the custom channel from a specific update level on Red Hat Satellite:

```
# spacewalk-create-channel --user=admin --server=localhost --version=6
--update=GOLD --release=Server --arch=x86_64 --destChannel=gold-rhel6
You have not specified a source channel, we will try to determine it
from inputs
Trying with source channel: rhel-x86_64-server-6
Creating channel, gold-rhel6, with arch x86_64 2797 packages in
source file to push.
Pushing 2797 packages, please wait.
Successfully pushed 2797 packages out of 2797

# spacewalk-create-channel -l admin -s localhost -d update1-rhel6 -D
/usr/share/rhn/channel-data/6-u1-server-x86_64
Password:
You have not specified a source channel, we will try to determine it
from inputs
Trying with source channel: rhel-x86_64-server-6
Creating channel, update1-rhel6, with arch x86_64 2857 packages in
```

```
source file to push.
Pushing 2857 packages, please wait.
Successfully pushed 2857 packages out of 2857
```

Where:

- ✱ `-lUSER, --user=USER` - The username to connect to the server.
- ✱ `-sSERVER, --server=SERVER` - The hostname or IP address of the Satellite or Spacewalk server to connect to. Defaults to localhost.
- ✱ `-vVERSION, --version=VERSION` - The version of the channel to create (e.g. 6, 5, 4).
- ✱ `-rRELEASE, --release=RELEASE` - The release of the channel to create (e.g. AS, ES, WS, Server, Client, Desktop).
- ✱ `-uUPDATE_LEVEL, --update=UPDATE_LEVEL` - The update level of a channel to create (e.g. GOLD, U1, U2, U3, U4, U5, U6, U7, U8, U9), where GOLD stands for the initial release.
- ✱ `-aARCH, --arch=ARCH` - The arch of the channel to create (e.g. i386, ia64, ppc, s390, s390x, x86\_64).
- ✱ `-dDEST_CHANNEL, --destChannel=DEST_CHANNEL` - The label of the destination channel. This will be created if not present.
- ✱ `-DDATAFILE, --data=DATAFILE` - Path to a list of rpms to move to the destination channel, only used if version, release, update, and arch are not specified.(Optional)



### Note

Only applicable to Red Hat Satellite 5.3 and newer.

## 1.2.10. Removing Software Packages

In addition to adding and removing packages within channels, there is also the option of deleting packages entirely from both the database and file system. Removal from the file system is delayed by about one hour. This can be done through the **Package Management** page, accessed by clicking **Manage Software Packages** on the left navigation bar.



### Warning

Although deleting packages from the database can be undone by uploading them again, the packages lose their association with any errata. Upon reloading, they must be re-associated with errata manually. See [Chapter 5, Errata Management](#) for instructions.

To remove packages from the database:

1. Go to the **Package Management** page and select an option containing the packages from the dropdown menu and click **View Packages**.
2. Select the appropriate checkboxes and click **Confirm Deletion**. A confirmation page appears with the packages listed. Click **Delete Package(s)** to delete the packages entirely.

**Note**

If using the Red Hat Network Package Manager to upload packages to a Red Hat Satellite Proxy Server, the actual packages are stored on the Proxy Server. Custom packages cannot be downloaded through the Red Hat Satellite, although they are listed. To remove the packages from the local disk you will need to login to the Satellite Proxy Server. See the *Red Hat Satellite Proxy Installation Guide* for details on where these packages are stored.

**1.2.11. Deleting Software Channels**

Red Hat Satellite Server and Red Hat Satellite Proxy Server administrators have the ability to remove unused channels. This action is conducted within the **Channels** → **Manage Software Channels** page. Click **delete software channel** at the top-right corner of the page to remove the channel. On the following page, click **Delete Channel** to finish the action.

**Note**

The **Channels** → **Manage Software Channels** page is described in detail in [Section 1.2.7, “Basic Channel Details”](#).

**Important**

Packages do not get deleted together with the channel. To delete packages from Red Hat Satellite, please See [Section 1.2.10, “Removing Software Packages”](#).

The following factors should be taken into consideration before removing a channel via the website:

- ✦ Packages from the channel will remain on the server even if the channel is removed. There is an option to delete them after.
- ✦ Any errata related to the channel may be orphaned after the channel deletion.
- ✦ The Satellite server will not delete a parent channel if a child channel exists. Delete all child channels before deleting the parent.
- ✦ Kickstart distributions must be dissociated from the channel or deleted before deleting the channel.
- ✦ If the established channel on the Proxy is connected to a Satellite, delete the channel on the Red Hat Satellite Proxy Server.

**1.2.12. Uploading and Maintaining Custom Packages**

Depending upon which Red Hat Network service is used, there are two different mechanisms for uploading packages to private channels.

Customers of Red Hat Satellite Proxy Server use the **Red Hat Network Package Manager** application, which sends package header information to the central Satellite Server and places the package itself into the local repository of the Proxy that invokes **Red Hat Network Package Manager**.

Customers of Red Hat Satellite Server use the **Red Hat Network Push** application, which sends package header information to the local Red Hat Satellite Server and places the package into the local repository of the Satellite that invoked **Red Hat Network Push**.

This section discusses both of these tools in detail.



### Warning

If you use both Red Hat Satellite Proxy Server and Red Hat Satellite Server, use only **Red Hat Network Push**. The Proxy-Satellite combination requires custom packages and channels to be uploaded to the Satellite only. From there, the Proxy servers obtain the packages and distribute them to client systems.

#### 1.2.12.1. Uploading Packages to Red Hat Satellite Proxy Server

**Red Hat Network Package Manager** allows an organization to serve custom packages associated with a Satellite channel privately through the Red Hat Satellite Proxy Server. Client systems that communicate to the Satellite via the Proxy will be able to download the package if they are subscribed to the channel. Systems that are not communicating to the Satellite via the Proxy, if subscribed to the channel, will only receive yum package meta-data and an error when trying to retrieve the package as the Satellite does not have a copy of the package on its local repository. If the organization wants the Red Hat Satellite Proxy Server to serve only official Red Hat Enterprise Linux packages and Organization owned packages do not install the **Red Hat Network Package Manager**.

To use the **Red Hat Network Package Manager**, install the **spacewalk-proxy-package-manager** RPM package and its dependencies. This package is available to registered Red Hat Satellite Proxy Server systems and is installed by running **yum install spacewalk-proxy-package-manager**.



### Note

Only the header information for the packages is uploaded to the Red Hat Satellite Server. The headers are required so that Red Hat Network can resolve package dependencies for the client systems. The actual package files (**\*.rpm**) are stored on the Red Hat Satellite Proxy Server. For this reason, custom packages cannot be downloaded through the Red Hat Satellite website, although they are listed. They must be retrieved by the client system using **yum install**.

##### 1.2.12.1.1. Configuring and Using the Red Hat Network Package Manager

Before using Red Hat Network Package Manager to upload packages into Red Hat Satellite, manually copy the packages to the Proxy server itself. For example, from a development host, use **scp**:

```
# scp foo.rpm root@rhnprefix.example.com:/tmp
```

When using Red Hat Network Package Manager to upload the packages into Red Hat Satellite, point to the files previously copied to the server.



## Note

Create at least one private channel to receive custom packages prior to upload into Red Hat Satellite, since a channel is required for systems to obtain the packages.

Run the following command on your Satellite Proxy Server to upload the package headers to the Red Hat Satellite Server and copy the packages to the Satellite Proxy Server repository:

```
# rhn_package_manager -c label_of_private_channel pkg-list
```

The *label\_of\_private\_channel* is the custom channel created to receive these packages. Be sure to use the precise channel label specified during its creation. If one or more channels are specified (using **-c** or **--channel**), the uploaded package headers are linked to all the channels identified. If a channel is not specified, the packages are deposited in the **No Channels** section of the **Package Management** page. See [Section 1.2.3, “Assigning Packages to Software Channels”](#) for instructions on reassigning packages.

The *pkg-list* reference represents the list of packages to be uploaded. These packages must already be physically copied to the Proxy host. Alternatively, use the **-d** option to specify the local directory that contains the packages to be added to the channel. **Red Hat Network Package Manager** can also read the list of packages from standard input (using **--stdin**).

Other options are specified in a configuration file, such as the Red Hat Satellite Server URL, the HTTP proxy username and password (if the HTTP proxy requires authentication), and the top directory where packages live. This special configuration must *not be edited* and is located at **/etc/rhn/default/rhn\_proxy\_package\_manager.conf**. The choices can be overridden in the default configuration file with settings in the main configuration file **/etc/rhn/rhn.conf** or via command line options passed to Red Hat Network Package Manager.

Parameters not set in this file are read from **.rhn\_package\_manager** in the home directory of the user currently logged in and finally from **/etc/rhn/rhn\_package\_manager.conf**. Make sure all of these files have the appropriate permissions to prevent others from reading them.

After uploading the packages, check to see if the local directory is in sync with the Red Hat Satellite Server's image of the channels:

```
# rhn_package_manager -s -c name_of_private_channel
```

This **-s** option lists all the missing packages, which are packages uploaded to the Red Hat Satellite Server but not present in the local directory. You must be an Organization Administrator to use this option. The application prompts you for your Red Hat Satellite username and password.

The **--copyonly** option copies the file listed in the argument into the specified channel without uploading to the Satellite. This is useful when a channel on a Red Hat Satellite Proxy Server is missing a package and you don't want to reimport all of the packages in the channel.

```
# rhn_package_manager -c channel-name --copyonly /path/to/missing/file
```

Use **Red Hat Network Package Manager** to retrieve a list of packages in a channel, as they are stored by the Red Hat Satellite Server:

```
# rhn_package_manager -l -c name_of_private_channel
```

The **-l** option lists the package name, version number, release number, architecture, and channel name for each package in the specified channel(s). See [Section 1.2.12.1.1, “Configuring and Using the Red Hat Network Package Manager”](#) for additional options.

[Section 1.2.12.1.1, “Configuring and Using the Red Hat Network Package Manager”](#) is a summary of all the command line options for **Red Hat Network Package Manager (rhn\_package\_manager)**:

**Table 1.1. rhn\_package\_manager options**

Option	Description
<b>-v, --verbose</b>	Increase verbosity of standard output messages.
<b>-d, --dir <i>DIRECTORY_NAME</i></b>	Process packages from this directory.
<b>-c, --channel <i>CHANNEL_NAME</i></b>	Specify the channel to receive packages. Multiple channels may be specified using multiple instances of <b>-c</b> (e.g.: <b>-c channel_one -c channel_two</b> )
<b>-n, --count <i>NUMBER</i></b>	Process this number of headers per call - the default is 32.
<b>-l, --list</b>	List the packages in the specified channel(s).
<b>-s, --sync</b>	Check if local directory is in sync with the server.
<b>-p, --printconf</b>	Print the current configuration and exit.
<b>--newest</b>	Push only the packages that are newer than those on the server. Note that source packages are special in that their versions are never compared to each other. Their newness is dependent on their associated binary packages. Using this option with Red Hat Network Package Manager and just a source package does upload the package, but the source package does not appear in the Red Hat Satellite Web interface until the associated binary package has been uploaded. Contrast this with <b>--source</b> . Using <b>--source --newest</b> together <i>does</i> update the stand-alone source package with newer packages and does not require an associated binary package to be uploaded first.
<b>--source</b>	Upload the indicated source packages. Doing this treats them as plain, stand-alone packages and <i>not</i> as special source packages associated with another, pre-existing binary package. For example, use this when application sources need to be distributed to developers and testers outside of regular source control management.
<b>--stdin</b>	Read the package names from standard input.
<b>--nosig</b>	Don't fail if packages are unsigned.
<b>--no-ssl</b>	Turn off SSL (not recommended).
<b>--stdin</b>	Read the package names from standard input.
<b>--username <i>USERNAME</i></b>	Specify the Red Hat Satellite username. If the username is not provided, you will be prompted for the username of a valid Channel Administrator.
<b>--password <i>PASSWORD</i></b>	Specify the Red Hat Satellite password. If the password is not provided, you will be prompted for the password of a valid Channel Administrator.
<b>--dontcopy</b>	In the post-upload step, do not copy the packages to their final location in the package tree.
<b>--copyonly</b>	Only copy the packages, do not re-import them.
<b>--test</b>	Only print a list of the packages to be pushed.
<b>-, --help</b>	Display the help screen with a list of options.

Option	Description
- <b>-usage</b>	Briefly describe the available options.
- <b>-copyonly</b>	Only copy packages



### Note

These command line options are also described in the **rhn\_package\_manager** manual page: **man rhn\_package\_manager**.

## 1.2.12.2. Uploading Packages to Red Hat Satellite Server

The **Red Hat Network Push** application allows organizations to serve custom packages associated with a private Red Hat Network channel through the Red Hat Satellite Server. If the Red Hat Satellite Server is only going to serve official Red Hat Enterprise Linux packages, there is no need to install **Red Hat Network Push**.

To use **Red Hat Network Push**, install the **rhnpush** package and its dependencies. This package is available to registered Red Hat Satellite Server systems and is installed by running **yum install rhnpush**.

**Red Hat Network Push** uploads RPM header information to the Red Hat Satellite Server database and places the RPM in the Red Hat Satellite Server package repository. Unlike the Red Hat Satellite Proxy Server's **Red Hat Network Package Manager**, **Red Hat Network Push** never distributes package information, even the headers, outside of the Red Hat Satellite Server database.



### Note

If the Satellite installation is enabled to support Solaris OS systems, it is possible to use Red Hat Network Push from a Solaris client to upload Solaris package content to custom Solaris channels.

### 1.2.12.2.1. Configuring the Red Hat Network Push Application

When **Red Hat Network Push** is installed, a central configuration file is installed in **/etc/sysconfig/rhn/rhnpushrc**. This file contains values for all the options contained in [Section 1.2.12.2.1, “Configuring the Red Hat Network Push Application”](#).

These distinct configuration files are useful in varying settings depending on the directory from which the **rhnpush** command is issued. Settings in the current directory (**./rhnpushrc**) take precedent over those in the user's home directory (**~/.rhnpushrc**), which are used before those in the central configuration file (**/etc/sysconfig/rhn/rhnpushrc**).

For instance, the current directory configuration file can be used to specify:

- ✧ The software channel to be populated
- ✧ The home directory configuration file to include the username to be invoked
- ✧ The central configuration file to identify the server to receive the packages

[Section 1.2.12.2.1, “Configuring the Red Hat Network Push Application”](#) contains all command line options for the **rhnpush** command:



Table 1.2. `rhnpush` options

Option	Description
<code>-v --verbose</code>	Increase verbosity, option can be used multiple times, that is, <code>-vv</code> , <code>-vvv</code> , and so forth.
<code>-d, --dir DIRECTORY</code>	Process packages from this directory.
<code>-c, --channel=CHANNEL_LABEL</code>	Specify the channel to receive packages. Note that this is required and is not the same as the channel's name. Multiple channels may be specified using multiple instances of <code>-c</code> (e.g. <code>-c CHANNEL_ONE -c CHANNEL_TWO</code> ).
<code>-n, --count N_HEADERS_PER_CALL</code>	Process this number of headers per call. Must be an integer. The default number is 25.
<code>-l, --list</code>	List only the specified channels.
<code>-r, --reldirRELATIVE_DIRECTORY</code>	Associate this relative directory with each file.
<code>-o, --orgidORGANIZATION_ID</code>	Include the organization's ID number. Must be an integer.
<code>-u, --username USERNAME</code>	Include the Red Hat Satellite username of the user that has administrative access to the specified channel. If not provided, <b>rhnpush</b> prompts for the username of a valid Channel Administrator. The username and password are cached in <code>~/ .rhnpushcache</code> for a limited time, five minutes being the default. Use <code>--new-cache</code> to force a new username and password.
<code>-p, --password PASSWORD</code>	Include the Red Hat Satellite password of user that has administrative access to the specified channel. If not provided, <b>rhnpush</b> prompts for the password of a valid Channel Administrator. The username and password are cached in <code>~/ .rhnpushcache</code> for a limited time, five minutes being the default. Use <code>--new-cache</code> to force a new username and password.
<code>-s, --stdin</code>	Read package list from standard input, for example from a piped <code>ls</code> command.
<code>-X, --exclude GLOB</code>	Exclude packages that match this glob expression.
<code>--force</code>	Force upload of a package, even if a package of that name and version currently exists in the channel. Without this option, uploading a pre-existing package returns an error.
<code>--nosig</code>	Don't fail if packages are unsigned.
<code>--new-cache</code>	Forces Red Hat Network Push to drop the username and password cache, then accept or ask for new ones. This is useful if mistakes are entered the first time.
<code>--newest</code>	Push only the packages that are newer than those on the server. Note that source packages are special in that their versions are never compared to each other. Their newness is dependent on their associated binary packages. Using this option with Red Hat Network Push and just a source package does upload the package, but the source package does not appear in the Red Hat Satellite Web interface until the associated binary package has been uploaded. Contrast this with <code>--source</code> . Using <code>--source --newest</code> together <i>does</i> update the stand-alone source package with newer packages and does not require an associated binary package to be uploaded first.
<code>--header</code>	Upload only the headers.



Option	Description
<b>--source</b>	Upload the indicated source packages. Doing this treats them as plain, stand-alone packages and <i>not</i> as special source packages associated with another, pre-existing binary package. For example, use this when you want to distribute application source to developers and testers outside of regular source control management.
<b>--server SERVER</b>	Specify the server to which packages are uploaded. Currently, a value of <b>http://localhost/APP</b> is necessary. This parameter is required.
<b>--test</b>	This only prints a list of the packages to be pushed, it doesn't actually push them.
<b>-h, --help</b>	Briefly describe the options.
<b>-, --usage</b>	View the usage summary.



### Note

These command line options are also described in the **rhnpush** manual page: **man rhnpush**.

#### 1.2.12.2.2. Using the Red Hat Network Push application



### Note

It is recommended to create at least one private channel to receive custom packages prior to upload, since a channel is required for systems to obtain the packages.

The following command uploads package headers to the Red Hat Satellite Server and copies the packages to the Red Hat Satellite Server package repository:

```
# rhnpush -c label_of_private_channel pkg-list
```

The **Red Hat Network Push** configuration file settings can be overridden by specifying options and values on the command line:

```
# rhnpush -c label_of_private_channel --server=localhost pkg-list
```

The *label\_of\_private\_channel* is the custom channel created to receive these packages. Be sure to use the precise channel label specified during its creation. If one or more channels are specified (using **-c** or **--channel**), the uploaded package headers are linked to all the channels identified. If there is no channel specified, the packages are deposited in the **No Channels** section of the **Package Management** page. See [Section 1.2.3, “Assigning Packages to Software Channels”](#) for instructions on reassigning packages.

The **--server** option specifies the server to which the packages are installed, and is required. **Red Hat Network Push** can be installed on external systems, but running **Red Hat Network Push** locally on the Red Hat Satellite Server is recommended.

The *pkg-list* reference represents the list of packages to be uploaded. Alternatively, use the **-d** option to specify the local directory that contains the packages to be added to the channel. **Red Hat Network Push** can also read the list of packages from standard input (using **--stdin**).

## 1.3. Managing Configuration Channels

Red Hat Satellite has features to manage the organization's configuration files. This includes files that are managed centrally in configuration channels and files that are managed locally via individual system profiles. Only a Configuration Administrator or a Satellite Administrator can see these files.

This allows organizations to deploy similar configurations to a group of systems and manage configuration changes centrally.

Centrally-managed files are those that are available to multiple systems; changes to a single file in a central configuration channel can affect many systems. In addition, there are local configuration channels. Each system with a Provisioning entitlement has a local configuration channel (also referred to as an override channel) and a Sandbox channel. Local configuration channels are not created within Red Hat Satellite; local configuration channels automatically exist for each system to which a Provisioning entitlement has been applied.

### 1.3.1. Preparing Systems for Config Management

For a system to have its configuration managed through Satellite, it must have the appropriate tools and the **config-enable** file installed. These tools may already be installed on the system, especially if the system was kickstarted with configuration management functionality. If not, they can be found within the Red Hat Network Tools child channel. The following packages should be downloaded and installed:

- ✧ **rhncfg** - The base libraries and functions needed by all **rhncfg-\*** packages.
- ✧ **rhncfg-actions** - The code required to run configuration actions scheduled via the Red Hat Network website.
- ✧ **rhncfg-client** - A command line interface to the client features of the Red Hat Network Configuration Management system.
- ✧ **rhncfg-management** - A command line interface used to manage Red Hat Network configuration.

### 1.3.2. Creating a New Configuration Channel

A configuration channel has to be created to house the configuration files that will be deployed to specific systems. To create a new configuration channel:

1. Log in as a Channel Administrator or an Organization Administrator.
2. Click the **Configuration** tab.
3. On the right-hand frame marked as *Configuration Actions*, click **Create a New Configuration Channel**.
4. Fill in the following fields:
  - ✧ **Name**
  - ✧ **Label**. This field must contain only alphanumeric characters, "-", "\_", and ".".
  - ✧ **Description**. You must enter a description. This field can contain any brief information that allows you to distinguish this channel from others.
5. Click **Create Config Channel**.

### 1.3.3. Adding Configuration Files to the Configuration Channel

1. Log in as a Channel Administrator or an Organization Administrator.
2. Click the **Configuration** tab.
3. On the submenu on the left, click on **Configuration Channels**.
4. Select the configuration channel the configuration files will be added to.
5. Click on the subtab **Add Files**.
6. Fill in the required fields:
  - ✦ *File to Upload* - the maximum allowed file size for configuration files is 128kb.
  - ✦ *Filename/Path* - the path in the target system the configuration file should be deployed to.
  - ✦ *Ownership* - the user and group that owns the file. If the user and group added in the fields do not exist on the systems where the file is being deployed to, the deployment will fail.
  - ✦ *File Permissions Mode* - permissions on the file based on who can modify it. For example, '644' for text files and '755' for directories and executables will allow global access or execution (but not modification).
  - ✦ *SELinux context* - enter SELinux context like: user\_u:role\_r:type\_t:s0-s15:c0.c1024.
  - ✦ *Macro Delimiters* - a listing of available macros are in the next section, [Section 1.3.4, "Including Macros in Configuration Files"](#)

### 1.3.4. Including Macros in Configuration Files

In traditional file management, you would be required to upload and distribute each file separately, even if the distinction is nominal and the number of variations is in the hundreds or thousands. Red Hat Satellite addresses this by allowing the inclusion of macros, or variables, within the configuration files it manages for Provisioning-entitled systems. In addition to variables for custom system information, the following standard macros are supported:

- ✦ rhn.system.sid
- ✦ rhn.system.profile\_name
- ✦ rhn.system.description
- ✦ rhn.system.hostname
- ✦ rhn.system.ip\_address
- ✦ rhn.system.custom\_info(key\_name)
- ✦ rhn.system.net\_interface.ip\_address(eth\_device)
- ✦ rhn.system.net\_interface.netmask(eth\_device)
- ✦ rhn.system.net\_interface.broadcast(eth\_device)
- ✦ rhn.system.net\_interface.hardware\_address(eth\_device)
- ✦ rhn.system.net\_interface.driver\_module(eth\_device)

To use this feature, either upload or create a configuration file through the **Configuration Channel Details** page. Then, open its **Configuration File Details** page and include the supported macros of your choosing. Ensure that the delimiters used to offset your variables match those set in the **Macro Start**

**Delimiter** and **Macro End Delimiter** fields and do not conflict with other characters in the file. We recommend that the delimiters be two characters in length and must not contain the percent (%) symbol.

As an example, you may have a file applicable to all of your servers that differs only in IP address and hostname. Rather than manage a separate configuration file for each server, you may create a single file, such as **server.conf**, with the IP address and hostname macros included, like so:

```
hostname={| rhn.system.hostname |}
ip_address={| rhn.system.net_interface.ip_address(eth0) |}
```

Upon delivery of the file to individual systems, whether through a scheduled action in the Red Hat Network website or at the command line with the **Red Hat Network Configuration Client (rhncfg-client)**, the variables will be replaced with the hostname and IP address of the system, as recorded in the Satellite's System Profile. In the above configuration file, for example, the deployed version resembles the following:

```
hostname=test.example.domain.com
ip_address=177.18.54.7
```

To capture custom system information, insert the key label into the custom information macro (rhn.system.custom\_info). For instance, if you developed a key labeled "asset" you can add it to the custom information macro in a configuration file to have the value substituted on any system containing it. The macro would look like this:

```
asset={@ rhn.system.custom_info(asset) @}
```

Upon deployment of the file to a system containing a value for that key, the macro gets translated, resulting in a string similar to the following:

```
asset=Example#456
```

To include a default value, for instance if one is required to prevent errors, you can append it to the custom information macro, like so:

```
asset={@ rhn.system.custom_info(asset) = 'Asset #' @}
```

This default is overridden by the value on any system containing it.

Using the **Red Hat Network Configuration Manager (rhncfg-manager)** will not translate or alter the files, as that tool is system agnostic. **rhncfg-manager** does not depend on system settings.



## Note

This feature only inserts macros into text files. Binary files cannot be interpolated.

### 1.3.5. Subscribing Systems to the Configuration Channel

In order for Red Hat Satellite to centrally manage configuration channels through the configuration channels, two requirements must be fulfilled:

- » Systems must be subscribed to the configuration channel.

- ✱ Configuration Management must be enabled on the systems. See [Section 1.3.6, “Enabling Configuration Management on Systems”](#) for the procedure.

To subscribe a system to the configuration channel:

1. Click **Configuration**. On the left-hand submenu, click **Configuration Channels**.
2. Select the configuration channel where the systems should be added.
3. On the channel page, click on the **Systems** → **Target Systems** subtab.
4. Select systems to be added to the configuration channel and click **Subscribe Systems**.

### 1.3.6. Enabling Configuration Management on Systems

The following requirements need to be met before configuration management can be enabled on systems:

- ✱ Target systems need a provisioning entitlement. See the *Systems* chapter for the procedure on how to add a provisioning entitlement to a system.
  - ✱ A subscription to the Red Hat Satellite Tools channel. See the *Systems* chapter for the procedure on how to change a child channel.
1. Log in as Channel Administrator or Satellite Administrator.
  2. Click on **Configuration**.
  3. On the right-hand frame marked as *Configuration Actions*, click **Enable Configuration Management on Systems**.
  4. Select systems to enable.
  5. Schedule the package installation of the **rhncfg-\*** packages. Select a time for these configuration packages to be installed.
  6. Click **Enable Red Hat Satellite Configuration Management**.
  7. Open a terminal console on the individual systems or remotely login as root. The following actions need to be performed:
    - a. Run this command to complete the pending **rhncfg-\*** package installation:

```
# rhn_check
```

- b. Run the following command to enable Red Hat Network actions:

```
# rhn-actions-control --enable-all
```

## Chapter 2. Organizations

Red Hat Satellite enables administrators to divide their deployments into organized containers. These containers (or *organizations*) assist in maintaining clear separation of purpose and ownership of systems and the content deployed to those systems.

Red Hat Satellite supports the creation and management of multiple *organizations* within one installation, allowing for the division of systems, content, and subscriptions across different groups. This chapter summarizes the basic concepts and tasks for multiple organization creation and management.

The **Organizations** Web interface allows administrators to view, create, and manage multiple Satellite organizations. Satellite administrators can allocate software and system entitlements across various organizations, as well as control an organization's access to system management tasks.

Satellite Administrators can create new organizations and assign administrators and entitlements for those organizations. Organization Administrators can assign groups, systems, and users for their organization. This division allows organizations to perform administrative tasks on their own without affecting other organizations.

The **Organizations** page contains a listing of organizations across the Satellite, with both User and System counts assigned to each organization. The **Organizations** page also features a **Trusts** page for any organizational trusts established.

### 2.1. Creating an Organization

1. To create a new organization, click **Admin** → **Organizations** → **Create New Organization**.
2. Type the organization name into the appropriate text box. The name should be between 3 and 128 characters.
3. Create an administrator for the organization, by providing the following information:
  - ✳ Enter a **Desired Login** for the organization administrator, which should be between 5 and 64 characters long. Consider creating a descriptive login name for the Organization Administrator account that matches administrative login names with the organization.
  - ✳ Create a **Desired Password** and **Confirm** the password.
  - ✳ Type in the **Email** address for the organization administrator.
  - ✳ Enter the **First Name** and **Last Name** of the organization administrator.
4. Click the **Create Organization** button to complete the process.

Once the new organization is created, the **Organizations** page will display with the new organization listed.

Satellite Administrators should consider reserving the *organization 1* Organization Administrator account for themselves. This will give them the ability to log in to the organization if required.



## Important

If Red Hat Satellite is configured for PAM authentication, avoid using PAM accounts for the main organization administrator account in new organizations. Instead, create a Satellite-local account for organization administrators and reserve PAM-authenticated accounts for Satellite logins with less elevated privileges. This will discourage users from logging in to the Red Hat Satellite with elevated privileges, as the potential for making mistakes is higher using these accounts.

## 2.2. Managing Organization Entitlements

Once you have created a new organization, it is important to assign entitlements for it. The following entitlements are necessary:

- ✦ *System Entitlement - Management* entitlement is a base requirement for all organizations to function correctly. The number of management entitlements allocated to an organization is equivalent to the maximum number of systems that can register to that organization on the Red Hat Satellite, regardless of the number of software entitlements available. For example, if there are 100 Red Hat Enterprise Linux Client entitlements available in total, but only 50 management system entitlements are available to the organization, only 50 systems are able to register to that organization. *Provisioning* is also recommended especially for systems managed in organizations.
- ✦ *Software Channel Entitlements* - for systems that use Red Hat channels, **Red Hat Enterprise Linux Server** would be required. The **Red Hat Network Tools** channel would also be recommended. The Red Hat Network Tools channel contains various client software required for extended Red Hat Satellite functionality, such as clients necessary for configuration management and kickstart support as well as the **rhn-virtualization** package, which is necessary for the entitlements of Xen and KVM virtual guests to be counted correctly.

### 2.2.1. Changing an Organization's System Entitlement

To manage system entitlements within the Organization:

1. Click the **Admin** menu, and select **Organization**.
2. Choose an organization from the list and select the **Subscriptions** tab. The page defaults to **System Entitlements**.
3. Change the **Proposed Totals** of system entitlements based on how many entitlements should be allocated to the organization. The maximum and minimum totals are indicated just below the fields.
4. Click **Update Organization** to update all changes.

### 2.2.2. Changing an Organization's Software Channel Entitlement

To change system entitlements within the Organization:

1. Click the **Admin** menu, and select **Organization**.
2. Choose an organization from the list and select the **Subscriptions** tab.
3. Within the **Subscriptions** interface, click the **Software Channel Entitlements** tab to see all entitlements for all organizations, and their usage. Change the **Regular Proposed Total** according to the planned channel entitlements that should be allocated to the organization. The minimum and maximum totals are indicated below the field.

Channel entitlements can be either *Regular* or *Flex*. Any system can use a regular entitlement. Flex entitlements can only be used by systems that have been detected as being guests of a supported virtualization type.

4. Click on **Update Organization** to update all changes.



### Note

Organization Administrators that create a custom channel can only use that channel within their organization unless an Organizational Trust is established between the organizations that want to share the channel. For more information about organizational trusts, see [Section 2.3.3, “Managing Organizational Trusts”](#).

## 2.3. Managing Multiple Organizations

Red Hat Satellite supports the creation and management of *multiple organizations* within one Satellite installation, allowing for the division of systems, content, and subscriptions across different organizations or specific groups. This section guides the user through basic setup tasks and explains the concepts of multiple organization creation and management within Red Hat Satellite.

### 2.3.1. Modeling your Satellite for Multi-Organization Use

The following examples detail two possible scenarios using the multiple organizations (or multi-organization) feature. It is a good idea to create an additional organization and use it on a trial basis for a limited set of systems/users to fully understand the impact of a multi-organization Satellite on your organization's processes and policies.

#### 2.3.1.1. Centrally-Managed Satellite for A Multi-Department Organization

In this first scenario, the Red Hat Satellite is maintained by a central group within a business or other organization (see [Section 2.3.1.1, “Centrally-Managed Satellite for A Multi-Department Organization”](#)). The Satellite administrator of Organization 1 (the administrative organization created during Satellite configuration) treats Organization 1 (the 'Administrative Organization') as a staging area for software and system subscriptions and entitlements.

The Satellite administrator's responsibilities include the configuration of the Satellite (any tasks available under the **Admin** area of the web interface), the creation and deletion of additional Satellite organizations, and the allocation and removal of software and system subscriptions and entitlements.

Additional organizations in this example are mapped to departments within a company. One way to decide what level to divide the various departments in an organization is to think about the lines along which departments purchase subscriptions and entitlements for use with Red Hat Satellite. To maintain centralized control over organizations in the Satellite, create an Organization Administrator account in each subsequently created organization so that you may access that organization for any reason.



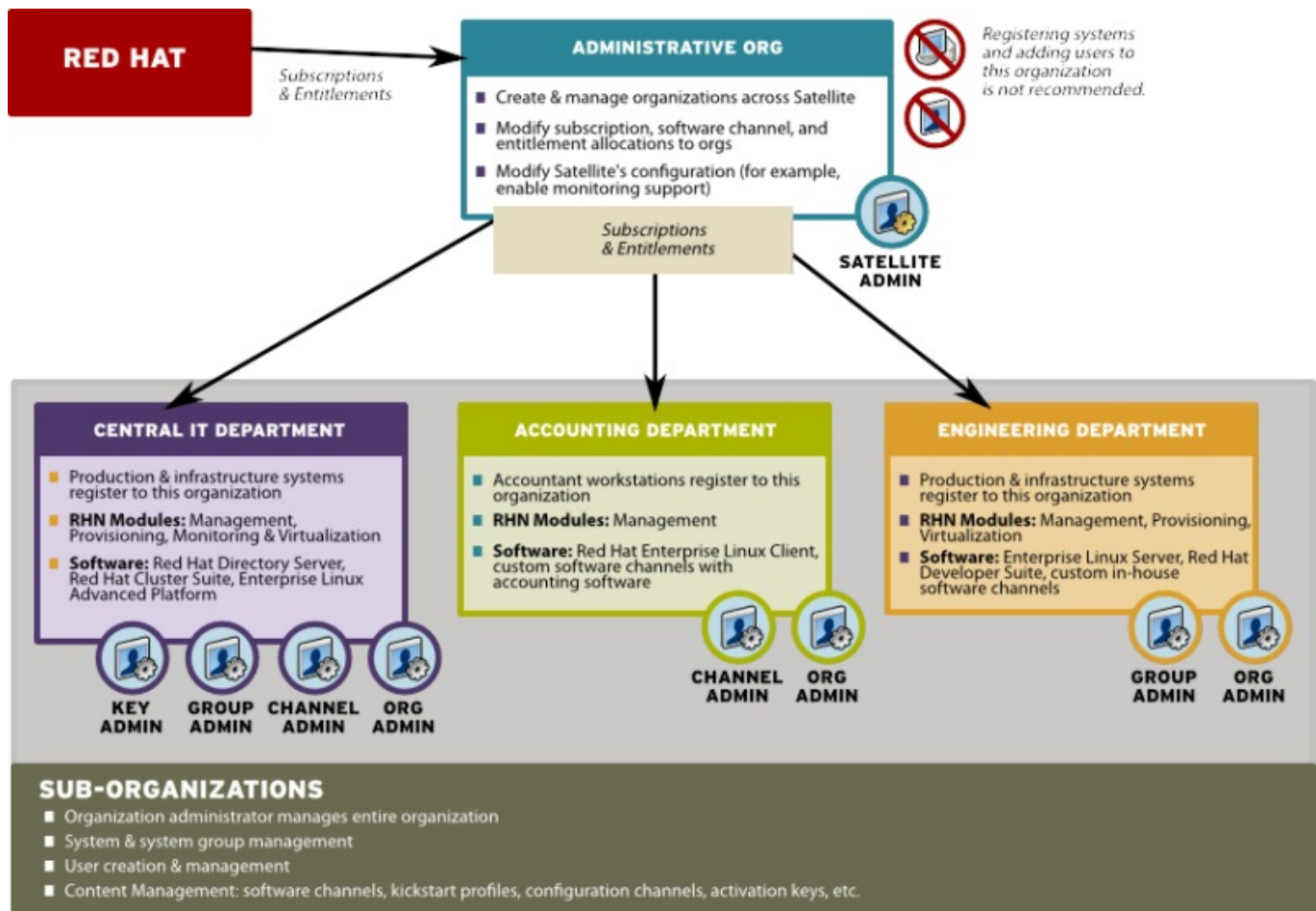


Figure 2.1. Centralized Satellite Management for Multi-Department Organization

### 2.3.1.2. Decentralized Management of Multiple Third Party Organizations

In this example, the Satellite is maintained by a central group, but each organization is treated separately without relations or ties to the other organizations on the Satellite. Each organization may be a customer of the group that manages the Satellite application itself.

While a Satellite consisting of sub-organizations that are all part of the same company may be an environment more tolerant of sharing systems and content between organizations, in this decentralized example sharing is less tolerable. Administrators can allocate entitlements in specific amounts to each organization. Each organization will have access to all Red Hat content synced to the Satellite if the organization has software channel entitlements for the content.

However, if one organization pushes custom content to their organization, it will not be available to other organizations. You cannot provide custom content that is available to all or select organizations without re-pushing that content into each organization.

In this scenario, Satellite Administrators might want to reserve an account in each organization to have login access. For example, if you are using Satellite to provide managed hosting services to external parties, reserve an account for yourself to access systems in that organization and push content.

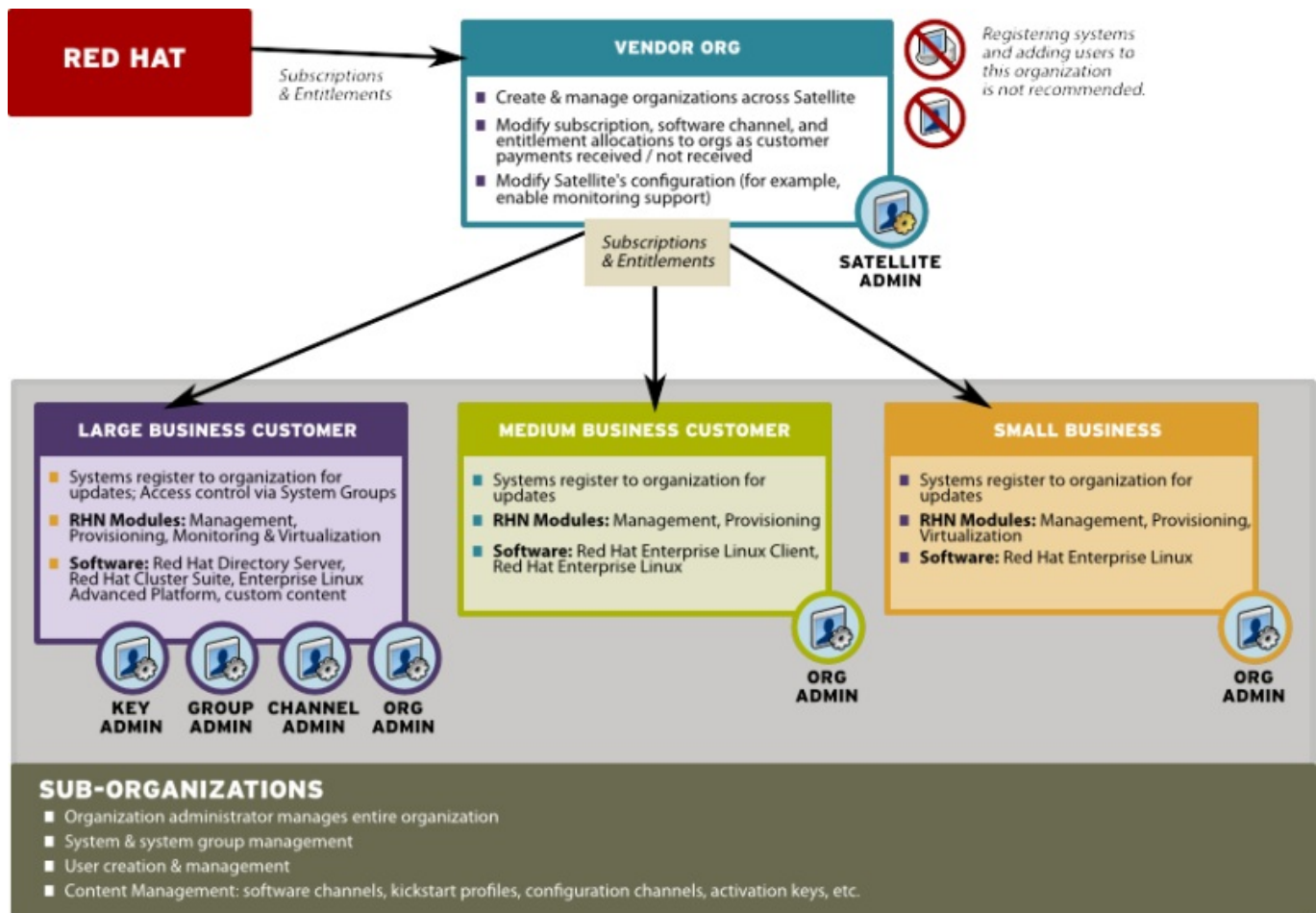


Figure 2.2. Decentralized Satellite Management for Multi-Department Organization

### 2.3.1.3. Recommendations for Multi-Organization Use

Regardless of the specific model you choose in the management of your multi-organization Satellite, these recommendations should be noted.

Administrative organizations (organization #1) should not be used to register systems and create users in any situation, unless you intend to:

- ✧ Use the Satellite as a single organization Satellite
- ✧ Are in the process of migrating from a single organization Satellite to a multiple organization Satellite

This is due to the following reasons:

1. The administrative organization is treated as a special case with respect to entitlements. You can only add or remove entitlements to this organization implicitly by removing them or adding them from the other organizations on the Satellite.
2. The administrative organization is a staging area for subscriptions and entitlements. When you associate the Satellite with a new certificate, any new entitlements will be granted to this organization by default. In order to make those new entitlements available to other organizations on the Satellite, you will need to explicitly allocate those entitlements to the other organizations from the administrative organization.
3. The Satellite server may only contain as many systems as there are entitlements in the Satellite certificate. Evaluate each organization's entitlement usage on the Satellite and decide how many entitlements are required for each organization to function properly. Each organization administrator should be aware of the entitlement constraint and manage the system profiles as required. Should

there be any issues, the Satellite administrator can step in to mediate entitlement concerns.



### Note

When logged in under a Satellite administrator, you cannot decrement the allocated entitlements to an organization below the number of entitlements that organization has actively associated with system profiles.

## 2.3.2. Configuring Systems in an Organization

Now that an organization has been created and requisite entitlements assigned to it, you can then assign systems to each organization.

There are two basic ways to register a system against a particular organization:

1. Registering Using Login and Password - If you provide a login and password created for a specified organization, the system will be registered to that organization. For example, if **user-123** is a member of the **Central IT** organization on the Satellite, the following command on any system would register that system to the **Central IT** organization on your Satellite:

```
# rhnreg_ks --username=user-123 --password=foobaz
```



### Note

The **--orgid** (for Red Hat Enterprise Linux 5) parameter in **rhnreg\_ks** are *not related* to Satellite registration or Red Hat Satellite's multiple organizations support.

2. Registering Using An Activation Key - You can also register a system to an organization using an activation key from the organization. Activation keys will register systems to the organization in which the activation key was created. Activation keys are a good registration method to use if you want to allow users to register systems into an organization without providing them login access to that organization. If you want to move systems between organizations, you may also automate the move with scripts using the activation keys.



### Note

Activation keys have a new format since Red Hat Satellite 5.1.0, so the first few characters of the activation key are used to indicate which organization (by ID number) owns the activation.

## 2.3.3. Managing Organizational Trusts

Organizations can share their resources with each other by establishing an *organizational trust* in the Satellite. An organizational trust is bi-directional, meaning that once a Satellite Administrator establishes a trust between two or more organizations, the Organization Administrator from each organization is free to share as much or as little of their resources as they need to. It is up to each Organization Administrator to determine what resources to share, and what shared resources from other organizations in the trust to use.

**Note**

Only Organization Administrators are able to share their custom content; Satellite Administrators only allocate system and software entitlements to each organization.

### 2.3.3.1. Establishing an Organizational Trust

A Satellite Administrator can create a trust between two or more organizations. To do this, click the **Organizations** link on the side menu on the **Admin** main page.

Click the name of one of the organizations and within the **Details** page, click the **Trusts** subtab.

On the **Trusts** subtab, there is a listing of all the other trusts on the Red Hat Satellite. Here you may use the **Filter by Organization** text box to narrow down a long list of organizations to a specific subset.

Click the checkbox next to the names of the organizations you want to be in the organizational trust with the current organization and click the **Modify Trusts** button.

### 2.3.3.2. Sharing Content Channels between Organizations in a Trust

Once an organizational trust has been established, organizations can now share content such as custom software channels with the other organizations in the trust. There are also three levels of channel sharing that can be applied to each channel for finer-grained channel access control.

**Note**

Organizations cannot share Red Hat Channels because they are available to all organizations that have entitlements to those channels.

To share a custom channel with another organization, perform the following steps:

1. Log in to the Satellite with the username of the Organization Administrator.
2. Click on the **Channels** → **Manage Software Channels**.
3. Click the custom channel that you want to share with the other organizations.
4. From the **Channel Access Control** section of the **Details** page, there are three choices for sharing in **Organizational Sharing**.
  - ✳ **Private** - Make the channel private so that it cannot be accessed by any organizations except the channel's owner.
  - ✳ **Protected** - Allow the channel to be accessed by specific trusted organizations of your choice.

**Note**

Choosing **Protected** sharing displays a separate page that prompts you to confirm that you are granting channel access to the organizations by clicking **Grant Access and Confirm**.

- » **Public** - Allow all organizations within the trust to access the custom channel.

Click the radio button next to your selection and click **Update Channel**.

Now, any other Organization Administrators within the trust for which you have granted access to your custom channel can allow their client systems to install and update packages from the shared channel.



### Note

If you have a system subscribed to a shared channel, and the organizational administrator of the shared channel changes access rights to the channel, then the system loses that channel. If the administrator changes a base channel, the system has no base channel on the **Systems** page and will not receive updates.

### 2.3.3.3. Migrating Systems from One Trusted Organization to Another

In addition to sharing software channels, organizations in a trust can migrate systems to other trusted organizations. There are two methods of doing this, one through the Satellite web interface and the other, by using a utility called **migrate-system-profile**.



### Note

The Satellite Administrator can migrate a system from one trusted organization to any other in the trust. However, Organization Administrators can only migrate a system from their own organization to another in the trust.

#### 2.3.3.3.1. Using the Satellite Interface to Migrate Systems

##### Procedure 2.1. To Migrate a System Between Organizations:

1. Click the **Systems** tab and then click the name of the system that you want to migrate.
2. Click **Details** → **Migrate** and select the name of the organization that you want to migrate the system to.
3. Click **Migrate System**.

#### 2.3.3.3.2. Using migrate-system-profile

**migrate-system-profile** usage is based on the command-line, and uses systemIDs and orgIDs as arguments to specify what is being moved and its destination organization.

To use the **migrate-system-profile** command, you must have the **spacewalk-utils** package installed. You do not need to be logged into the Satellite server to use **migrate-system-profile**; however, if you do not you will need specify the hostname or IP address of the server as a command-line switch.

**Note**

When an organization migrates a system with the **migrate-system-profile** command, the system does not carry any of the previous entitlements or channel subscriptions from the source organization. However, the system's history is preserved, and can be accessed by the new Organization Administrator in order to simplify the rest of the migration process, which includes subscribing to a base channels and granting entitlements.

Verify the ID of the system to be migrated, the ID of the organization the system will migrate to, and the hostname or IP address of the Satellite server if you are running the command from another machine.

**Note**

Find the system ID and organization ID using either the Web UI or the **spacewalk-report** tool.

Once you have this data, the usage from the command line is as follows:

```
# migrate-system-profile --satellite {SATELLITE HOSTNAME OR IP} --
systemId={SYSTEM ID} --to-org-id={DESTINATION ORGANIZATION ID}
```

**Example 2.1. Migration from one department to another**

The Finance department wants to migrate a workstation from the Engineering department but the Finance Organization Administrator does not have shell access to the Red Hat Satellite server. This example uses the following data:

- ✱ Finance department organization ID is **2**
- ✱ The workstation has a system ID of **10001020**
- ✱ The Red Hat Satellite hostname is **satserver.example.com**

The Finance Organization Administrator would type the following from a shell prompt:

```
# migrate-system-profile --satellite=satserver.example.com --
systemId=10001020 --to-org-id=2
```

The Finance Organization Administrator is then prompted for their username and password (unless they specified it using **--username=** and **--password=** at the command-line).

The Finance Organization Administrator would then be able to see the system from the **Systems** page when logged into the Red Hat Satellite web interface. The Finance Organization Administrator can then finish the migration process by assigning a base channel and granting entitlements to the client like any other system registered to his organization, which is available from the system's **History** page in the **Events** subtab.

Satellite Administrators that need to migrate several systems at once can use the **--csv** option of **migrate-system-profile** to automate the process using a simple comma-separated list of systems to migrate.

A line in the CSV file should contain the ID of the system to be migrated as well as destination organization's ID in the following format:

```
systemId, to-org-id
```

A compatible CSV could look like the following:

```
1000010000,3  
1000010020,1  
1000010010,4
```

For more information about using **migrate-system-profile** see the manual page by typing **man migrate-system-profile** or for a basic help screen type **migrate-system-profile -h**.



## Chapter 3. System Provisioning

### 3.1. Provisioning Through Red Hat Satellite

Provisioning is the process of configuring a physical or virtual machine to a predefined known state. Red Hat Satellite provisions systems using the *kickstart* process. To use the provisioning functionality, one or more *target* machines are required. The target machines can be either physical, bare metal systems, or virtual machines. To use Red Hat Satellite's virtual machine provisioning functionality, create the virtual machines using either Xen or KVM.

System administrators can use an automated installation method to install Red Hat Enterprise Linux on their machines through the kickstart installation method. Using kickstart, a system administrator can create a single file containing the answers to all the questions that would normally be asked during a typical installation.

Kickstart files can be kept on a single server system and read by individual computers during the installation. This installation method can support the use of a single kickstart file to install Red Hat Enterprise Linux on multiple machines.

Base images, kickstart files, and other content can be accessed using HTTP by using the Satellite server URL. For example, to access kickstart files for Red Hat Enterprise Linux 6 for 64-bit on the Satellite server, the base URL would be `http://satellite.example.com/ks/dist/ks-rhel-x86_64-server-6-6.4`, followed by the name of the package you wish to download, such as:  
`http://satellite.example.com/ks/dist/ks-rhel-x86_64-server-6-6.4/GPL`.

The *Red Hat Enterprise Linux Installation Guide* contains an in-depth discussion of kickstart.

#### Definitions

Some terms used throughout this section:

##### Kickstarting

The process of installing a Red Hat Enterprise Linux system in an automated manner requiring little or no user intervention. Technically, *kickstart* refers to a mechanism in the Anaconda installation program that allows concise description of the contents and configuration of a machine to be written into the installer, which it then acts on. Such a concise system definition is referred to as a *Kickstart profile*.

##### Kickstart profile

The kickstart file is a text file that specifies all of the options needed to kickstart a machine, including partitioning information, network configuration, and packages to install. In Red Hat Satellite, a Kickstart Profile is a superset of a traditional Anaconda kickstart definition, as Satellite's implementation builds on Cobbler's enhancements to kickstarting. A Kickstart Profile presumes the existence of a Kickstart Tree.

##### Kickstart Tree

The software and support files needed in order to kickstart a machine. This is also often called an "install tree". This is usually the directory structure and files pulled from the installation media that ships with a particular release. In Cobbler terminology, a Kickstart Tree is part of a distribution.

##### PXE (Preboot eXecution Environment)

A low-level protocol that makes it possible to kickstart bare metal machines (usually physical, or *real*, machines) on power-up with no pre-configuration of the target machine itself. PXE relies on a



DHCP server to inform clients about bootstrap servers. PXE must be supported in the firmware of the target machine in order to be used. It is possible to use the virtualization and reinstall facilities of Satellite without PXE, though PXE is very useful for booting new physical machines, or reinstalling machines that are not registered to Satellite.

## Provisioning Scenarios

The types of provisioning scenarios supported by Red Hat Satellite:

### New installations

Provisioning a system that has not previously had an operating system installed (also known as *bare metal* installations).

### Virtual installations

Satellite supports KVM, Xen fully-virtualized guests, and Xen para-virtualized guests.

### Re-provisioning

Both physical and guest systems can be re-provisioned, provided that they have been registered to the same Satellite instance. See [Section 3.5, “Reprovisioning”](#).

### 3.1.1. Kickstart Explained

When a machine is to receive a network-based kickstart, the following events must occur in this order:

1. After being placed on the network and turned on, the machine's PXE logic broadcasts its MAC address and a request to be discovered.
2. If a static IP address is not being used, the DHCP server recognizes the discovery request and extends an offer of network information needed for the new machine to boot. This includes an IP address, the default gateway to be used, the netmask of the network, the IP address of the TFTP or HTTP server holding the bootloader program, and the full path and file name of that program relative to the server's root.
3. The machine applies the networking information and initiates a session with the server to request the bootloader program.
4. The bootloader, once loaded, searches for its configuration file on the server from which it was itself loaded. This file dictates which kernel and kernel options, such as the initial RAM disk (initrd) image, should be executed on the booting machine. Assuming the bootloader program is SYSLINUX, this file is located in the **pxelinux.cfg** directory on the server and named the hexadecimal equivalent of the new machine's IP address. For example, a bootloader configuration file for Red Hat Enterprise Linux 6 should contain:

```
default=0
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title Red Hat Enterprise Linux Server (2.6.32-279.22.1.el6.x86_64)
  root (hd0,0)
  kernel /vmlinuz-2.6.32-279.22.1.el6.x86_64 ro root=/dev/sda2
  crashkernel=auto ks=http://example.com/ks.cfg
  initrd /initramfs-2.6.32-279.22.1.el6.x86_64.img
```

5. The machine accepts and uncompresses the init image and kernel, boots the kernel, and initiates a

kickstart installation with the options supplied in the bootloader configuration file, including the server containing the kickstart configuration file.

6. This kickstart configuration file in turn directs the machine to the location of the installation files.
7. The new machine is built based upon the parameters established within the kickstart configuration file.

### 3.1.2. Prerequisites

The provisioning entitlement is required in order for System Provisioning to work. Without this entitlement, the tab for Provisioning/Kickstarting will not appear on the Satellite.

The organization's infrastructure should also be prepared to handle kickstart. Some factors to consider before creating kickstart profiles are:

- ✦ A DHCP server. This is not required for kickstarting, but a DHCP server will ease the need to configure network settings in the kickstart file. You may also boot from the network. If you do not have a DHCP server and are using static IP addresses, you should select static IP while developing your kickstart profile.
- ✦ An FTP server. It can be used in place of hosting the kickstart distribution trees via HTTP.

If you are conducting a bare metal kickstart:

1. Configure DHCP to assign required networking parameters and the bootloader program location.
2. Specify the kernel to be used and the appropriate kernel options within the bootloader configuration file.

#### 3.1.2.1. Required Packages

If the system is using a custom distribution, it will require the following packages, which are available from any **rhn-tools** Red Hat Network (RHN) channel:

- ✦ **koan**
- ✦ **spacewalk-koan**

It is advisable to clone an existing **rhn-tools** channel in order to have access to these packages from your custom channel.

Red Hat Satellite expects the **kernel** and **initrd** files to be in specific locations within the kickstart tree. However, these locations are different for different architectures. The table below explains the different locations:

**Table 3.1. Required Distribution Files by Architecture**

Architecture	kernel	Initial RAM Disk image
IBM System z	<i>TREE_PATH/images/kernel.img</i>	<i>TREE_PATH/images/initrd.img</i>
PowerPC	<i>TREE_PATH/ppc/ppc64/vmlinuz</i>	<i>TREE_PATH/ppc/ppc64/initrd.img</i>
All other architectures	<i>TREE_PATH/images/pxeboot/vmlinuz</i>	<i>TREE_PATH/images/pxeboot/initrd.img</i>

#### 3.1.2.2. Kickstart Trees

There must be at least one kickstart tree installed on the Satellite in order to use kickstart provisioning. Kickstart trees can be installed either automatically or manually.

### Procedure 3.1. Installing Kickstart Trees Automatically

For all distributions that have a base channel in Red Hat Satellite, kickstart trees can be installed automatically. This occurs as part of normal channel synchronization via **satellite-sync**.

1. Choose which distribution to base the kickstarts on and locate that distribution's base channel, as well as its corresponding Red Hat Network Tools channel.

For example, to use Red Hat Enterprise Linux 6 with x86 architecture, the **rhel-x86\_64-server-6** channel and its corresponding Red Hat Network Tools channel **rhn-tools-rhel-x86\_64-server-6** are required.

2. If it is a connected Satellite, synchronize the Satellite server with the Red Hat servers directly using the **satellite-sync** command. If the Satellite server is disconnected, it is necessary to obtain disconnected channel dumps from the Red Hat servers and synchronize with those.
3. Synchronizing the channel will automatically create a corresponding kickstart tree for that distribution.

### Procedure 3.2. Installing Kickstart Trees Manually

To kickstart a custom distribution, which is usually a distribution not supported by Red Hat, or a beta version of Red Hat Enterprise Linux, create the corresponding kickstart tree manually. An installation ISO is required for the distribution used for the kickstart.

1. Copy the installation ISO to the Satellite server and mount it to **/mnt/iso**.
2. Copy the contents of the ISO to a custom location. It is recommended to create a directory within **/var/satellite** for all custom distributions. For example, copying a Red Hat Enterprise Linux 6 beta distribution's contents to **/var/satellite/custom-distro/rhel-x86\_64-server-6-beta/**.
3. Use the Red Hat Satellite web interface to create a custom software channel. Use **Channels** → **Manage Software Channels** → **Create New Channel** to create a parent channel with an appropriate name and label. For the example used above, use the label **rhel-6-beta**.
4. Push the software packages from the tree location to the newly created software channel using the **rhnpush** command:

```
# rhnpush --server=http://localhost/APP -c 'rhel-6-beta' \ -d
/var/satellite/custom-distro/rhel-x86_64-server-6-beta/Server/
```

The sub-directory within the tree could be different depending on the distribution.

5. Once the software packages have been pushed, they can be deleted from within the tree path using the **rm** command. The packages are still stored on the Satellite server within the channel, and are no longer required in the tree.

```
# rm /var/satellite/custom-distro/rhel-x86_64-server-6-
beta/Server/*.rpm
```

You can choose to leave the software packages within the kickstart tree. This will allow them to be installed with the **yum** command at any time later on.

6. Use the Red Hat Satellite web interface to create the distribution. Use **Systems** → **Kickstart** →

**Distributions** → **create new distribution** to create the distribution, using an appropriate label and the full tree path (such as `/var/satellite/custom-distro/rhel-x86_64-server-6-beta/`). Select the base channel that was created earlier, and the correct Installer Generation (such as **Red Hat Enterprise Linux 6**). To complete the creation, select **Create Kickstart Distribution**.

7. To maintain the same software across multiple environments and systems, the Red Hat Network Tools child channel from an existing Red Hat Enterprise Linux base channel can be cloned as a child channel of the newly created base channel. Cloning a child channel can be done by:
  - a. On the Satellite web interface, click on **Channels** → **Manage Software Channels** → **Clone Channel**
  - b. Choose the Child Channel you wish to clone from the drop down box **Clone From:** and choose the clone state.
  - c. Click **Create Channel**.
  - d. Fill in the necessary information and choose the parent channel that the cloned child channel needs to be under.
  - e. Click **Create Channel**.

### 3.1.2.3. Kickstart Profiles

Kickstart profiles specify the configuration options to be used for the installation.

Kickstart profiles can be created using a *wizard* interface, which generates a profile based on the answers you give to a series of questions. They can also be created using the *raw method*, which gives complete control over the contents of the profile.

#### Procedure 3.3. Creating a Kickstart Profile with a Wizard

1. Select **Systems** → **Kickstart** → **Create a New Kickstart Profile**
2. Provide an appropriate **Label**, and select the desired **Base Channel** and **Kickstartable Tree**
3. Select the desired **Virtualization Type**. See [Section 3.1.2.3, “Kickstart Profiles”](#) for more information about virtualization types. Click **next** to continue.
4. Select the download location for the kickstart profile. For custom distributions, enter the location of its tree as a URL (both HTTP and FTP are supported), otherwise, use the default option. Click **next** to continue.
5. Enter the root password and click **finish** to complete the profile creation.
6. The complete kickstart profile will be created. View the profile by clicking **Kickstart File**.

#### Procedure 3.4. Creating a Kickstart Profile with the Raw Method

1. Select **Systems** → **Kickstart** → **Upload a New Kickstart File**
2. Provide an appropriate **label**, and select the desired **distribution**
3. Select the desired **Virtualization Type**. See [Section 3.1.2.3, “Kickstart Profiles”](#) for more information about virtualization types.

4. If there is an existing kickstart profile, upload the file. Otherwise, write the kickstart profile in the **File Contents** text box.

Here is a sample raw kickstart that can be used as a starting point:

```
install
text
network --bootproto dhcp
url --url http://$http_server/ks/dist/org/1/ks-rhel-x86_64-server-6-6.4
lang en_US
keyboard us
zerombr
clearpart --all
part / --fstype=ext3 --size=200 --grow
part /boot --fstype=ext3 --size=200
part swap --size=1000 --maxsize=2000
bootloader --location mbr
timezone America/New_York
auth --enablemd5 --enablesshadow
rootpw --iscrypted $1$X/CrCfCE$x0veQ088TCm2VprcMkH.d0
selinux --permissive
reboot
firewall --disabled
skipx
key --skip

%packages
@ Base

%post
$SNIPPET('redhat_register')
```

5. The Red Hat Satellite server does not handle the specified distribution as the **url** in the kickstart, so remember to include the **url --url** option in the profile, similar to the following:

```
url --url http://$http_server/ks/dist/org/1/my_distro
```

Replace **my\_distro** with the distribution label and **1** with your org id.

6. Raw kickstart profiles use **\$http\_server** instead of the Satellite's host name. This will be filled in automatically when the kickstart template is rendered.
7. The **redhat\_register** snippet is used to handle registration.

**Create Kickstart Profile**

**Kickstart Details**

Each kickstart file you upload to satellite will need a label so that you can refer to it later - please choose a label for this kickstart and enter it below. Entries marked with an asterisk (\*) are required.

**Label\*:**

**Kickstartable Tree\*:**

**Virtualization Type:**   
**NOTE:** Changing the Virtualization Type may require changes to the kickstart profile's bootloader and partition options, potentially overwriting user customizations. Please visit the [Partitioning](#) tab to verify the new settings.

**File Contents:**

☒ Toggle editor

**Tip:** Alternatively, you can upload a new version in the 'File to Upload' section below and select 'Upload File'.

**Tip:** If you wish to re-use an existing Kickstart Profile we recommend you copy-paste the template from `/var/lib/rhn/kickstarts` instead of from your browser. This will ensure that the template variables will be preserved and the registration process will work properly after the kickstart.

**File to Upload**

32.06 KB/s 0.999 Done


Figure 3.1. Raw Kickstart

## Virtualization Types

All kickstart profiles have a virtualization type associated with them. This table outlines the different options:

Table 3.2. Virtualization Types

Type	Description	Uses
none	No virtualization	Use this type for normal provisioning, bare metal installations, and virtualized installation that are not Xen or KVM (such as VMware, or Virtage)
KVM Virtualized Guest	KVM guests	Use this type for provisioning KVM guests

Type	Description	Uses
<b>Xen Fully-Virtualized Guest</b>	Xen guests	Use this type for provisioning Xen guests
<div>  <b>Note</b>            This option requires hardware support on the host, but does not require a modified operating system on the guest.         </div>		
<b>Xen Para-Virtualized Guest</b>	Xen Guests	Use this type for provisioning a virtual guest with Xen para-virtualization. Para-virtualization is the fastest virtualization mode. It requires a PAE flag on the system CPU, and a modified operating system. Only Red Hat Enterprise Linux 5 supports guests under para-virtualization.
<b>Xen Virtualization Host</b>	Xen hosts	Use this type for provisioning a virtual host with Xen para-virtualization. Xen para-virtualized guests and hosts are supported, if the hardware is compatible. Supported on Red Hat Enterprise Linux 5 only.

Kickstart profiles created to be used as Xen hosts must include the **kernel-xen** package in the **%packages** section.

Kickstart profiles created to be used as KVM hosts must include the **qemu** package in the **%packages** section.

Fully virtualized systems may require virtualization support to be turned on in the computer's BIOS menu.



### Note

For more information about kickstart, see the *Kickstart Installations* section in the *Red Hat Enterprise Linux Installation Guide*.

### 3.1.2.4. Templating

Kickstart *templating* allows the inclusion of variables, snippets, and flow control statements such as **for** loops and **if** statements in the kickstart files. This is achieved using the **cheetah** tool.

There are a variety of reasons why templating might be useful:

- » Reusing a particular section of a kickstart, such as a disk partitioning section, between multiple kickstarts.

- ✱ Performing some **%post** actions consistently across multiple kickstarts.
- ✱ Defining a snippet across multiple server roles such as DNS server, proxy server, and web server. For example, a web server might have the following snippet defined:

```
httpd
mod_ssl
mod_python
```

To create a web server profile, include the web server snippet in the **%package** section of the kickstart file. For a profile to be both a web server and a proxy server, include both snippets in the package section. To add another package to the web server snippet, **mod\_perl** for example, update the snippet, and all profiles that are using that snippet are dynamically updated.

## Variables

Templating allows the defining of a variable to be used throughout a kickstart file. Variables are subject to a form of inheritance that allows them to be set at one level and overridden at levels below them. So, if a variable is defined at the system level, it will override the same variable defined at the profile or kickstart tree levels. Likewise, if a variable is defined at the profile level, it will override the same variable defined at the kickstart tree level.



### Note

Note that kickstart tree variables cannot be defined for automatically generated kickstart trees, such as those created when a satellite synchronization is performed.

## Snippets

Snippets reuse pieces of code between multiple kickstart templates. They can span many lines, and include variables. They can be included in a kickstart profile by using the text **\$SNIPPET( 'snippet\_name' )**. A snippet can be made for a package list, for a **%post** script, or for any text that would normally be included in a kickstart file.

To manage snippets navigate to **Systems → Kickstart → Kickstart Snippets**.

The **Kickstart Snippets** page displays several default snippets that cannot be edited, but are available to be used by any organization. Default snippets can be used in kickstarts that have been either written on or uploaded to the Red Hat Satellite server. Default snippets are stored on the Red Hat Satellite server's file system in **/var/lib/cobbler/snippets/**. Once Kickstart profiles are created, review the contents of **/var/lib/rhn/kickstarts/** to see how snippets are incorporated into the profiles.

The **redhat\_register** snippet is a default snippet that is used to register machines to a Red Hat Satellite server as part of a kickstart. It uses a variable called **redhat\_management\_key** to register the machine. To use the snippet, set the **redhat\_management\_key** variable at either the system, profile, or distribution level and then add **\$SNIPPET( 'redhat\_register' )** to a **%post** section of the kickstart. Any wizard-style kickstarts that are generated by the Red Hat Satellite server will already include this snippet in the **%post** section.

The **Custom Snippets** tab allows the viewing and editing of snippets created for use by the organization. New snippets can be created by clicking **create new snippet**. Custom snippets are stored in the **/var/lib/rhn/kickstarts/snippets/** directory. Red Hat Satellite stores snippets for different organizations in different directories, so custom snippets will be stored with a filename similar to the following, where **1** is the organization ID:



```
$SNIPPET('spacewalk/1/snippet_name')
```

To determine the text to use to insert the snippet in the kickstart, look for the **Snippet Macro** column on the snippet list, or on the snippet details page.



## Note

Snippets exist at a global level and do not share the same inheritance structure as variables. However, variables can be used within snippets to change the way they behave when different systems request a kickstart.

English (change) Knowledgebase Documentation USER: jsherrill ORGANIZATION: Justin Sherrill Preferences Sign Out

RED HAT NETWORK SATELLITE Systems Search

Overview Systems Errata Channels Configuration Schedule Users Monitoring Help 1 SYSTEM SELECTED MANAGE CLEAR

**Kickstart Snippets** create new snippet

Default Snippets Custom Snippets All Snippets

Use kickstart snippets to store common blocks of code that can be shared across multiple kickstart profiles in RHN Satellite. When you create a kickstart snippet, all kickstart profiles including that snippet will be updated accordingly.

0 1 2 3 4 5 6 7 8 9 A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Filter by Snippet Name:  Display 25 items per page 1 - 1 of 1

Go

Snippet Name	Snippet Macro
MyCustomSnippet	\$SNIPPET('spacewalk/28/MyCustomSnippet')

1 - 1 of 1

Tip: Copy and paste the snippet macro into your kickstart profiles to make the full snippet appear in that kickstart profile.

Copyright © 2002-09 Red Hat, Inc. All rights reserved. Privacy statement Legal statement redhat.com  
RHN Satellite release 5.3

Figure 3.2. Kickstart Snippets

## Escaping Special Characters

The **\$** and **#** characters are used during templating for specifying variables and control flow. If these characters are needed for any other purpose in a script, they will need to be escaped so that they are not recognized as a variable. This can be achieved in several ways:

- ✦ Placing a backslash character (**\**) before every instance of **\$** or **#** that needs to be ignored during templating.

- ✧ Wrap the entire script in **#raw ... #end raw**

All **%pre** and **%post** scripts created using the wizard-style kickstarts are wrapped with **#raw...#end raw** by default. This can be toggled using the **Template** checkbox available when editing a **%post** or **%pre** script.

- ✧ Include **#errorCatcher Echo** in the first line of the snippet.

### Example 3.1. Escaping Special Characters in templates

This example describes how to escape special characters in kickstart templates.

The following bash script needs to be inserted in a **%post** section:

```
%post
echo $foo > /tmp/foo.txt
```

Without the **\$** being escaped, the templating engine will try to find a variable named **\$foo** and would fail because **foo** does not exist as a variable.

The simplest way to escape the **\$** is by using a backslash character (**\**):

```
%post
echo \ $foo > /tmp/foo.txt
```

This will cause **\ \$foo** to be rendered as **\$foo**.

A second method is to wrap the entire bash script in **#raw ... #end raw**, as follows

```
%post
#raw
echo $foo > /tmp/foo.txt
#end raw
```

The final method is to include **#errorCatcher Echo** in the first line of the kickstart template. This instructs the templating engine to ignore any variables that do not exist and print out the text as is. This option is already included in the wizard-style kickstarts, and can be included in any raw kickstarts created manually.

### 3.1.2.5. Kickstarting from Bare Metal

When a machine has no existing operating system or has the wrong operating system installed, it is referred to as a *bare metal* machine. There are three ways to provision a machine from bare metal:

- ✧ Standard operating system installation media
- ✧ PXE boot
- ✧ Yaboot for PowerPC

### Procedure 3.5. Booting from Installation Media

1. Insert installation media into the machine. The media must match the kickstart intended to use. For example, if the kickstart is configured to use the **ks-rhel-x86\_64-server-6-6.4** kickstart tree, use the Red Hat Enterprise Linux 6.4 64-bit installation media.
2. At the boot prompt, activate the kickstart by giving this command:

```
linux ks=http://satellite.example.com/path/to/kickstart
```

3. Once the system boots up, download the kickstart file, and install automatically.

### Procedure 3.6. PXE Booting

To perform a PXE boot, each system must support PXE booting at the BIOS level. Additionally, a DHCP server is required, even if the systems are to be configured statically after installation.

1.



#### Important

If a DHCP server is deployed on another system on the network, administrative access to the DHCP server is required in order to edit the DHCP configuration file.

### Prerequisites

The latest cobbler-loaders package needs to be installed. This is to ensure that the **pxelinux.0** boot image is installed and available on the Satellite before PXE booting. To install the latest version:

```
# yum install cobbler-loaders
```

If the machines reside on multiple networks, ensure that all of the machines can connect to the DHCP server. This can be achieved by multi-homing the DHCP server (using either a real or trunked VLAN) and configuring any routers or switches to pass the DHCP protocol across network boundaries.

Configure the DHCP server so that it points to the PXE server by setting the **next-server** address for the systems to be managed by Red Hat Satellite.

To use hostnames when performing the installation, configure the DHCP server to point to the domain and IP addresses, by including the following lines:

```
option domain-name DOMAIN_NAME;
option domain-name-servers IP_ADDRESS1, IP_ADDRESS2;
```

2. On the DHCP server, switch to the root user and open the **/etc/dhcpd.conf** file. Append a new class with options for performing PXE boot installation:

```
allow booting;
allow bootp;
class "PXE" {
    match if substring(option vendor-class-identifier, 0, 9) =
```

```
"PXEClient";
  next-server 192.168.2.1;
  filename "pxelinux.0";
}
```

This class will perform the following actions:

- a. Enable network booting with the **bootp** protocol
- b. Create a class called **PXE**. If a system is configured to have PXE first in its boot priority, it will identify itself as **PXEClient**.
- c. The DHCP server directs the system to the Cobbler server at the IP address *192.168.2.1*
- d. The DHCP server refers to the boot image file at **/var/lib/tftpboot/pxelinux.0**

Restart your DHCP server:

```
# service dhcpd restart
```

3. Configure Xinetd. Xinetd is a daemon that manages a suite of services including TFTP, the FTP server used for transferring the boot image to a PXE client.

Enable Xinetd using the **chkconfig** command:

```
# chkconfig xinetd on
```

Alternatively, switch to the root user and open the **/etc/xinetd.d/tftp** file. Locate the **disable = yes** line and change it to read **disable = no**.

4. Start the Xinetd service so that TFTP can start serving the **pxelinux.0** boot image:

```
# chkconfig --level 345 xinetd on
# /sbin/service xinetd start
```

The **chkconfig** command turns on the **xinetd** service for all user runlevels, while the **/sbin/service** command turns on **xinetd** immediately.

### Procedure 3.7. Yaboot Booting

Yaboot is the boot system that operates within the Open Firmware layer for PowerPC. Performing a kickstart requires some configuration to your environment and PowerPC clients.

1. Configure the boot order for your PowerPC clients. This involves accessing the Open Firmware interface and running the following commands at the prompt.

List the aliases of all devices on your system using the **devalias** command:

```
0 > devalias
ibm,sp          /vdevice/IBM,sp@4000
disk
/pci@8000000020000002/pci@2,4/pci1069,b166@1/scsi@1/sd@5,0
network         /pci@8000000020000002/pci@2/ethernet@1
net             /pci@8000000020000002/pci@2/ethernet@1
network1        /pci@8000000020000002/pci@2/ethernet@1,1
scsi            /pci@8000000020000002/pci@2,4/pci1069,b166@1/scsi@0
```

```
nvramp      /vdevice/nvramp@4002
rtc         /vdevice/rtc@4001
screen     /vdevice/vty@300000000
ok
```

Display the current boot order by checking the **boot-device** environment variable:

```
0 > printenv boot-device
----- Partition: common ----- Signature: 0x70 -----
-----
boot-device      /pci@8000000020000002/pci@2,3/ide@1/disk@0
/pci@8000000020000002/pci@2,4/pci1069,b166@1/scsi@1/sd@5,0
ok
```

Add the **network** device to the top of the boot order by setting the **boot-device** environment variable with the **network** device first and then the existing boot devices:

```
0 > setenv boot-device network
/pci@8000000020000002/pci@2,3/ide@1/disk@0
/pci@8000000020000002/pci@2,4/pci1069,b166@1/scsi@1/sd@5,0
```

The system is now configured to boot from the **network** device first. If **network** fails, the system will boot from the remaining devices in the boot order.

- Set the system configuration properties on the Satellite server. For example, the following command creates a new system called **myppc01** using a specific MAC address on the network:

```
# cobbler system add --name myppc01 --hostname myppc01.example.com --
profile rhel6webserver--kopts "console=hvc0 serial" --interface 0 --
mac 40:95:40:42:F4:46
```

This also creates a set of directories for the Yaboot bootloader and configuration based upon the MAC address specified. For example, the command previously used would create the following directories:

```
/var/lib/tftpboot/ppc/40-95-40-42-F4-46
/var/lib/tftpboot/etc/40-95-40-42-F4-46
```

The first directory (**/var/lib/tftpboot/ppc/40-95-40-42-F4-46**) contains the **ramdisk** and **mlinuz** files used for for Yaboot. The second directory (**/var/lib/tftpboot/etc/40-95-40-42-F4-46**) contains the configuration file (**yaboot.conf**) with the Yaboot settings.

See [Section 3.1.4.6, “Adding a System to Cobbler”](#) for more information about using Cobbler to provision systems.

- On the DHCP server, switch to the root user and open the **/etc/dhcpd.conf** file. Append a new entry with options for performing Yaboot installation. For example:

```
allow booting;
allow bootp;
class "Yaboot" {
    match if substring(option vendor-class-identifier, 0, 9) =
```

```
"AAPLBSDPC";
  next-server 192.168.2.1;
  filename "yaboot";
}
```

This class will perform the following actions:

- a. Enable network booting with the **bootp** protocol
- b. Create a class called **Yaboot**. If a system is configured to have Yaboot first in its boot priority, it will identify itself as **AAPLBSDPC**.
- c. The DHCP server directs the system to the Cobbler server at the IP address *192.168.2.1*
- d. The DHCP server refers to the Yaboot image file at created previously with **cobbler**

Restart your DHCP server:

```
# service dhcpd restart
```

4. Configure Xinetd. Xinetd is a daemon that manages a suite of services including TFTP, the FTP server used for transferring the boot image to a PowerPC client.

Enable Xinetd using the **chkconfig** command:

```
# chkconfig xinetd on
```

Alternatively, switch to the root user and open the `/etc/xinetd.d/tftp` file. Locate the **disable = yes** line and change it to read **disable = no**.

5. Start the Xinetd service so that TFTP can start serving the Yaboot boot image:

```
# chkconfig --level 345 xinetd on
# /sbin/service xinetd start
```

The **chkconfig** command turns on the **xinetd** service for all user runlevels, while the **/sbin/service** command turns on **xinetd** immediately.

### 3.1.3. Using Activation Keys

Red Hat Network Management and Provisioning customers with the Activation Key Administrator role (including Satellite Administrators) can generate activation keys through the Red Hat Satellite website. These keys can then be used to register a Red Hat Enterprise Linux system, entitle the system to a Red Hat Satellite service level and subscribe the system to specific channels and system groups through the command line utility **rhnmreg\_ks**.



#### Note

System-specific activation keys created through the **Reactivation** subtab of the **System Details** page are not part of this list because they are not reusable across systems.

## Procedure 3.8. Managing Activation Keys

To generate an activation key:

1. Select **Systems** → **Activation Keys** from the top and left navigation bars.
2. Click the **create new key** link at the top-right corner.



### Warning

In addition to the fields listed below, Red Hat Satellite customers may also populate the **Key** field itself. This user-defined string of characters can then be supplied with **rhnmreg\_ks** to register client systems with the Satellite. *Do not insert commas in the key.* All other characters are accepted. Commas are problematic since they are the separator used when including two or more activation keys at once. See [Section 3.1.3, “Using Activation Keys”](#) for details.

3. Provide the following information:

- ✳ **Description** - User-defined description to identify the generated activation key.
- ✳ **Usage** - The maximum number of registered systems that can be registered to the activation key at any one time. Leave blank for unlimited use. Deleting a system profile reduces the usage count by one and registering a system profile with the key increases the usage count by one.
- ✳ **Base Channels** - The primary channel for the key. Selecting nothing will enable you to select from all child channels, although systems can be subscribed to only those that are applicable.
- ✳ **Add-on Entitlements** - The supplemental entitlements for the key, which includes Monitoring, Provisioning, Virtualization, and Virtualization Platform. All systems will be given these entitlements with the key.
- ✳ **Universal default** - Whether or not this key should be considered the primary activation key for your organization. Any client registering to Satellite without an activation key specified will use the Universal Default activation key. Ideally, the Universal Default activation key would include a standard or minimum set of channels and entitlements for basic system registration. For more information on Universal Default activation keys, see <https://access.redhat.com/solutions/1140083>.

Click **Create Activation Key**.

After creating the unique key, it appears in the list of activation keys along with the number of times it has been used. Note that only Activation Key Administrators can see this list. At this point, you may associate child channels and groups with the key so that systems registered with it automatically subscribe to them.

To change information about a key, such as the channels or groups, click its description in the key list, make your modifications in the appropriate tab, and click the **Update Activation Key** button. To disassociate channels and groups from a key, deselect them in their respective menus by **Ctrl**-clicking their highlighted names. To remove a key entirely, click the **delete key** link in the top-right corner of the edit page.

A system may be set to subscribe to a base channel during registration with an activation key. However, if the activation key specifies a base channel that is not compatible with the operating system of the systems, the registration fails. For example, a Red Hat Enterprise Linux 6 for x86\_64 system cannot register with an Activation Key that specifies a Red Hat Enterprise Linux 5 for x86\_64 base channel. A system is always allowed to subscribe to a custom base channel.

To disable system activations with a key, deselect the corresponding checkbox under the **Enabled** column in the key list. The key can be re-enabled by selecting the checkbox. After making these changes, click the **Update Activation Keys** button on the bottom right-hand corner of the page.

### Procedure 3.9. Using Multiple Activation Keys at Once

The provisioning entitlement allows multiple activation keys at the command line or in a single kickstart profile. This allows the organization to aggregate the aspects of various keys without recreating a new key specific to the desired systems, simplifying the registration and kickstart processes while slowing the growth of the key list.

Registering with multiple activation keys requires some caution; conflicts between some values cause registration to fail. Conflicts in the following values do not cause registration to fail: software packages, software child channels, and config channels. Conflicts in the remaining properties are resolved in the following manner:

- ✱ base software channels - registration fails
  - ✱ entitlements - registration fails
  - ✱ enable config flag - configuration management is set
1. Create multiple individual activation keys. See [Section 3.1.3, “Using Activation Keys”](#) for instructions.
  2. On the command line, include all the activation keys, separated by a comma, as an option in **rhgreg\_ks**:

```
# rhgreg_ks --  
activationkey=activationkey1,activationkey2,activationkey3
```

This may also be added into the kickstart profile within the **Post** tab of the **Kickstart Details** page. See [Section 3.1.2.3, “Kickstart Profiles”](#) for instructions.



#### Warning

Do not use system-specific activation keys along with other activation keys; registration fails in this event.

### 3.1.4. Using Cobbler

Red Hat Satellite features the *Cobbler* server that allows administrators to centralize their system installation and provisioning infrastructure. Cobbler is an installation server that provides various methods of performing unattended system installations, whether it be server, workstation, or guest systems in a full or para-virtualized setup.

Cobbler has several tools to assist in pre-installation guidance, kickstart file management, installation environment management, and more.





## Warning

This section explains supported uses of Cobbler, which include:

- ✦ Kickstart template creation and management using the Cheetah template engine and Kickstart Snippets
- ✦ Virtual machine guest installation automation with the **koan** client-side tool
- ✦ Installation environment analysis using the **cobbler check** command
- ✦ Building installation ISOs with PXE-like menus using the **cobbler buildiso** command for Satellite systems with x86\_64 architecture.

Red Hat supports only those Cobbler functions that are either listed within our formal documentation, exposed within the Satellite Web UI and API, or articles listed in Red Hat Customer Portal under the subject of *Supported Cobbler Usage*.

### 3.1.4.1. Cobbler Requirements

To use Cobbler as a PXE boot server, check the following guidelines:

- ✦ To use Cobbler for system installation with PXE, install and configure the **tftp-server** package.
- ✦ To use Cobbler to PXE boot systems for installation, your Satellite server requires either the ability to act as a DHCP server for Cobbler PXE booting or access to your network DHCP server. Edit **/etc/dhcp.conf** to change **next-server** to the hostname or IP address of your Cobbler server.
- ✦ The latest **cobbler-loaders** package needs to be installed. This is to ensure that the **pxelinux.0** boot image is installed and available on the Satellite before PXE booting. To install the latest version:

```
# yum install cobbler-loaders
```

#### 3.1.4.1.1. Configuring Cobbler with DHCP

Cobbler supports bare metal kickstart installation of systems configured to perform network boots using a PXE boot server. To properly implement a Cobbler installation server, administrators need to either have administrative access to the network's DHCP server or implement DHCP on the Cobbler server itself.

#### Procedure 3.10. Configuring an Existing DHCP Server

If you have a DHCP server deployed on another system on the network, you will need administrative access to the DHCP server in order to edit the DHCP configuration file so that it points to the Cobbler server and PXE boot image.

1. Log in as root on the DHCP server.
2. Edit the **/etc/dhcpd.conf** file and append a new class with options for performing PXE boot installation. For example:

```
allow booting;
allow bootp;
class "PXE" {
  match if substring(option vendor-class-identifier, 0, 9) =
```

```
"PXEClient";
next-server 192.168.2.1;
filename "pxelinux.0";
}
```

Following each action step-by-step in the above example:

- a. The administrator enables network booting with the **bootp** protocol.
- b. The administrator then creates a class called **PXE**. A system that is configured to have PXE first in its boot priority will identify itself as a **PXEClient**.
- c. The DHCP server directs the system to the Cobbler server at 192.168.2.1.
- d. Finally, the DHCP server retrieves the **pxelinux.0** bootloader file.

#### 3.1.4.1.2. Configuring Xinetd and TFTP for Cobbler

Xinetd is a daemon that manages a suite of services, including TFTP, the FTP server used for transferring the boot image to a PXE client. To configure TFTP:

1. Log in as root.
2. Edit the `/etc/xinetd.d/tftp` and change the option:

```
disable = yes
```

to

```
disable = no
```

3. Start the xinetd service:

```
# chkconfig --level 345 xinetd on
# /sbin/service xinetd start
```

#### 3.1.4.1.3. Configuring SELinux and IPTables for Cobbler Support

Red Hat Enterprise Linux is installed with SELinux support and a secure firewall. Both are enabled by default. To properly configure a Red Hat Enterprise Linux server to use Cobbler, you must first configure these system and network safeguards to allow connections to and from the Cobbler Server.

### Procedure 3.11. Enabling Cobbler Support in SELinux

To enable SELinux for Cobbler support:

1. Log in as root.
2. Set the SELinux Boolean to allow HTTPD web service components:

```
# setsebool -P httpd_can_network_connect true
```

The **-P** switch is essential, as it enables HTTPD connection persistently across all system reboots.

### Procedure 3.12. IPTables Configuration

Once SELinux has been configured, configure IPTables to allow incoming and outgoing network traffic on the Cobbler server.

1. Log in as root.
2. Add the following rules to the existing IPTables firewall rule sets to open the Cobbler-related ports:

✎ For TFTP:

```
# /sbin/iptables -A INPUT -m state --state NEW -m tcp -p tcp --dport
69 -j ACCEPT
# /sbin/iptables -A INPUT -m state --state NEW -m udp -p udp --dport
69 -j ACCEPT
```

✎ For HTTPD:

```
# /sbin/iptables -A INPUT -m state --state NEW -m tcp -p tcp --dport
80 -j ACCEPT
# /sbin/iptables -A INPUT -m state --state NEW -m tcp -p tcp --dport
443 -j ACCEPT
```

✎ For Cobbler and Koan XMLRPC:

```
# /sbin/iptables -A INPUT -m state --state NEW -m tcp -p tcp --dport
25151 -j ACCEPT
```

3. Save the firewall configuration:

```
# /sbin/iptables-save > /etc/sysconfig/iptables
```

### 3.1.4.2. Configuring Cobbler with `/etc/cobbler/settings`

Most Cobbler configuration is done within the `/etc/cobbler/settings` file. The file contains several configurable settings, and offers detailed explanations for each setting regarding how it affects the functionality of Cobbler and whether it is recommended for users to change the setting for their environment.

Most of the settings can be left default and Cobbler will run as intended. For more information about configuring Cobbler settings, consult the `/etc/cobbler/settings` file, which documents each setting in detail.

### 3.1.4.3. Syncing and Starting the Cobbler Service

1. Before starting the cobbler service, run a check on the cobbler service to make sure that all the prerequisites are configured according to the organization's needs:

```
# cobbler check
```

2. Start the Satellite server with the following command:

```
# /usr/sbin/rhn-satellite start
```



## Warning

Do not start or stop the **cobblerd** service independent of the Satellite service, as doing so may cause errors and other issues.

Always use **/usr/sbin/rhn-satellite** to start or stop Red Hat Satellite.

### 3.1.4.4. Adding a Distribution to Cobbler

If all prerequisites have been met and Cobbler is now running, you can now begin adding a distribution to the Cobbler.

Use **cobbler** to create a distribution with the following syntax:

```
# cobbler distro add --name=string --kernel=path --initrd=path
```

The **--name=*string*** switch is a label used to differentiate one distro choice from another (for example, **rhel5server**)

The **--kernel=*path*** switch specifies the path to the kernel image file

The **--initrd=*path*** switch specifies the path to the initial ramdisk (initrd) image file.

### 3.1.4.5. Adding a Profile to Cobbler

Once you have added a distribution to Cobbler, you can then add profiles to Cobbler.

Cobbler profiles associate a distribution to additional options, like kickstart files. Profiles are the core unit of provisioning and there must be at least one Cobbler profile for every distribution added. For example, two profiles might be created for a web server and a desktop configuration. While both profiles use the same distro, the profiles are for different installation types.

For information about creating and configuring kickstart profiles from the Red Hat Satellite interface, see [Section 3.1.2.3, “Kickstart Profiles”](#).

Use **cobbler** to create profiles with the following syntax:

```
# cobbler profile add --name=string --distro=string [--kickstart=url] [--virt-file-size=gigabytes] [--virt-ram=megabytes]
```

The **--name=*string*** is the unique label for the profile, such as **rhel5webserver** or **rhel4workstation**.

The **--distro=*string*** switch specifies the distribution that will be used for this particular profile. Distributions were added in [Section 3.1.4.4, “Adding a Distribution to Cobbler”](#).

The **--kickstart=*url*** option specifies the location of the kickstart file (if available).

The **--virt-file-size=*gigabytes*** option allows you to set the size of the virtual guest file image. The default is 5 gigabytes if left unspecified.

The **--virt-ram=*megabytes*** option specifies how many megabytes of physical RAM that a virtual guest system can consume. The default is 512 megabytes if left unspecified.

### 3.1.4.6. Adding a System to Cobbler

Once the distributions and profiles for Cobbler have been created, add systems to Cobbler. System records map a piece of hardware on a client with the cobbler profile assigned to run on it.



#### Note

If you are provisioning via **koan** and PXE menus alone, it is not required to create system records. However, system records are useful when:

- ✧ System-specific kickstart templating is required
- ✧ A specific system always receives specific set of content
- ✧ There is a specific role intended for a specific client

The following command adds a system to the Cobbler configuration:

```
# cobbler system add --name=string --profile=string --mac-  
address=AA:BB:CC:DD:EE:FF
```

The **--name=string** is the unique label for the system, such as **engineeringserver** or **frontofficeworkstation**.

The **--profile=string** specifies one of the profile names added in [Section 3.1.4.5, “Adding a Profile to Cobbler”](#).

The **--mac-address=AA:BB:CC:DD:EE:FF** option allows systems with the specified MAC address to automatically be provisioned to the profile associated with the system record if they are being kickstarted.

For more options, such as setting hostname or IP addresses, see the Cobbler manpage by typing **man cobbler** at a shell prompt.



#### Important

A system with the name **default** has a special function. It sets all undefined systems to use a specific profile through PXE. Without a **default** system, PXE falls to local boot for unconfigured systems. Include only **default** as the name and the profile, such as:

```
# cobbler system add --name=default --profile=rhel5webserver
```

### 3.1.4.7. Using Cobbler Templates

Within the Red Hat Satellite web interface, there are facilities to create variables for use with kickstart distributions and profiles.

Kickstart variables are a part of an infrastructural change in Satellite to support *templating* in kickstart files. In the context of kickstart files, templates are files that hold descriptions used to build actual kickstart files, rather than creating specific kickstarts.

These templates are then shared by various profiles and systems that have their own variables and corresponding values. These variables modify the templates and software called a *template engine* parses the template and variable data into a usable kickstart file. Cobbler uses an advanced template engine called *Cheetah* that provides support for templates, variables, and snippets.

Advantages of using templates include:

- ✧ Robust features that allow administrators to create and manage large amounts of profiles or systems without duplication of effort or manually creating kickstarts for every unique situation
- ✧ While templates can become complex and involve loops, conditionals and other enhanced features and syntax, they can also be used simply to make kickstart files without such complexity

#### 3.1.4.7.1. Using Templates

Kickstart templates can have static values for certain common items such as PXE image filenames, subnet addresses, and common paths such as `/etc/sysconfig/network-scripts/`. However, where templates differ from standard kickstart files are in their use of variables.

For example, a standard kickstart file may have a networking passage that looks similar to the following:

```
network --device=eth0 --bootproto=static --ip=192.168.100.24 --  
netmask=255.255.255.0 --gateway=192.168.100.1 --nameserver=192.168.100.2
```

However, in a kickstart template file, the networking passage may look similar to the following:

```
network --device=$net_dev --bootproto=static --ip=$ip_addr --  
netmask=255.255.255.0 --gateway=$my_gateway --nameserver=$my_nameserver
```

These variables will be substituted with the values set in your kickstart profile variables or in your system detail variables. If the same variables are defined in both the profile and the system detail, then the system variable takes precedence.

#### 3.1.4.7.2. Kickstart Snippets

If you have common configurations that are the same across all kickstart templates and profiles, you can utilize the *Snippets* feature of Cobbler to take advantage of code reuse.

Kickstart snippets are sections of kickstart code that can be called by a `$$SNIPPET()` function that will be parsed by Cobbler and substitute that function call with the contents of the snippet.

For example, if you had a common hard drive partition configuration for all servers, such as:

```
clearpart --all  
part /boot --fstype ext3 --size=150 --asprimary  
part / --fstype ext3 --size=40000 --asprimary  
part swap --recommended  
  
part pv.00 --size=1 --grow  
  
volgroup vg00 pv.00  
logvol /var --name=var vgroup=vg00 --fstype ext3 --size=5000
```

You could take that snippet, save it to a file (such as `my_partition`), and place the file in `/var/lib/cobbler/snippets/` so that Cobbler can access it.

You can then use the snippet by using the **\$SNIPPET()** function in your kickstart templates. For example:

```
$SNIPPET('my_partition')
```

Wherever you invoke that function, the Cheetah parser will substitute the function with the snippet of code contained in the **my\_partition** file.

### 3.1.4.8. Using Koan

Whether you are provisioning guests on a virtual machine or reinstalling a new distribution on a running system, koan works in conjunction with Cobbler to provision systems.

#### 3.1.4.8.1. Using Koan to Provision Virtual Systems

If you have created a virtual machine profile as documented in [Section 3.1.4.5, “Adding a Profile to Cobbler”](#), you can use **koan** to initiate the installation of a virtual guest on a system.

For example, say you've created a Cobbler profile like the following:

```
# cobbler add profile --name=virtualfileservr --distro=rhel-i386-server-5 -  
-virt-file-size=20 --virt-ram=1000
```

This profile is for a fileserver running Red Hat Enterprise Linux 5 with a 20GB guest image size and allocated 1GB of system RAM.

To find the name of the virtual guest system profile, run the following with **koan**:

```
# koan --server=hostname --list=profiles
```

This command lists all of the available profiles created with **cobbler profile add**.

Then, begin the process of creating the image file and starting the installation of the virtual guest system:

```
# koan --virt --server=cobbler-server.example.com --  
profile=virtualfileservr --virtname=marketingfileserver
```

The command specifies that a virtual guest system be created from the Cobbler server (hostname cobbler-server.example.com) using the **virtualfileservr** profile. The **virtname** option specifies a label for the virtual guest, which by default is labeled with the system's MAC address.

Once installation of the virtual guest is complete, it can be used as any other virtual guest system.

#### 3.1.4.8.2. Using Koan to Re-install Running Systems

There may be instances where you need to re-install a machine with another operating system while it is still running. **koan** can help you by destructively replacing a running system with a new installation from the available Cobbler profiles.

To replace a running system and install a new one, run the following command *on the system itself*:

```
# koan --replace-self --server=hostname --profile=name
```

This command, when executed on the running system to be replaced, will start the provisioning process and replace its own system using the profile in **--profile=name** on the Cobbler server specified in **--server=hostname**.

### 3.1.4.9. Building ISOs with Cobbler

Some environments might lack PXE support. To overcome this, the **cobbler buildiso** command provides functionality to create a boot ISO image containing a set of distributions and kernels, and a menu similar to PXE network installations.

Define the name and output location of the boot ISO using the **--iso** option.

```
# cobbler buildiso --iso=/path/to/boot.iso
```

The boot ISO includes all profiles and systems by default. Limit these profiles and systems using the **--profiles** and **--systems** options.

```
# cobbler buildiso --systems="system1,system2,system3" \
  --profiles="profile1,profile2,profile3"
```



#### Note

Building ISOs with the cobbler buildiso command is supported for all architectures except the s390x architecture.



#### Note

The **cobbler buildiso --standalone** option is not supported with Red Hat provided kickstart trees. The standalone option is used for provisioning machines that have no network connectivity to the cobbler server, but all of the additional functionality Satellite provisioning provides requires a network connection to the Satellite. If you need to install Red Hat Enterprise Linux on a machine with no network connectivity, install Red Hat Enterprise Linux by downloading the ISO image.

## 3.2. Provisioning Through Red Hat Satellite Proxy

Provisioning can also be achieved using an Red Hat Satellite Proxy that has been installed and registered to Red Hat Satellite .

1. When provisioning a virtual guest or doing a reprovisioning of a system, select the desired proxy from the **Select Satellite Proxy** drop down box.
2. For a bare metal installation, replace the Red Hat Satellite's fully qualified domain name (FQDN) with that of the proxy's FQDN. For example, if the URL to the kickstart file is:

```
http://satellite.example.com/ks/cfg/org/1/label/myprofile
```

3. Kickstart through to the Red Hat Satellite Proxy:

```
http://proxy.example.com/ks/cfg/org/1/label/myprofile
```

## 3.3. Provisioning Virtualized Guests



Virtual Guest Provisioning is supported in Red Hat Satellite 5 using the following virtualization technologies:

- ✧ KVM Virtualized Guest
- ✧ Xen Fully-Virtualized Guest
- ✧ Xen Para-Virtualized Guest

### Procedure 3.13. Provisioning a Virtualized Guest

1. Check that the host system has a **Virtualization** or **Virtualization Platform** system entitlement.
2. On the **Systems** page, select the appropriate virtual host, then select **Virtualization** → **Provisioning**. Select the appropriate kickstart profile and enter a guest name.
3. To configure additional parameters such as guest memory and CPU usage, click the **Advanced Configuration** button. The following options can be configured:
  - ✧ Network: static or DHCP
  - ✧ Kernel options
  - ✧ Package profile synchronization: when the kickstart finishes the system will synchronize its package profile to that of another system or a stored profile
  - ✧ Memory allocation: RAM (Defaults to 512MB)
  - ✧ Virtual disk size
  - ✧ Virtual CPUs (Defaults to 1)
  - ✧ Virtual bridge: The networking bridge used for the install. Defaults to **xenbr0** for Xen provisioning, and **virbr0** for KVM.



#### Note

The **virbr0** networking bridge will not allow outside networking. If outside networking is required, configure the host to create an actual bridge instead. However, **xenbr0** is an actual bridge, and it is recommended for use if possible.

- ✧ Virtual storage path: Path to either a file, LVM Logical Volume, directory, or block device with which to store the guest's disk information, such as **/dev/sdb**, **/dev/LogVol100/mydisk**, **VolGroup00**, or **/var/lib/xen/images/myDisk**.
4. Click **Schedule Kickstart and Finish**.

## 3.4. Kickstarting through Cloned or Custom Channels

There are two methods to install a system from cloned or custom channels:

1. Kickstart a system from a Red Hat Enterprise Linux channel with **@Base** mentioned under **%packages**. Under the same kickstart, specify an activation key to subscribe to the cloned or custom channels. Also mention the packages to install from the cloned or custom channel under packages section of the activation key. Kickstarting a system through this method initially installs from Red Hat

Enterprise Linux Base channel with minimal packages and then registers the system to the cloned channel on the Satellite 5 server using the activation key. After registration, update the system through the kickstart.

2. Alternatively, if you aim to kickstart the system directly from your cloned or custom channel, create a distribution tree for the channel. An example procedure is outlined below.

### Procedure 3.14. Creating a Distribution Tree for a Cloned Channel

1. Copy your cloned channel. For example, for Red Hat Enterprise Linux 6.6:

```
# cd /var/satellite/rhn/kickstart/
# mkdir custom-distro-rhel-6.6
# cd custom-distro-rhel-6.6
# cp -rpv ../ks-rhel-x86_64-server-6-6.6/* .
```

2. Set the permissions for all files in the tree to be 644 with **apache:apache** as the ownership.

```
# find . -type f -print0 | xargs -0 chmod 644
# find . -type f -print0 | xargs -0 chown apache:apache
```

Set the permissions for all directories in the tree to be 755 with **apache:root** as the ownership.

```
# find . -type d -print0 | xargs -0 chmod 755
# find . -type d -print0 | xargs -0 chown apache:root
```

3. Login to the Satellite 5 server web UI and navigate to **Systems** → **Kickstart** → **Distribution** → **create new distribution**
4. Set the following values:

- ✧ **Distribution Label:** custom-distro-rhel-6.6
- ✧ **Tree Path:** /var/satellite/rhn/kickstart/custom-distro-rhel-6.6/
- ✧ **Installer Generation:** 6

When complete, click **Create Kickstart Distribution**.

This creates a kickstart profile with the base channel set to your cloned channel and kickstart tree.



#### Note

For Red Hat Enterprise Linux 6 systems, registering and subscribing kickstarted cloned channel systems requires an activation key that uses the cloned channel as the base channel. Add this activation key to the cloned channel kickstart. The newly kickstarted system registers and subscribes to same cloned channel from which it kickstarted.

## 3.5. Reprovisioning

Reinstalling an existing system is referred to as *reprovisioning*. Reprovisioning can be done using the Red Hat Satellite web interface, and the system will use the same system profile that it had before it was reprovisioned. This will preserve information and settings about the system such as entitlements, system groups and subscribed channels.

A system being reprovisioned with a different major Red Hat Enterprise Linux version as the one currently in the system will preserve the system profile, entitlements and server groups. However, because the base channel changed, all child channels are lost.

If the kickstart profile chosen for reprovisioning contains an activation key, reprovisioning will execute all the instructions in the activation key without removing any existing relationships the system has. The system will retain its system profile, channel subscriptions, organization and group affiliations. The system will just be added to any additional organizations, groups or any additional packages the activation key contains.

Reprovisioning can be scheduled from the **Provisioning** tab while viewing a system. To configure additional options, go to the **Advanced Configurations** page, which allows the configuration of details such as kernel options, networking information, and package profile synchronization. The **Kernel Options** section provides access to the kernel options used during kickstart and **Post Kernel Options** are the kernel options that will be used after the kickstart is complete and the system is booting for the first time.

### Example 3.2. Configuring Kernel Options and Post Kernel Options

This example describes the difference between kernel options and post kernel options in the reprovisioning configuration process.

To establish a VNC connection to monitor the kickstart remotely, include **vnc vncpassword=PASSWORD** in the **Kernel Options** line.

To boot the kernel of the resulting system with the **noapic** kernel option, add **noapic** to the **Post Kernel Options** line.

### Procedure 3.15. File Preservation

The *File Preservation* feature can be used to keep files from being lost during a reprovisioning. This feature stores files temporarily during the kickstart and restores them after the reprovisioning is complete.



#### Note

File preservation lists are only available on wizard-style kickstarts, and can only be used during reprovisioning.

1. Go to **Systems** → **Kickstart** → **File Preservation** → **create new file preservation list** and create a list of files to preserve.
2. Go to **Systems** → **Kickstart** → **Profiles** and associate the file preservation list with a kickstart by selecting the desired profile.
3. Go to **System Details** → **File Preservation** and select the file preservation list.

## 3.6. Provisioning Rollbacks with Snapshots

Snapshots are taken when an action takes place on a system. These snapshots identify groups, channels, packages, and configuration files. Satellite administrators with provisioning entitlements have the ability to roll

back the package profile, local configuration files and RHN settings of systems through snapshot rollbacks.



## Note

While snapshot rollbacks support the ability to revert *certain* changes to the system, this is not applicable to every scenario. For example, you can roll back a set of RPM packages, but rolling back across multiple update levels is not supported.

The **Snapshots** subtab is located within **System Details** → **Provisioning** → **Snapshots**. The **Snapshots** subtab lists all the snapshots for the system, including:

- ✧ The reason the snapshot was taken
- ✧ The time it was taken
- ✧ The number of tags applied to each snapshot

### Procedure 3.16. Performing a Snapshot Rollback

To revert to a previous configuration:

1. Click on **Systems**.
2. Click on the system that requires a roll back.
3. Choose the **Provisioning** → **Snapshots** tab.
4. Click the **Reason** of the snapshot taken and review the potential changes on the provided subtabs, starting with **Rollback**.
5. Check each subtab for the specific changes that will be made to the system during the rollback:
  - ✧ group memberships
  - ✧ channel subscriptions
  - ✧ installed packages
  - ✧ configuration channel subscriptions
  - ✧ configuration files
  - ✧ snapshot tags
6. When satisfied with the reversion, return to the **Rollback** subtab and click the **Rollback to Snapshot** button. To see the list again, click Return to snapshot list.

### 3.6.1. Using Snapshot Tags

Snapshot tags are a means to add meaningful descriptions to system snapshots. They can be used to indicate known working configurations, successful upgrades or other milestones.

### Procedure 3.17. Creating Snapshot Tags

On the **Snapshot** tab:

1. Click **create new system tag**.

2. Enter a descriptive term in the **Tag name** field.

3. Click **Tag Current Snapshot**.

To use the tags, click the tag name in the **Snapshot Tags** list.

## Chapter 4. System Management

Red Hat Satellite provides system-level support and management of Red Hat Systems and networks of systems. This chapter will discuss systems and how to organize these systems into functional groups inside the organization for effective management.

### 4.1. Registering Systems to Satellite

Systems are client machines that requests package updates from Red Hat Satellite. These systems can be physical machines or virtualized systems that have been configured to register and receive updates from the Satellite. Registering systems to Satellite is an important step, as the client system will, by default, register to Red Hat Network, instead of the organization's Satellite. For information about how to register, see the relevant chapter on registering clients to the Satellite server in the *Red Hat Satellite Client Configuration Guide*.

#### 4.1.1. Using Red Hat Network Bootstrap to Register a System

Red Hat Network provides a tool that automates much of the manual reconfiguration for registering systems, this tool is called **Red Hat Network Bootstrap**. **Red Hat Network Bootstrap** plays an integral role in the **Red Hat Satellite Server Installation Program**, enabling generation of the bootstrap script during installation.

Red Hat Satellite Proxy Server administrators and administrators with updated Satellite settings require a bootstrap tool that can be used independently. **Red Hat Network Bootstrap**, invoked with the command `/usr/bin/rhn-bootstrap`, serves that purpose and comes installed by default on both Red Hat Satellite Server and Red Hat Satellite Proxy Server.

If used correctly, the script this tool generates can be run from any client system to conduct the following tasks:

- » Redirect client applications to the Red Hat Satellite Proxy or Satellite
- » Import custom GPG keys
- » Install SSL certificates
- » Register the system to Red Hat Network and particular system groups and channels with the help of activation keys
- » Perform miscellaneous post-configuration activities, including updating packages, performing reboots, and altering Red Hat Network configuration



#### Warning

There are inherent risks to using a script to conduct configuration. Security tools such as SSL certificates are installed by the script itself; therefore they do not yet exist on the systems and cannot be used to process transactions. This allows for the possibility of someone impersonating the Satellite and transmitting bad data. This is mitigated by the fact that virtually all Satellites and client systems operate behind customer firewalls and are restricted from outside traffic. Registration is conducted via SSL and is therefore protected.

The bootstrap script **bootstrap.sh** is automatically placed in the **/var/www/html/pub/bootstrap/** directory of the Red Hat Network Server. From there it can be downloaded and run on all client systems. Note that some preparation and post-generation editing is required, as identified in the following sections. See [Section 4.1.1.4, “Configuring Red Hat Network Bootstrap Options”](#) for an example script.

#### 4.1.1.1. Preparing for Red Hat Network Bootstrap Installation

Since **Red Hat Network Bootstrap (rhn-bootstrap)** depends on other components of the Red Hat Network infrastructure to properly configure client systems, those components must be prepared before script generation. The following list identifies initial measures:

- Generate activation keys to be called by the script(s). Activation keys can be used to register Red Hat Enterprise Linux systems, entitle them to an Red Hat Network service level, and subscribe them to specific channels and system groups, all in one action. Note that the organizational account must have Management entitlements available to use an activation key, while inclusion of multiple activation keys at once requires Provisioning entitlements. Generate activation keys through the **Activation Keys** page within the **Systems** category of the Red Hat Satellite website (either the central Red Hat Network Servers for Proxy or the fully qualified domain name of the Satellite).
- Red Hat recommends RPMs be signed by a custom GNU Privacy Guard (GPG) key. Make the key available so that it can be referred to from the script. Generate the key as described in the *Red Hat Satellite Reference Guide* and place the key in the **/var/www/html/pub/** directory of the Red Hat Satellite Server. See the *Importing Custom GPG Keys* section in the *Red Hat Satellite Reference Guide*.
- To deploy the CA SSL public certificate through the script, have the certificate or the package (RPM) containing that certificate available on that Red Hat Network Server and include it during script generation with the **--ssl-cert** option. See the SSL Infrastructure section of the *Client Configuration Guide* for details.
- Have the values ready to develop one or many bootstrap scripts, depending on the variety of systems to be reconfigured. Since **Red Hat Network Bootstrap** provides a full set of reconfiguration options, use it to generate different bootstrap scripts to accommodate each type of system. For instance, **bootstrap-web-servers.sh** might be used to reconfigure the Web servers, while **bootstrap-app-servers.sh** can handle the application servers. See [Section 4.1.1.4, “Configuring Red Hat Network Bootstrap Options”](#) for the complete list.

#### 4.1.1.2. Generating Bootstrap Scripts

Now that all of the necessary components are in place, use **Red Hat Network Bootstrap** to generate the required scripts. Log into your Red Hat Satellite Server or Red Hat Satellite Proxy Server as root and issue the **rhn-bootstrap** command followed by the desired options and values. If no options are included, a **bootstrap.sh** file is created in the **bootstrap/** subdirectory that contains the essential values derived from the server, including hostname, the SSL certificate, if it exists, SSL and GPG settings, and a call for the **client-config-overrides.txt** file.

At a minimum, Red Hat strongly recommends the scripts also accommodate activation keys, GPG keys, and advanced configuration options in the following manner:

- Use the **--activation-keys** option to include keys, taking into account the entitlement requirements identified in [Section 4.1.1.1, “Preparing for Red Hat Network Bootstrap Installation”](#).
- Use the **--gpg-key** option to identify the key path and filename during script generation. Otherwise, use the **--no-gpg** option to turn off this verification on client systems. Red Hat recommends retaining this security measure.
- Include the **--allow-config-actions** flag to enable remote configuration management on all client systems touched by the script. This feature is useful in reconfiguring multiple systems simultaneously.

- ✱ Include the **--allow-remote-commands** flag to enable remote script use on all client systems. Like configuration management, this feature aids in reconfiguring multiple systems.

When done, the command will look something like this:

```
# rhn-bootstrap --activation-keys KEY1,KEY2 \
  --gpg-key /var/www/html/pub/MY_CORPORATE_PUBLIC_KEY \
  --allow-config-actions \
  --allow-remote-commands
```

Remember to include the actual key names. See [Section 4.1.1.4, “Configuring Red Hat Network Bootstrap Options”](#) for the complete list of options.

### 4.1.1.3. Using the Red Hat Network Bootstrap Script

Once the script has been prepared for use, it is now ready to be run. Log into the Red Hat Satellite Server or Red Hat Satellite Proxy Server, navigate to the **/var/www/html/pub/bootstrap/** directory and run the following command, altering the hostname and name of the script as needed to suit the system type:

```
# cat bootstrap-EDITED-NAME.sh | ssh root@CLIENT_MACHINE1 /bin/bash
```

A less secure alternative is to use either **wget** or **curl** to retrieve and run the script from every client system. Log into each client machine and issue the following command, altering script and hostname accordingly:

```
# wget -qO - \
  https://your-satellite.example.com/pub/bootstrap/bootstrap-EDITED-
NAME.sh \
  | /bin/bash
```

Or with **curl**:

```
# curl -Sks \
  https://your-satellite.example.com/pub/bootstrap/bootstrap-EDITED-
NAME.sh \
  | /bin/bash
```

When this script has been run on each client system, all should be configured to use the Red Hat Network Server.

### 4.1.1.4. Configuring Red Hat Network Bootstrap Options

The **Red Hat Network Bootstrap** offers many command line options for creating client bootstrap scripts. Although descriptions of these options can be found within the following table, ensure that they are available in the version of the tool installed on the Red Hat Network Server by issuing the command **rhn-bootstrap -h** or reviewing its man page.

**Table 4.1. Red Hat Network Bootstrap Options**

Option	Description
<b>-h, --help</b>	Display the help screen with a list of options specific to generating the bootstrap script.



Option	Description
<code>--activation-keys=ACTIVATION_KEYS</code>	Activation key(s) with multiple entries separated by a comma and no space.
<code>--overrides=OVERRIDES</code>	Configuration overrides filename. The default is <code>client-config-overrides.txt</code> .
<code>--script=SCRIPT</code>	The bootstrap script filename. The default is <code>bootstrap.sh</code> .
<code>--hostname=HOSTNAME</code>	The fully qualified domain name (FQDN) of the server to which client systems will connect.
<code>--ssl-cert=SSL_CERT</code>	The path to the organization's public SSL certificate, either a package or a raw certificate. It will be copied to the <code>--pub-tree</code> option. A value of <code>""</code> will force a search of <code>--pub-tree</code> .
<code>--gpg-key=GPG_KEY</code>	The path to the organization's public GPG key, if used. It will be copied to the location specified by the <code>--pub-tree</code> option.
<code>--http-proxy=HTTP_PROXY</code>	The HTTP proxy setting for the client systems in the form <code>hostname:port</code> . A value of <code>""</code> disables this setting.
<code>--http-proxy-username=HTTP_PROXY_USERNAME</code>	If using an authenticating HTTP proxy, specify a username. A value of <code>""</code> disables this setting.
<code>--http-proxy-password=HTTP_PROXY_PASSWORD</code>	If using an authenticating HTTP proxy, specify a password.
<code>--allow-config-actions</code>	Boolean; including this option sets the system to allow all configuration actions via Red Hat Network. This requires installing certain <code>rhncfg-*</code> packages, possibly through an activation key.
<code>--allow-remote-commands</code>	Boolean; including this option sets the system to allow arbitrary remote commands via Red Hat Network. This requires installing certain <code>rhncfg-*</code> packages, possibly through an activation key.
<code>--no-ssl</code>	<i>Not recommended</i> - Boolean; including this option turns SSL off on the client system.
<code>--no-gpg</code>	<i>Not recommended</i> - Boolean; including this option turns GPG checking off on the client system.
<code>--pub-tree=PUB_TREE</code>	<i>Change not recommended</i> - The public directory tree where the CA SSL certificate and package will land; the bootstrap directory and scripts. The default is <code>/var/www/html/pub/</code> .
<code>--force</code>	<i>Not recommended</i> - Boolean; including this option forces bootstrap script generation despite warnings.

Option	Description
<b>-v, --verbose</b>	Display verbose messaging. Accumulative; <b>-vvv</b> causes extremely verbose messaging.

#### 4.1.1.5. Manually Scripting the Red Hat Network Bootstrap Configuration

Note that this section provides an alternative to using **Red Hat Network Bootstrap** to generate the bootstrap script. Below are instructions that should assist in creating a bootstrap script from scratch.

All of the initial techniques have shared a common theme: the deployment of necessary files in a centralized location to be retrieved and installed using simple, scriptable commands run on each client. In this section, we explore putting all of these pieces together to create a single script that can be invoked by any system in your organization.

By combining all of the commands learned in the previous section and putting them in the most sensible order, we are able to produce the script below:

```
# Reconfigure the clients to talk to the correct server.

perl -p -i -e 's/s/www\.rhns\.redhat\.com/proxy-or-sat\.example\.com/g' \
/etc/sysconfig/rhn/rhn_register \
/etc/sysconfig/rhn/up2date

# Install the SSL client certificate for your company's
# Red Hat Satellite Server or Red Hat Network Proxy Server.
rpm -Uvh http://proxy-or-sat.example.com/pub/rhn-org-trusted-ssl-cert-
*.noarch.rpm

# Reconfigure the clients to use the new SSL certificate.
perl -p -i -e 's/^sslCA/#sslCA/g;' \
/etc/sysconfig/rhn/up2date /etc/sysconfig/rhn/rhn_register
echo "sslCACert=/usr/share/rhn/RHN-ORG-TRUSTED-SSL-CERT" \
>> /etc/sysconfig/rhn/up2date
echo "sslCACert=/usr/share/rhn/RHN-ORG-TRUSTED-SSL-CERT" \
>> /etc/sysconfig/rhn/rhn_register

# Download the GPG key needed to validate custom packages.
wget -O - -q http://proxy-or-sat.example.com.com/pub/YOUR-RPM-GPG-KEY

# Import that GPG key to your GPG keyring.
rpm --import /path/to/YOUR-RPM-GPG-KEY
```

This script comprises a clean and repeatable process that should fully configure any potential Red Hat Satellite client in preparation for registration to a Red Hat Satellite Proxy Server or Red Hat Satellite. Remember, key values, such as the URL of the Red Hat Satellite Server, its public directory, and the actual GPG key must be inserted into the placeholders listed within the script. Also, depending on the environment, additional modifications may be required. Although this script may work nearly verbatim, it should be used as a guide.

Like its components, this script may be centrally located. By placing this script in the **/pub/** directory of the server, running **wget -O-** on it, and piping the output to a shell session, the entire bootstrap process can be run with a single command from each client:

```
# wget -O - http://proxy-or-sat.example.com.com/pub/bootstrap_script | bash
```



### Warning

Running a shell script directly from input piped in over a Web connection obviously has some inherent security risks. Therefore, it is vital to ensure the security of the source server in this instance.

This one-line command may then be invoked across all of the systems on a network. This script may also be a good addition to the `%post` section of an existing kickstart script.

#### 4.1.1.6. Implementing Kickstart

The best time to make configuration changes to a system is when that system is first being built. For customers who already use kickstart effectively, the bootstrapping script is an ideal addition to that process.

Once all of the configuration issues have been resolved, a system may also register with the local Red Hat Network Servers using the **rhnreg\_ks** utility that comes with the **rhn-setup** RPMs. This section discusses the proper use of **rhnreg\_ks** to register systems.

The **rhnreg\_ks** utility uses *activation keys* to register, entitle, and subscribe systems to specified channels in one swift motion. To find out more about activation keys, see the Red Hat Update Agent and Red Hat Network Website sections of the *Red Hat Network Management Reference Guide*.

The following commented kickstart file is an ideal example of how a system can be configured from start to finish using Red Hat Satellite.

```
# Generic 7.2 kickstart for laptops in the Widget Corporation (widgetco)

# Standard kickstart options for a network-based install. For an
# explanation of these options, consult the Red Hat Enterprise Linux
# Customization Guide.

lang en_US
langsupport --default en_US en_US
keyboard defkeymap
network --bootproto dhcp
install
url --url ftp://ftp.widgetco.com/pub/redhat/linux/7.2/en/os/i386
zerombr yes
clearpart --all
part /boot --size 128 --fstype ext3 --ondisk hda
part / --size 2048 --grow --fstype ext3 --ondisk hda
part /backup --size 1024 --fstype ext3 --ondisk hda
part swap --size 512 --ondisk hda
bootloader --location mbr
timezone America/New_York
rootpw --iscrypted $1$78Jnap82Hnd0PsjnC8j3sd2Lna/Hx4.
auth --useshadow --enablemd5 --krb5realm .COM --krb5kdc auth.widgetco.com \
--krb5adminserver auth.widgetco.com
```

```

mouse --emulthree genericps/2
xconfig --card "S3 Savage/MX" --videoram 8192 --resolution 1024x768 \
  --depth 16 --defaultdesktop=GNOME --startxonboot --noprobe \
  --hsync 31.5-48.5 --vsync 40-70

reboot

# Define a standard set of packages. Note: Red Hat Network client
# packages are found in the Base channel. This is quite a minimal
# set of packages

%packages
@ Base
@ Utilities
@ GNOME
@ Laptop Support
@ Dialup Support
@ Software Development
@ Graphics and Image Manipulation
@ Games and Entertainment
@ Sound and Multimedia Support

%post
( # Note that we run the entire %post section as a subshell for logging.

# Use the one-line command for the bootstrap script. Assuming that the
# script has been properly configured, it should prepare the system
# fully for usage of local Red Hat Network Servers.

wget -O- http://proxy-or-sat.example.com/pub/bootstrap_script | /bin/bash

# The following is an example of rhnreg_ks usage, the kickstart
# utility for rhn_register. This demonstrates the usage of the
# --activationkey flag, which describes an activation key. For example,
# this activation key could be set up in the Web interface to join this
# system to the "Laptops" group and the local "Laptop Software"
# channel. Note that this section applies only to Proxy server users, as
# this step is handled by the Satellite bootstrap script.
#
# For more information about activation keys, consult the Red Hat Network
# Management Reference Guide.

/usr/sbin/rhnreg_ks --activationkey=6c933ea74b9b002f3ac7eb99619d3374

# End the subshell and capture any output to a post-install log file.
) 1>/root/post_install.log 2>&1

```

#### 4.1.1.7. Sample Bootstrap Script

The `/var/www/html/pub/bootstrap/bootstrap.sh` script generated by the Red Hat Satellite Server installation program provides the ability to reconfigure client systems to access the Red Hat Satellite Server easily. It is available to both Red Hat Satellite Server and Red Hat Satellite Proxy Server customers through the **RHN Bootstrap** tool. After modifying the script for a particular use, it can be run on each client machine.

Review the sample and its comments, beginning with a hash mark (#), for additional details. Follow the steps in the *Getting Started Guide* to prepare the script for use.

```
#!/bin/bash
echo "Red Hat Satellite Server Client bootstrap script v4.0"

# This file was autogenerated. Minor manual editing of this script (and
# possibly the client-config-overrides.txt file) may be necessary to
# complete
# the bootstrap setup. Once customized, the bootstrap script can be triggered
# in one of two ways (the first is preferred):
#
# (1) centrally, from the RHN Satellite Server via ssh (i.e., from the
#     RHN Satellite Server):
#     cd /var/www/html/pub/bootstrap/
#     cat bootstrap-<edited_name>.sh | ssh root@<client-hostname>
# /bin/bash
#
# ...or...
#
# (2) in a decentralized manner, executed on each client, via wget or
# curl:
#     wget -qO- https://<hostname>/pub/bootstrap/bootstrap-
# <edited_name>.sh | /bin/bash
#     ...or...
#     curl -Sks https://<hostname>/pub/bootstrap/bootstrap-
# <edited_name>.sh | /bin/bash

# SECURITY NOTE:
# Use of these scripts via the two methods discussed is the most expedient
# way to register machines to your RHN Satellite Server. Since "wget" is
# used
# throughout the script to download various files, a "Man-in-the-middle"
# attack is theoretically possible.
#
# The actual registration process is performed securely via SSL, so the
# risk
# is minimized in a sense. This message merely serves as a warning.
# Administrators need to appropriately weigh their concern against the
# relative security of their internal network.

# PROVISIONING/KICKSTART NOTE:
# If provisioning a client, ensure the proper CA SSL public certificate is
# configured properly in the post section of your kickstart profiles (the
# RHN Satellite or hosted web user interface).

# UP2DATE/RHN_REGISTER VERSIONING NOTE:
# This script will not work with very old versions of up2date and
# rhn_register.

echo
echo
echo "MINOR MANUAL EDITING OF THIS FILE MAY BE REQUIRED!"
echo
echo "If this bootstrap script was created during the initial installation"
```

```

echo "of an RHN Satellite, the ACTIVATION_KEYS, and ORG_GPG_KEY values will"
echo "probably *not* be set (see below). If this is the case, please do the"
echo "following:"
echo "  - copy this file to a name specific to its use."
echo "    (e.g., to bootstrap-SOME_NAME.sh - like bootstrap-web-
servers.sh.)"
echo "  - on the website create an activation key or keys for the system(s)
to"
echo "    be registered."
echo "  - edit the values of the VARIABLES below (in this script) as"
echo "    appropriate:"
echo "    - ACTIVATION_KEYS needs to reflect the activation key(s) value(s)"
echo "      from the website. XKEY or XKEY,YKEY"
echo "    - ORG_GPG_KEY needs to be set to the name(s) of the corporate
public"
echo "      GPG key filename(s) (residing in /var/www/html/pub) if
appropriate. XKEY or XKEY,YKEY"
echo
echo "Verify that the script variable settings are correct:"
echo "  - CLIENT_OVERRIDES should be only set differently if a customized"
echo "    client-config-overrides-VER.txt file was created with a
different"
echo "    name."
echo "  - ensure the value of HOSTNAME is correct."
echo "  - ensure the value of ORG_CA_CERT is correct."
echo
echo "Enable this script: comment (with #'s) this block (or, at least just"
echo "the exit below)"
echo
exit 1

# can be edited, but probably correct (unless created during initial
install):
# NOTE: ACTIVATION_KEYS *must* be used to bootstrap a client machine.
ACTIVATION_KEYS=
ORG_GPG_KEY=

# can be edited, but probably correct:
CLIENT_OVERRIDES=client-config-overrides.txt
HOSTNAME=yoursatellite.hostname.com

ORG_CA_CERT=RHN-ORG-TRUSTED-SSL-CERT
ORG_CA_CERT_IS_RPM_YN=0

USING_SSL=1
USING_GPG=1

REGISTER_THIS_BOX=1

ALLOW_CONFIG_ACTIONS=1
ALLOW_REMOTE_COMMANDS=1

FULLY_UPDATE_THIS_BOX=1

# Set if you want to specify profilename for client systems.
# NOTE: Make sure it's set correctly if any external command is used.

```

```

#
# ex. PROFILENAME="foo.example.com" # For specific client system
#     PROFILENAME=`hostname -s`      # Short hostname
#     PROFILENAME=`hostname -f`      # FQDN
PROFILENAME="" # Empty by default to let it be set automatically.

#
# -----
# DO NOT EDIT BEYOND THIS POINT -----
# -----
#
# an idea from Erich Morisse (of Red Hat).
# use either wget *or* curl
# Also check to see if the version on the
# machine supports the insecure mode and format
# command accordingly.

if [ -x /usr/bin/wget ] ; then
    output=`LANG=en_US /usr/bin/wget --no-check-certificate 2>&1`
    error=`echo $output | grep "unrecognized option"`
    if [ -z "$error" ] ; then
        FETCH="/usr/bin/wget -q -r -nd --no-check-certificate"
    else
        FETCH="/usr/bin/wget -q -r -nd"
    fi
else
    if [ -x /usr/bin/curl ] ; then
        output=`LANG=en_US /usr/bin/curl -k 2>>&1`
        error=`echo $output | grep "is unknown"`
        if [ -z "$error" ] ; then
            FETCH="/usr/bin/curl -Sks0"
        else
            FETCH="/usr/bin/curl -Ss0"
        fi
    fi
fi
HTTP_PUB_DIRECTORY=http://${HOSTNAME}/pub
HTTPS_PUB_DIRECTORY=https://${HOSTNAME}/pub
if [ $USING_SSL -eq 0 ] ; then
    HTTPS_PUB_DIRECTORY=${HTTP_PUB_DIRECTORY}
fi

INSTALLER=up2date
if [ -x /usr/bin/zypper ] ; then
    INSTALLER=zypper
elif [ -x /usr/bin/yum ] ; then
    INSTALLER=yum
fi
echo
echo "UPDATING RHN_REGISTER/UP2DATE CONFIGURATION FILES"
echo "-----"

```

```

echo "* downloading necessary files"
echo "  client_config_update.py..."
rm -f client_config_update.py
$FETCH ${HTTPS_PUB_DIRECTORY}/bootstrap/client_config_update.py
echo "  ${CLIENT_OVERRIDES}..."
rm -f ${CLIENT_OVERRIDES}
$FETCH ${HTTPS_PUB_DIRECTORY}/bootstrap/${CLIENT_OVERRIDES}

if [ ! -f "client_config_update.py" ] ; then
    echo "ERROR: client_config_update.py was not downloaded"
    exit 1
fi
if [ ! -f "${CLIENT_OVERRIDES}" ] ; then
    echo "ERROR: ${CLIENT_OVERRIDES} was not downloaded"
    exit 1
fi

echo "* running the update scripts"
if [ -f "/etc/sysconfig/rhn/rhn_register" ] ; then
    echo "  . rhn_register config file"
    /usr/bin/python -u client_config_update.py
    /etc/sysconfig/rhn/rhn_register ${CLIENT_OVERRIDES}
fi
echo "  . up2date config file"
/usr/bin/python -u client_config_update.py /etc/sysconfig/rhn/up2date
${CLIENT_OVERRIDES}

if [ ! -z "$ORG_GPG_KEY" ] ; then
    echo
    echo "* importing organizational GPG key"
    for GPG_KEY in $(echo "$ORG_GPG_KEY" | tr "," " "); do
        rm -f ${GPG_KEY}
        $FETCH ${HTTPS_PUB_DIRECTORY}/${GPG_KEY}
        # get the major version of up2date
        # this will also work for RHEL 5 and systems where no up2date is installed
        res=$(LC_ALL=C rpm -q --queryformat '%{version}' up2date | sed -e
        's/\..*//g')
        if [ "x$res" == "x2" ] ; then
            gpg $(up2date --gpg-flags) --import $GPG_KEY
        else
            rpm --import $GPG_KEY
        fi
    done
fi

echo
echo "* attempting to install corporate public CA cert"
if [ $ORG_CA_CERT_IS_RPM_YN -eq 1 ] ; then
    rpm -Uvh --force --replacefiles --replacepkgs
    ${HTTPS_PUB_DIRECTORY}/${ORG_CA_CERT}
else
    rm -f ${ORG_CA_CERT}
    $FETCH ${HTTPS_PUB_DIRECTORY}/${ORG_CA_CERT}
    mv ${ORG_CA_CERT} /usr/share/rhn/
fi

```



```

if [ "$INSTALLER" == zypper ] ; then
    if [ $ORG_CA_CERT_IS_RPM_YN -eq 1 ] ; then
        # get name from config
        ORG_CA_CERT=$(basename $(sed -n 's/^sslCACert *= */p'
/etc/sysconfig/rhn/up2date))
    fi
    test -e "/etc/ssl/certs/${ORG_CA_CERT}.pem" || {
        test -d "/etc/ssl/certs" || mkdir -p "/etc/ssl/certs"
        ln -s "/usr/share/rhn/${ORG_CA_CERT}"
/etc/ssl/certs/${ORG_CA_CERT}.pem"
    }
    test -x /usr/bin/c_rehash && /usr/bin/c_rehash /etc/ssl/certs/ | grep
"${ORG_CA_CERT}"
fi

echo
echo "REGISTRATION"
echo "-----"
# Should have created an activation key or keys on the RHN Satellite
Server's
# website and edited the value of ACTIVATION_KEYS above.
#
# If you require use of several different activation keys, copy this file
and
# change the string as needed.
#
if [ -z "$ACTIVATION_KEYS" ] ; then
    echo "**** ERROR: in order to bootstrap RHN clients, an activation key or
keys"
    echo "          must be created in the RHN web user interface, and the"
    echo "          corresponding key or keys string (XKEY,YKEY,...) must
be mapped to"
    echo "          the ACTIVATION_KEYS variable of this script."
    exit 1
fi

if [ $REGISTER_THIS_BOX -eq 1 ] ; then
    echo "* registering"
    files=""
    directories=""
    if [ $ALLOW_CONFIG_ACTIONS -eq 1 ] ; then
        for i in "/etc/sysconfig/rhn/allowed-actions
/etc/sysconfig/rhn/allowed-actions/configfiles"; do
            [ -d "$i" ] || (mkdir -p $i && directories="$directories $i")
        done
        [ -f /etc/sysconfig/rhn/allowed-actions/configfiles/all ] ||
files="$files /etc/sysconfig/rhn/allowed-actions/configfiles/all"
        [ -n "$files" ] && touch $files
    fi
    if [ -z "$PROFILENAME" ] ; then
        profilename_opt=""
    else
        profilename_opt="--profilename=$PROFILENAME"
    fi
    /usr/sbin/rhnreg_ks --force --activationkey "$ACTIVATION_KEYS"
$profilename_opt

```

```

RET="$?"
[ -n "$files" ] && rm -f $files
[ -n "$directories" ] && rmdir $directories
if [ $RET -eq 0 ]; then
    echo
    echo "**** this system should now be registered, please verify ****"
    echo
else
    echo
    echo "**** Error: Registering the system failed."
    echo
    exit 1
fi
else
    echo "* explicitly not registering"
fi

if [ $ALLOW_CONFIG_ACTIONS -eq 1 ] ; then
    echo
    echo "* setting permissions to allow configuration management"
    echo "  NOTE: use an activation key to subscribe to the tools"
    if [ "$INSTALLER" == zypper ] ; then
        echo "          channel and zypper install/update rhncfg-actions"
    elif [ "$INSTALLER" == yum ] ; then
        echo "          channel and yum upgrade rhncfg-actions"
    else
        echo "          channel and up2date rhncfg-actions"
    fi
    if [ -x "/usr/bin/rhn-actions-control" ] ; then
        rhn-actions-control --enable-all
        rhn-actions-control --disable-run
    else
        echo "Error setting permissions for configuration management."
        echo "  Please ensure that the activation key subscribes the"
    fi
    if [ "$INSTALLER" == zypper ] ; then
        echo "  system to the tools channel and zypper install/update rhncfg-
actions."
    elif [ "$INSTALLER" == yum ] ; then
        echo "  system to the tools channel and yum updates rhncfg-
actions."
    else
        echo "  system to the tools channel and up2dates rhncfg-
actions."
    fi
    fi
    exit
fi
fi

if [ $ALLOW_REMOTE_COMMANDS -eq 1 ] ; then
    echo
    echo "* setting permissions to allow remote commands"
    echo "  NOTE: use an activation key to subscribe to the tools"
    if [ "$INSTALLER" == zypper ] ; then
        echo "          channel and zypper update rhncfg-actions"
    elif [ "$INSTALLER" == yum ] ; then
        echo "          channel and yum upgrade rhncfg-actions"

```

```

else
    echo "        channel and up2date rhncfg-actions"
fi
if [ -x "/usr/bin/rhn-actions-control" ] ; then
    rhn-actions-control --enable-run
else
    echo "Error setting permissions for remote commands."
    echo "    Please ensure that the activation key subscribes the"
    if [ "$INSTALLER" == zypper ] ; then
        echo "        system to the tools channel and zypper updates rhncfg-
actions."
    elif [ "$INSTALLER" == yum ] ; then
        echo "        system to the tools channel and yum updates rhncfg-
actions."
    else
        echo "        system to the tools channel and up2dates rhncfg-
actions."
    fi
    exit
fi
fi

echo
echo "OTHER ACTIONS"
echo "-----"
if [ $FULLY_UPDATE_THIS_BOX -eq 1 ] ; then
    if [ "$INSTALLER" == zypper ] ; then
        echo "zypper --non-interactive up zypper zypp-plugin-spacewalk; rhn-
profile-sync; zypper --non-interactive up (conditional)"
    elif [ "$INSTALLER" == yum ] ; then
        echo "yum -y upgrade yum yum-rhn-plugin; rhn-profile-sync; yum
upgrade (conditional)"
    else
        echo "up2date up2date; up2date -p; up2date -uf (conditional)"
    fi
else
    if [ "$INSTALLER" == zypper ] ; then
        echo "zypper --non-interactive up zypper zypp-plugin-spacewalk; rhn-
profile-sync"
    elif [ "$INSTALLER" == yum ] ; then
        echo "yum -y upgrade yum yum-rhn-plugin; rhn-profile-sync"
    else
        echo "up2date up2date; up2date -p"
    fi
fi
echo "but any post configuration action can be added here. "
echo "-----"
if [ $FULLY_UPDATE_THIS_BOX -eq 1 ] ; then
    echo "* completely updating the box"
else
    echo "* ensuring $INSTALLER itself is updated"
fi
if [ "$INSTALLER" == zypper ] ; then
    zypper ref -s
    zypper --non-interactive up zypper zypp-plugin-spacewalk
    if [ -x /usr/sbin/rhn-profile-sync ] ; then

```

```

        /usr/sbin/rhn-profile-sync
    else
        echo "Error updating system info in RHN Satellite."
        echo "    Please ensure that rhn-profile-sync is installed and rerun
it."
    fi
    if [ $FULLY_UPDATE_THIS_BOX -eq 1 ] ; then
        zypper --non-interactive up
    fi
elif [ "$INSTALLER" == yum ] ; then
    /usr/bin/yum -y upgrade yum yum-rhn-plugin
    if [ -x /usr/sbin/rhn-profile-sync ] ; then
        /usr/sbin/rhn-profile-sync
    else
        echo "Error updating system info in RHN Satellite."
        echo "    Please ensure that rhn-profile-sync is installed and rerun
it."
    fi
    if [ $FULLY_UPDATE_THIS_BOX -eq 1 ] ; then
        /usr/bin/yum -y upgrade
    fi
else
    /usr/sbin/up2date up2date
    /usr/sbin/up2date -p
    if [ $FULLY_UPDATE_THIS_BOX -eq 1 ] ; then
        /usr/sbin/up2date -uf
    fi
fi
echo "-bootstrap complete-"

```

## 4.2. Managing Systems with Satellite

Once systems are registered to Red Hat Satellite, these systems will appear in the **Overview** page of the **Systems** tab. The **Overview** page provides a summary of systems, including their status, the number of associated errata and packages, the base channel the system is under, and the entitlement level. The systems can then be managed using several methods.

### 4.2.1. Managing Individual Systems

The Satellite administrator can manage systems individually by clicking on the system name while on the **Overview** page of the **Systems** tab. This will take you to the **Details** tab for the system. The Details page has numerous tabs that provide specific system information as well as other identifiers unique to the system.

#### 4.2.1.1. Controlling the System's Power Management



#### Important

This feature is available only for Red Hat Satellite servers on x86\_64 architecture.

Satellite provides power management features for physical machines through **cobbler** integration with IPMI. Prior to using power management features, Satellite requires installation of the IPMI fencing agent on the server. Install the fencing agent using the following command:

```
# yum install fence-agents
```

This installs a set of fencing agents, including the IPMI agent.

After completing the fencing agent installation, Satellite requires some minor configuration to use the IPMI fencing agent. Edit the `/etc/rhn/rhn.conf` file and add the following property to the end of the file:

```
java.power_management.types = ipmilan
```

Save the file and restart the Satellite server:

```
# rhn-satellite restart
```

Use the following procedure to use Satellite's power management features.

1. Navigate to **Systems** → **System**.
2. Choose a system to manage.
3. Check your System's provisioning entitlement is enabled by navigating to **Details** → **Properties**. Check **Provisioning** under **Add-On Entitlements** and click **Update Properties**.
4. Navigate to **Provisioning** → **Power Management**.
5. Choose IPMI for the fencing agent **Type**.
6. Enter the **Network address** for your IPMI power management board. Satellite communicates with the power management board using the fencing agent.
7. Enter the **Username** and **Password** for the power management board.
8. Click **Save**.

You can now control the system's basic power management including **Power On**, **Power Off**, and **Reboot**.

#### 4.2.1.2. Viewing the System's Hardware Profile

The **Hardware** tab shows you the system's hardware profile. It includes information such as DMI, Networking details, and the description of what drivers are being used by the machine. This is useful in identifying the machine or when vendor information is needed.

Should any of the hardware on the system change or if the list is incomplete, you may also refresh the hardware list on this page by clicking **Schedule Hardware Refresh**.

1. Click **Systems** → **Systems**.
2. Click on the system name.
3. Click on **Hardware**.

#### 4.2.1.3. Scheduling a System Reboot from the Satellite

This will schedule a remote reboot of the system. This is suitable in situations where the system administrator is not in close proximity to the Satellite and requires a reboot for troubleshooting purposes.

1. Click **Systems** → **Systems**.

2. Click on the system name.
3. On the right-hand side of the page under **System Events**, click **Schedule System Reboot**.

#### 4.2.1.4. Changing a System's Base Channel Subscriptions

Base channel subscriptions are chosen by default based on the hardware and system profile that has been sent to Red Hat Satellite upon registration. However, this setting can be changed if there are issues that occur. To change the system's base channel:

1. Click **Systems** → **Systems**.
2. Click on the system name.
3. On the left-hand side of the page, under **Subscribed Channels**, click on **Alter Channel Subscriptions**.
4. Scroll down to the **Base Software Channel** section on the bottom of the page and choose the base channel you wish to apply.
5. Click **Confirm**.



#### Note

If you change the base channel subscription of the system, Satellite will unsubscribe your system from all previously subscribed channels and subscribe your system to the new base software channel you have chosen. Any additional child channels will have to be re-added.

#### 4.2.1.5. Changing a System's Child Channel Subscriptions

Additional channels may be added based on specific requirements that a system has, such as optional packages in the Red Hat Network Tools channel or Virtualization capabilities. To subscribe your system to additional channels, follow the steps below.

1. Click **Systems** → **Systems**.
2. Click on the system name.
3. On the left-hand side of the page, under **Subscribed Channels**, click on **Alter Channel Subscriptions**.
4. Choose the child channels that the system requires by ticking the checkboxes. You may choose as many as needed. Note that some channels may consume additional software entitlements if chosen.
5. Click **Change Subscriptions**.

#### 4.2.1.6. Adding Provisioning/Monitoring Entitlements to a System

Provisioning or Monitoring entitlements can be added to systems inside the Satellite, provided that these entitlements are available. These entitlements are relevant to systems that have the following requirements:

- ✦ *Provisioning*- This entitlement is required by a system that needs the ability to use kickstart, package rollback and configuration file management.
- ✦ *Monitoring*- This entitlement allows Satellite with Monitoring-entitled clients to notify administrators of system performance issues before they become critical.

1. Click **Systems** → **Systems**.
2. Click on the system name.
3. Click on **Details** → **Properties**.
4. Select the required entitlements in the **Add-On Entitlements** section.
5. Click **Update Properties**.

#### 4.2.1.7. Remotely Installing New Packages

The ability to choose packages to install remotely can be used to manage new or changed requirements in a system. If systems are powered down when the remote install is scheduled, the action will be performed once the system is back up.

Follow these steps to install packages from Red Hat Satellite:

1. Click **Systems** → **Systems**.
2. Click on the system name.
3. Click on **Software** → **Packages** → **Install**.
4. Select the packages to install on the system.
5. Click **Install Selected Packages**.
6. Choose to schedule the installation on a specific time or as soon as possible.
7. Click **Confirm**.



#### Note

If additional commands are required to install packages into the system, click **Add Remote Command to Package Install**. Add the script required to boot it and click **Confirm Remote Command and Schedule Package Install**. This additional step may be useful for systems that have special configuration requirements.

#### 4.2.1.8. Remotely Upgrading Packages

To remotely upgrade packages from the Red Hat Satellite:

1. Click **Systems** → **Systems**.
2. Click on the system name.
3. Click on **Software** → **Packages** → **Upgrade**
4. Select packages to be upgraded.
5. Choose to schedule the installation on a specific time or as soon as possible.
6. Click **Confirm**.

**Note**

If additional commands are required to install packages into the system, click **Add Remote Command to Package Install**. Add the script required to boot it and click **Confirm Remote Command and Schedule Package Install**. This additional step may be useful for systems that have special configuration requirements.

#### 4.2.1.9. Locking the System against Changes

In events where it is a requirement for a system to be kept unchanged, the system can be locked. A locked system will receive no package upgrades or any system-changing actions that occur even if the system is in a system group.

To lock a system:

1. Click **Systems** → **Systems**.
2. Click on the system name.
3. Click **Lock System** in the **Lock Status** field in the *System Info* section.

**Note**

A system that has pending actions cannot be locked. To cancel the scheduled actions, click on **Events** → **Cancel Events**.

#### 4.2.1.10. Configuring System Currency Weights/Multipliers

Included in Satellite is a System Currency report which lists registered systems ordered by score. The score is determined by the totals of the errata relevant to the systems. A specific weighted score per category per errata adds to the total score where the default weight awards critical security errata with the heaviest weight and enhancement errata with the lowest. The report can be used to prioritize maintenance actions on the systems registered to the Satellite. The following values are default system currency values stored in `/usr/share/rhn/config-defaults/rhn_java.conf`:

```
# multiplier for critical security errata
java.sc_crit = 32
# multiplier for important security errata
java.sc_imp = 16
# multiplier for moderate important security errata
java.sc_mod = 8
# multiplier for low important security errata
java.sc_low = 4
# multiplier for bugfix errata
java.sc_bug = 2
# multiplier for enhancement errata
java.sc_enh = 1
```

They can be overridden by adding them to the `/etc/rhn/rhn.conf` file. The higher the score, the higher the importance is given to the system in the report.

To change the default weight/multipliers of each errata type:



1. Log in as root on the Satellite server.
2. Edit the **/etc/rhn/rhn.conf** with your chosen text editor:

```
# vim /etc/rhn/rhn.conf
```

3. Add the custom values you wish to change in the **/etc/rhn/rhn.conf** file:

```
java.sc_imp = desired_value
java.sc_mod = desired_value
java.sc_low = desired_value
java.sc_bug = desired_value
java.sc_enh = desired_value
```

4. Save the **/etc/rhn/rhn.conf** file.
5. Restart the Satellite service:

```
# rhn-satellite restart
```



### Note

In Red Hat Satellite 5.5, the Currency Report and weight/multiplier values can also be retrieved using the Red Hat Network API Calls. See the *API Overview* for more information about the following calls:

- ✱ **system.getSystemCurrencyMultipliers** - this API call provides information about the current weight/multiplier configuration.
- ✱ **system.getSystemCurrencyScores** - this API call returns a list of systems, their scores and the counts of applicable errata of each type.

## 4.2.2. Managing System Groups

When administrators plan to create logical divisions of the organization's systems whether because of system use or role, Red Hat Satellite provides the ability to do so from the "System Groups" functionality. Segregating systems into logical grouping allows the administrator to maintain similar package versions across all the systems in the group, maintaining a standard build that is easier to manage.

The procedures on this section will assist in setting up a system group as well as basic guidance on System Group management.

### 4.2.2.1. Adding a System to a System Group

These steps show how to add an individual system to a System Group.

1. Click **Systems** → **Systems**.
2. Click on the system name.
3. Click **Groups** → **Join**.
4. Select the group or groups that the system should be added to.
5. Click **Join Selected Groups**.

#### 4.2.2.2. Adding Multiple Systems to the System Group

Multiple systems can be added to the system group instead of individually. This is most effective action to choose if there are more than one system to add to the system group.

1. Click **Systems** → **System Groups**.
2. Click on the desired system group.
3. Click the **Target Systems** subtab.



#### Note

Target Systems are eligible systems that are on the Satellite and can be added to the group.

4. Click **Add Systems**.

#### 4.2.2.3. Adding Group Administrators to the Group

These steps show how to add users as administrators of a specific group.

1. Click **Systems** → **System Groups**.
2. Click on the desired system group.
3. Click the **Admins** subtab.
4. Click on the usernames that need to be group administrators.
5. Click **Update**.



#### Note

Organization Administrators can administer all the systems within the Satellite.

#### 4.2.2.4. Removing Systems from the System Group

These are the steps to remove systems from a specific system group.

1. Click **Systems** → **System Groups**.
2. Click on the desired system group.
3. Click on **Systems**.
4. Select all the systems to be removed from the System Group.
5. Click **Remove Systems**.

#### 4.2.2.5. Applying Errata to Affected Systems in a Group

Errata can be remotely applied to multiple systems using the following method:

1. Click **Systems** → **System Groups**.

2. Click on the desired system group.
3. Click on the **Errata** subtab.
4. Select the *Advisory* that you would like to apply to the affected systems.
5. Click **Affected Systems** to view a list of systems that are affected by the erratum.
6. Select the systems you wish to apply the erratum to or choose **Select All**.
7. Click **Apply Errata**.

### 4.2.3. Managing Systems with System Set Manager

System Set Manager (SSM) allows administrators to work with large numbers of systems in the Red Hat Satellite server, performing system management, maintenance, and deployment. SSM is used when working with systems that are in different system groups but require similar maintenance, configuration or deployment changes.

There are other various features that System Set Manager can offer:

- » Scheduling errata updates, as well as upgrading, installing and removing packages
- » Managing systems' channel membership, the deployment of configuration channels and configuration channel subscriptions
- » Provisioning systems, running remote commands and tagging systems for snapshot rollbacks
- » Migrating systems to another organization and setting custom values for selected systems in the Satellite

In order to use SSM, make sure that systems have been added to SSM. This is the first action that should be undertaken before using SSM features.

#### 4.2.3.1. Adding Systems to SSM

Adding systems to SSM will allow the administrator to administer updates, configuration changes, etc to a specific group of systems regardless of the system group they belong to. To add a system to SSM:

1. Click **Systems** → **Systems**.
2. Click on the system name.
3. On the top right-hand corner of the page, click **add to ssm**.

This should add the system to the system list in SSM.

#### 4.2.3.2. Scheduling Errata Updates in SSM

To schedule errata updates for systems in the working group in SSM:

1. Click **Systems** → **Systems Set Manager**.
2. Click *Schedule Errata Updates* on the main page or click the **Errata** subtab.
3. Select the *Advisory* that you would like to apply to the affected systems. Choose as many as is applicable for the set of systems.
4. Click **Apply Errata**.

### 4.2.3.3. Managing Channel Memberships

The Channel Administrator can change the base channels the systems are subscribed to. Valid channels are either channels created by the organization or the default Red Hat base channel for specific operation system versions and processor type. Choosing a new base channel will unsubscribe the system from all previous channels. All child channels will have to be re-subscribed.

To change channel memberships:

1. Click **Systems** → **System Set Manager** → **Channels**.
2. As discussed, base channels need to be selected prior to subscribing to child channels. If only child channel subscriptions are to be changed, go the next step. If the base channel needs to be changed, follow these steps to subscribe to a base channel:
  - a. Click the **Base Channels** subtab.
  - b. Choose the base channel you wish to subscribe to from the **Desired Base Channel**.
  - c. Click **Confirm Subscriptions**.
  - d. Click the **Child Channels** subtab to go back to the Child Channel page.
3. Select **Subscribe** to subscribe the selected systems to the channel. Select **Unsubscribe** to unsubscribe the selected systems to the channel and select **Do Nothing** if no changes should be made for the channel subscriptions.
4. Click **Alter Subscriptions** to save the changes made.
5. A summary of changes will appear to confirm the changes made in the previous screen. Review these changes and select **Change Subscriptions** when the changes are correct.

### 4.2.3.4. Enabling Configuration Management with SSM

To manage different configuration files in a system, configuration management needs to be enabled.

#### Requirements

The following requirements are necessary to enable configuration management in a system:

- A provisioning entitlement. See the *Systems* chapter for the procedure on how to add a provisioning entitlement to a system.
- A subscription to the Red Hat Satellite Tools channel. See the *Systems* chapter for the procedure on how to change a child channel.
- An organization administrator.
- System subscribed to SSM. In order for SSM to perform this action, the systems need to be subscribed to SSM.

1. Click **Systems** → **System Set Manager** → **Configuration**.
2. Click the **Enable Configuration** subtab.
3. Schedule the package installation of the **rhcfg-\*** packages. Select a time for these configuration packages to be installed.
4. Click **Enable Red Hat Satellite Configuration Management**.

5. Open a terminal console on the individual systems or remotely login as root. The following actions need to be performed:
  - a. Run this command to complete the pending **rhncfg-\*** package installation:

```
# rhn_check
```

- b. Run the following command to enable Red Hat Network actions:

```
# rhn-actions-control --enable-all
```

#### 4.2.3.5. Subscribing to Configuration Channels with SSM

Configuration Channels are discussed further in the *Channel Management* chapter. This procedure only covers how to subscribe systems to configuration channels through SSM.

##### Requirements

In order for SSM to subscribe systems to channels, the following requirements need to be fulfilled:

- The system must be added to SSM. See the *Adding Systems to SSM* procedure in this section.
  - Configuration management should be enabled. See the *Enabling Configuration Management with SSM*. procedure in this chapter.
  - Configuration management requires that the system has a provisioning entitlement. See the *Systems* chapter for the procedure on how to add a provisioning entitlement to a system.
1. Click **Systems** → **System Set Manager** → **Configuration** or **Systems** → **System Set Manager** → **config channel subscriptions**.
  2. Select the channels the systems should be subscribed to.
  3. Click **Continue**.
  4. Choose a channel on the list and use the up and down arrows to change the priority. This assigns a rank to the channel. Ranking the channels will allow higher ranked channels to override any configuration changes from files with the same path in the lower channels.
  5. Select the configuration channel's priority by using the radio buttons on the side. This will rank the priority of the configuration channels listed here against the current configuration channels on the system.
  6. Click **Apply Subscriptions**.
  7. Confirm the systems against the configuration channels they will be subscribed to. Once all the information is confirmed correct, click **Confirm**.

#### 4.2.3.6. Deploying Configuration Channels through SSM

To deploy a changed configuration file associated with systems through SSM:

1. Click **Systems** → **System Set Manager** → **Configuration** or **Systems** → **System Set Manager** → **Deploy config channels**.
2. Select the filenames of the files to be deployed.

3. Schedule the deployment of the configuration file by choosing to deploy it as soon as possible or choose a specific date and time.
4. Click **Confirm File Deploy** to confirm the configuration deployment.

#### 4.2.3.7. Tagging Systems for Provisioning

Tagging the system allows it to rollback to the most recent snapshot of the system when the tag was created. To tag systems:

1. Click **Systems** → **System Set Manager** → **Tag**.
2. Fill in the *Tag name* field.
3. Click **Tag Current Snapshots**.

#### 4.2.3.8. Running Remote Commands using SSM

To run remote commands using SSM:

1. Click **Systems** → **System Set Manager** → **remote commands**.
2. Fill in the following fields:
  - ✦ Run as user
  - ✦ Run as group
  - ✦ Timeout (in seconds)
  - ✦ Script
3. Set a scheduled date and time for the shell script to run on the target systems.
4. Click **Schedule Remote Command**.

#### 4.2.4. Managing Action Chains

Satellite provides the ability to run several actions in a sequence. For example, to configure a webserver, you might need to install the **httpd** package on a system, upload a set of configuration file in **/etc/httpd/conf.d/**, and finally run a script to start the **httpd** service. Satellite can schedule and execute these actions in order using the action chaining features.

Action chaining can link the following actions in a sequence:

- ✦ Install a package
- ✦ Update a package
- ✦ Remove a package
- ✦ Verify a package
- ✦ Apply errata
- ✦ Run a remote command
- ✦ Deploy a configuration file
- ✦ Reboot a system

This applies to both individual systems and system sets.

To add an action to an action chain, initiate the action but select **Add to Action Chain** on the confirmation page, then click **Confirm** to add the action.

**Figure 4.1. Add to Action Chain**

Navigate to **Schedule** → **Action Chains** to view a list of your action chains. Click on the **Label** of an action chain to view its actions, edit their order, or delete individual actions from the chain. In addition, the **Schedule** section provides fields to select the date and time to schedule the action. After selecting an appropriate date and time to schedule the action, click the **Save and Schedule** button and Satellite executes the action chain at the date and time specified.

## 4.3. Managing Virtualized Client Systems

In order to manage and provision client systems, synchronize content from Red Hat Network's central servers to the Red Hat Satellite.

Synchronize at least the following channels:

For Red Hat Enterprise Linux 5:

- ✱ Red Hat Enterprise Linux Server (v. 5 for 32-bit x86) - rhel-i386-server-5 (and all child channels)
- ✱ Red Hat Network Tools for RHEL Server (v. 5 for 32-bit x86) - rhn-tools-rhel-i386-server-5
- ✱ Red Hat Enterprise Linux Server Virtualization (v. 5 for 32-bit x86) - rhel-i386-server-vt-5 (and all child channels)

For Red Hat Enterprise Linux 6:

- ✱ Red Hat Enterprise Linux Server (v. 6 for 64-bit x86\_64) - rhel-x86\_64-server-6 (and all child channels)
- ✱ Red Hat Network Tools for RHEL Server (v. 6 for 64-bit x86\_64) - rhn-tools-rhel-x86\_64-server-6

### 4.3.1. Setting Up the Host System for Your Virtual Systems

Before creating guest systems, prepare the host system by creating a Red Hat Enterprise Linux Server kickstart profile. Use that kickstart profile to install the operating system on your host. Once these steps are complete, you can proceed to provision virtual guests.

#### 4.3.1.1. Creating a Kickstart Profile for the Guest Systems

1. Log in to the Satellite's web interface. Navigate to the **Kickstart Overview** screen by clicking the **Manage Kickstarts** link in the **Tasks** widget in **Your Red Hat Network**, or by clicking on the **Systems** tab, followed by the **Kickstart** subtab in the left navigation bar.

2. On the **Kickstart Overview** page, click the **Create a New Kickstart Profile** link in the **Kickstart Actions** widget in the upper right corner.
3.
  - a. Enter a label for your profile that will enable you to distinguish it from your other profiles. For the remaining instructions, we'll assume the label is **host-system-for-virtual-guests**.
  - b. For the **Base Channel** field, select Red Hat Enterprise Linux (v.5 or 6 for \$ARCH) (where \$ARCH is the architecture of your host system).

**Note**

You may install 32-bit Red Hat Enterprise Linux 5 or 6 on a 64-bit host system. If you choose to do this, however, please be aware that your guest systems must also run the 32-bit version of Red Hat Enterprise Linux.

- c. In the **Kickstartable Tree** field, select **ks-rhel-\$ARCH-server-5 (or 6)** where \$ARCH is the architecture of your host system.
  - d. Select the **Virtualization Type**.

**Note**

If you are changing the **Virtualization Type** of an existing kickstart profile, it may also modify the bootloader and partition options, potentially overwriting any user customizations. Be sure to review the **Partitioning** tab to verify these settings when changing the **Virtualization Type**.

- e. Click the **Next** button in the lower right of the screen to continue on to the next step.

**Note**

If any of the fields are missing the options indicated above, you may not have successfully synced software channel content to your Satellite from Red Hat's servers.

4. Select the location of the distribution files for the installation of your host system. There will already be a **Default Download Location** filled out and selected for you on this screen. Click the **Next** button on this screen to continue to Step 3.

**Note**

If the default download location is missing, you may not have successfully synced software channel content to your Satellite from Red Hat's server.

5. Choose a root password to set on the host system you will be provisioning, and click **Finish** to finish creation of the profile.



- You will be shown the newly-created kickstart profile. You may browse through the various tabs of the profile and modify the settings as you see fit, but this is not necessary as the default settings work well for the majority of cases.

In order to be able to remotely start and stop the guest using the Satellite web interface, you will need to include the package **acpid**.

### 4.3.1.2. Kickstarting Your Host System

Kickstart your host system using your newly-created kickstart profile. There are three different scenarios for kickstarting your host system. Please read through these scenarios below, and follow the instructions for the scenario that applies best to you:

#### 4.3.1.2.1. Your Host System Does Not have Red Hat Enterprise Linux Installed

Create a boot CD to initiate the kickstart on your host system. You will be able to use the kickstart profile we created in earlier steps to provision the host. Note you must have physical access to the machine you intend to use in order to follow these steps:

- You will find an ISO to create a boot CD for your host by using **ssh** to log into your Satellite. It is at the following location on your satellite:

```
/var/satellite/rhn/kickstart/ks-rhel-i386-server-5.3/images/boot.iso
```



#### Note

It is possible to use a flash-memory USB key to boot your system in order to kickstart it. See the *Red Hat Enterprise Linux Installation Guide* for tips on how to do this. Note that your host system's hardware must support boot via these devices.

- Insert the boot CD in the drive and reboot the system, making sure the CD-ROM drive is set as the primary boot device in the system's BIOS.
- After reboot, you will find yourself at a boot prompt. Type the following command at this prompt to start your kickstart:

```
linux \
ks=http://your-satellite.example.com/ks/label/the profile label you
created earlier
```



#### Note

For some systems, you may need to add **ksdevice=eth0** to the command above or disable one of two or more NICs in the system's BIOS to avoid confusion during the kickstart process.

- The kickstart for your host system will begin. It will take around fifteen minutes to complete. Upon successful completion of this kickstart, you will have provisioned a host system for your virtual guest and registered it to your Satellite.

#### 4.3.1.2.2. Your Host System Has Red Hat Enterprise Linux 6 or 7 Installed

Register your host system to your Satellite and check to see if the required KVM packages are installed on the system. If they are not, install them using the Satellite.



### Note

On Red Hat Enterprise Linux 6, virtualization is only supported on 64-bit Intel and AMD machines.



### Note

The Xen virtualization host is not currently supported on Red Hat Enterprise Linux 6.

1. Register your host system to your Red Hat Satellite. Use **ssh** to connect to your host system then issue the following command as root:

```
# rhnreg_ks --serverUrl=http://your-satellite.example.com/XMLRPC \
--username=username --password=password
```



### Note

If your host system is already registered to a different Red Hat Network server, add the **--force** option to the command above.

2. Open up the host system's profile in the Satellite web interface. Log into the web interface of your Satellite at <https://your-satellite.example.com/>. Click on the **Systems** tab in the top navigational bar. You will see the host system you just registered - click on its profile name to access its system profile page.
3. Make sure your system has access to the software channels it needs to access the software required for hosting virtual guests. From your host system's profile page, click on the **Alter Channel Subscriptions** link on the profile page under the **Subscribed Channels** header. Check the **RHEL Virtualization** and **Red Hat Network Tools for RHEL Server** checkboxes and click the **Change Subscriptions** button underneath the list of channels.
4. Check if the necessary software installed for hosting virtual guests is on the system. On the host system, issue the following command as root:

For Red Hat Enterprise Linux 6:

```
# rpm -q qemu-kvm rhn-virtualization-host python-virtinst
```

For Red Hat Enterprise Linux 7:

```
# rpm -q qemu-kvm rhn-virtualization-host virt-install
```

If **rpm** indicates these packages are not installed, you must install them by running the following command as root on the system:

For Red Hat Enterprise Linux 6:

```
# yum install qemu-kvm rhn-virtualization-host python-virtinst
```

For Red Hat Enterprise Linux 7:

```
# yum install qemu-kvm rhn-virtualization-host virt-install
```

- Restart the machine to pick up the changes, or use the appropriate **modprobe** command for your processor:

```
# modprobe kvm_intel
```

or:

```
# modprobe kvm_amd
```

- Install and run the **osad** package in order for your host system to be responsive to commands sent from the Satellite, such as start, pause, resume, and shutdown. To install:

```
# yum install -y osad
```

After installation, start the **osad** process:

```
# /sbin/service osad restart
```

- Your host system will now be ready for Red Hat Network virtual guest provisioning.

#### 4.3.1.3. Your Host System Has Red Hat Enterprise Linux 5 Installed

You should register your host system to your Satellite and check to see if the required Xen or KVM packages are installed on the system. If they are not, install them using the Satellite.

- Register your host system to your Satellite. Use **ssh** to connect to your host system. Register your host system to your satellite issuing the following command as root:

```
# rhnreg_ks --serverUrl=http://your-satellite.example.com/XMLRPC \
  --username=username --password=password
```



#### Note

If your host system is already registered to a different Red Hat Network server, add the **--force** option to the command above.

- Open up the host system's profile in the Satellite web interface. Log into the web interface of your Satellite at <https://your-satellite.example.com/>. Click on the **Systems** tab in the top navigational bar. You will see the host system you just registered - click on its profile name to access its system profile page.
- Make sure your system has access to the software channels it needs to access the software required for hosting virtual guests. From your host system's profile page, click on the **Alter Channel Subscriptions** link on the profile page under the **Subscribed Channels** header. Check the **RHEL Virtualization** and **Red Hat Network Tools for RHEL Server** checkboxes and

click the **Change Subscriptions** button underneath the list of channels.

4. Check to see if you have the necessary software installed for hosting virtual guest on the system. On the host system, issue the following command as root:

```
# rpm -q xen kernel-xen rhn-virtualization-host
```

For KVM, issue the following command as root:

```
# rpm -q kvm kmod-kvm rhn-virtualization-host python-virtinst
```

If **rpm** indicates these packages are not installed, you must install them by running the following command as root on the system:

```
# yum install xen kernel-xen rhn-virtualization-host
```

For KVM users, install by running the following command as root:

```
# yum install kvm kmod-kvm rhn-virtualization-host python-virtinst
```

For Xen, you will then need to edit the **/etc/grub.conf** configuration file to boot the new xen kernel by default. To do this, select the lines in **grub.conf** that pertain to the xen kernel from the beginning of the **title** line to the end of the **initrd** line, copy the lines, delete them, and paste them into the file as the first kernel entry in **grub.conf**. Also ensure that the value of the default variable at the top of **grub.conf** is set to a value of '0'.



### Note

If you ever update the kernel on the host system, the standard kernel is the default choice upon reboot. To ensure that the Xen kernel is chosen by default, change the following value in the **/etc/sysconfig/kernel** file:

```
DEFAULTKERNEL=kernel
```

Change the value to **kernel-xen**:

```
DEFAULTKERNEL=kernel-xen
```

5. Restart the machine to pick up the changes, or use the appropriate **modprobe** command for your processor:

```
# modprobe kvm_intel
```

or:

```
# modprobe kvm_amd
```

6. Reboot the system ensuring that it boots into the xen kernel. Use the command **uname -r** to see if the running kernel is a xen kernel. If you do not see the Xen string in the name of the kernel you have not booted into the correct kernel.



### Note

If the system already has Xen and **kernel-xen** installed you do not need to reboot after installing **rhn-virtualization-host**.

7. Install and run the **osad** package in order for your host system to be responsive to commands sent from the Satellite, such as start, pause, resume, and shutdown. To install:

```
# yum install -y osad
```

After installation start the **osad** process:

```
# /sbin/service osad restart
```

8. Your host system will now be ready for Red Hat Network virtual guest provisioning.

## 4.3.2. Setting Up Your Virtual Systems

To work with virtual guest systems create a kickstart profile that will allow you to easily provision virtual guests.

### 4.3.2.1. Create a Kickstart Profile for the Guest Systems

1. Log on to the Satellite's web interface. Navigate to the **Kickstart Overview** screen by clicking on the **Manage Kickstarts** link in the **Tasks** widget in **Overview**, or by clicking on **Systems** in the top navigation bar and then **Kickstart** from the left navigation bar.
2. On the **Kickstart Overview** page, click the **Create a new Kickstart Profile** link in the **Kickstart Actions** widget in the upper right corner.
3. The next page displayed is Step 1 of the kickstart profile creation process:
  - a. Enter a label for the profile that will allow you to distinguish it from the other profiles. A good choice would be **guest-system**.
  - b. For the **Base Channel** field, select **Red Hat Enterprise Linux \$PRODUCT (v.5 or 6 for \$ARCH)** where \$ARCH is the architecture of your host system's operating system and \$PRODUCT is either Server or Client.



### Note

Red Hat Enterprise Linux Client 5 or Red Hat Enterprise Linux Client 6 may not be available for selection if you did not synchronize the Client software channels to your Satellite.

**Note**

The channel labels for Red Hat Enterprise Linux 5 or Red Hat Enterprise Linux 6 and Red Hat Enterprise Linux 5 Desktop or Red Hat Enterprise Linux 6 Desktop refer to 'server' and 'client' respectively.

- c. For the **Kickstartable Tree** field, select **ks-rhel-\$ARCH-\$PRODUCT-5.3** where \$ARCH is the architecture of your host system and \$PRODUCT is either 'server' or 'client', depending on which product with which you would like to provision your guest.
- d. Select the **Virtualization Type**.

**Note**

If you are changing the **Virtualization Type** of an existing kickstart profile, it may also modify the bootloader and partition options, potentially overwriting any user customizations. Be sure to review the **Partitioning** tab to verify these settings when changing the **Virtualization Type**.

- e. Click the **Next** button in the lower right of the screen to continue on to the next step.
4. For Step 2 of the kickstart profile creation process, select the location of the distribution files for the installation of your guest system. There will already be a **Default Download Location** filled out and selected for you on this screen. Click the **Next** button on this screen to continue to Step 3.

**Note**

Red Hat Enterprise Linux Client 5 or Red Hat Enterprise Linux Client 6 may not be available for selection if you did not synchronize the Client software channels to your Satellite.

5. For Step 3 of the kickstart profile creation process, choose a root password for the guest system you are provisioning, and click **Next** to finish creation of the profile.

This completes kickstart profile creation. After completing Step 3 you will be taken to the profile details. You may browse through the various tabs of the profile and modify the settings as you see fit, but this is not necessary as the default settings work well for the majority of cases. While the interface allows you to allocate less, we strongly recommend allocating at least 3 GB of storage for your guest system with this kickstart profile.

#### 4.3.2.2. Provisioning Your Guest Systems

1. Log into the Satellite's web interface. Browse to your host system's profile by clicking on the **Systems** tab in the top navigation bar, and click on the system's name.
2. To schedule a kickstart for a guest system, go to the **Virtualization** → **Provisioning** tab in the host system's profile. For the **Guest Name** field choose **guest1**. For the **Memory Allocation**, **Virtual CPUs**, and **Storage** fields, the default values should be fine. Feel free to change these as desired, taking note of the advice provided for each field in the interface. For the **Kickstart Profile** field, select the previously created guest system profile.

- Click on the **Schedule Kickstart and Finish** button in the lower-right corner of the screen. You will be taken to the **Kickstart Status** page where you can follow along with the guest's kickstart progress. The status screen will indicate when the kickstart is successfully completed. To view your new guest, click on the **Virtualization** tab of the host system's profile on the Satellite. To view a list of virtual systems, navigate to **Systems** → **Systems** → **Virtual Systems**.



### Note

If you do not see the **Initiate a kickstart** guest message on the **Kickstart Status** page shortly after scheduling the kickstart of the guest, you may be missing **osad** on your host.

Host systems require the **osad** package in order to be responsive to commands sent from the Satellite, such as start, pause, resume, and shutdown. If **osad** is not installed and running, the host system will not receive these commands from the web interface for 2.5 hours, or the next time that the Red Hat Network daemon runs.

You can check whether or not **osad** is installing and running by checking the **OSA Status** field in the host system's profile on the Satellite. If the **OSA Status** field does not exist or the field indicates that the system has not contacted Satellite in several minutes you will need to install **osad** before you can successfully provision a guest on the host system. Run **yum install -y osad** as root.



### Note

You may receive the following message from the **Kickstart Status** page during the guest's kickstart:

```
The install process on the guest system has not communicated to
Red Hat Network in
the past n minutes. This may be due to a hung install process,
or it
may just be due to a slow install because of hardware
constraints. A
log of the installation process is available, you may wish to
review
it to troubleshoot this issue.
```

This message should not cause alarm unless more than twenty minutes have passed. To check if the kickstart is continuing look at the installation log to make sure there are no errors, reload the Kickstart Status page, and check that the Last File Request field continues to be updated.

- If you would like to register additional guests to your host, repeat the steps above. It is important to remember that you can only provision one guest at a time. If you attempt to schedule a guest kickstart while another is currently taking place, the current guest kickstart process will be canceled and the new guest kickstart process will begin.
- View your newly-created virtual guest's system in the Satellite's web interface by clicking on the **Virtualization** tab in the host system's profile. Then, click on the profile name of your virtual

system to view its Satellite system profile.

### 4.3.2.3. Managing Your Virtual Guest Entitlements

Red Hat Satellite features Flex Guest entitlements that enable you to assign entitlements to your virtual guests without consuming a standard entitlement reserved for physical systems.

To manage your Flex Guest entitlements, click **Overview** -> **Subscription Management** -> **Virtualization Entitlements** -> **Flex Guest Entitlement Consumers**. This page lists all virtual guests consuming Flex Guest entitlements.

To find and convert any virtual guests that consume standard entitlements, click the **Guests Consuming Regular Entitlements** subtab.

### 4.3.3. Working With Your Virtual Systems

Once you have set up your virtual system, you can then manage and customize it via various methods, including connecting via SSH and via the virtualization management interface on the host system.



#### Note

This section deals primarily with Xen hosts. In Red Hat Enterprise Linux 6, Xen is currently not supported, and KVM is the recommended virtualization method. See the *Red Hat Enterprise Linux Virtualization Guide* for detailed instructions on KVM usage.

#### 4.3.3.1. Logging into Virtual Systems Directly via SSH

1. You will need to locate the virtual system's IP address. Locate it by navigating to the **Systems** → **Virtual Systems** tab and clicking on the virtual system's profile name.
2. On the virtual system's profile page, you'll find the IP address in the left-hand column in the **IP Address** field.
3. Connect to the IP address by using **ssh** as root with the password you set for the virtual system in the kickstart profile.

#### 4.3.3.2. Gaining Console Access Via the Host

1. Connect to the host system and determine the ID number of the guest you would like to work with. Connect to the host system via **ssh** and run the following command:

```
# xm list
```

This will provide you with a list all of the guests created on your Satellite, including their ID number. The guest we created earlier, **guest1**, will be in this list with an ID number. For example, if this guest has been assigned an ID of 2, then:

2. Run the following command to access the console of this virtual system:

```
# xm console 2
```

You will immediately be able to view a login prompt on **guest1**.



3. Log in to **guest1** as root using the same password you set in the kickstart profile you used to provision the system.

(There may be some messages on the screen. In this case, hit the **Enter** key on your keyboard to receive a fresh login prompt.)

4. To exit the guest console and return to the host system's command prompt, you may hit the **Ctrl** and **]** keys on your keyboard simultaneously.

#### 4.3.3.3. Installing Software Via the Satellite Web Interface

1. Browse to the virtual system's profile in your Satellite's web interface by logging in and navigating to **Systems** → **Systems** → **Virtual Systems** and clicking on the name of your virtual system's profile.
2. In the virtual system's profile, click on the **Software+Packages** tab.
3. Click on **Install New Packages** in the **Packages** tab menu.
4. Select the packages you wish to install and click the **Install Selected Packages** button in the lower right-hand corner of the screen.
5. Review the package install details and click on the **Confirm** button in the lower right-hand corner of the screen.
6. The package install will take place the next time the guest system checks in with the Satellite. To force the install to take place immediately, you may run the **rhnc** command on the guest system.

#### 4.3.3.4. Installing Software Via Yum From the Virtual System

Your virtual system was registered to your Satellite as part of the guest provisioning process, so the **yum** command can be used to install and update software. For example, to install the text editor vim, issue the following command as root:

```
# yum install -y vim-enhanced
```

#### 4.3.3.5. Restarting Guests when Host Reboots

By default, when a host system reboots, the guests are not restarted and must be manually started by the administrator.

However, the **rhnc-virtualization-host** service can restart guests automatically in the event of a host system reboot.

To use this service, follow these steps:

1. Locate the guest's config file on the host in **/etc/sysconfig/rhn/virt/**. It will be named by UUID, but the correct file can be found by using the **grep** command to search for the guest name within the UUID files.
2. When you have found the UUID file corresponding to your guest system, create a symbolic link from the UUID file to the **/etc/sysconfig/rhn/virt/auto/** directory.

```
# ln -s /etc/sysconfig/rhn/virt/GUEST_UUID.xml  
/etc/sysconfig/rhn/virt/auto/
```

### 4.3.3.6. Deleting Virtual Systems

Deleting a virtual system is a multi-step process.

1. Shut down the virtual system that you wish to delete. You may do this by browsing to the host system's profile in the Satellite web interface, clicking on the virtualization tab, and selecting the virtual systems that you would like to delete. Finish shutting down by clicking the **Shutdown Systems** button at the bottom of the screen.
2. Delete the virtual system from Satellite. This is accomplished by selecting the virtual system's checkbox and clicking the **Delete System** button at the bottom of the screen.



#### Note

Allow for at least two minutes between shutting down a virtual system and deleting it. Otherwise, the virtual system may not shut down properly and you will delete it while it is running. If you delete a virtual system from Satellite while it is running, it will reappear on the Satellite the next time it checks in. If this happens, simply shutdown the system, wait two minutes, and delete it again.

3. Delete the disk image for the virtual system you would like to delete. You will find the disk image for **guest1**, for example, at the following location on the host system:

```
/var/lib/xen/disk-images/guest1.disk
```

Delete it with the following command:

```
# rm /var/lib/xen/disk-images/guest1.disk
```

If **guest1** was on a KVM host system the disk image could be found at:

```
/var/lib/libvirt/images/guest1.img
```

4. Finally, you must delete the Red Hat Network configuration files from the host system. To locate the Red Hat Network configuration file for **guest1**, run the following command:

```
# grep guest1 /etc/sysconfig/rhn/virt/*.xml
```

Then delete the file indicated. For example:

```
# rm /etc/sysconfig/rhn/virt/14e5cfbf72342515236ad74b260c2f6b.xml
```

5. You have successfully deleted a guest system from your host system and from Satellite.

## Chapter 5. Errata Management

Custom errata enables organizations to issue errata alerts for the packages in their custom channels, schedule errata deployment and manage errata across organizations.



### Warning

If the organization is using both Red Hat Satellite Proxy and Red Hat Satellite, manage errata only on the Satellite, since the Proxy servers receive updates directly from it. Managing errata on a Proxy in this combined configuration risks putting your servers out-of-sync.

There are two types of errata managed by the Errata section:

1. **Published Errata** - displays the errata alerts the organization has created and disseminated. To edit an existing published errata, follow the steps described in [Section 5.1, “Creating and Editing Errata”](#). To distribute the errata, click **Send Notification** on the top-right corner of the **Errata Details** page. The errata alert is sent to the administrators of all affected systems.
2. **Unpublished Errata** - displays the errata alerts your organization has created but not yet distributed. To edit an existing unpublished errata, follow the steps described in [Section 5.1, “Creating and Editing Errata”](#). To publish the errata, click **Publish Errata** on the top-right corner of the **Errata Details** page. Confirm the channels associated with the errata and click the **Publish Errata** button, now in the lower-right corner. The errata alert is shifted to the **Published** page awaiting distribution.

### 5.1. Creating and Editing Errata

Follow this procedure to make a custom errata alert:

1. On the top navigation bar, click on **Errata** then select **Manage Errata** on the left navigation bar. From the **Errata Management** page, click on **create new erratum**.
2. Enter a label for the erratum in the **Advisory** field, ideally following a naming convention adopted by your organization. Note that this label cannot begin with the letters "RH" (capitalized or not) to prevent confusion between custom errata and those issued by Red Hat .
3. Then, complete all remaining required fields and click the **Create Errata** button. View standard Red Hat Errata Alerts for examples of properly completed fields.

Red Hat Satellite administrators may also create errata by cloning an existing one. This cloning preserves package associations and simplifies issuing errata. See [Section 5.4, “Cloning Errata”](#) for instructions.

To edit an existing errata alert's details, click its advisory in the **Errata Management** page, make the changes in the appropriate fields of the **Details** tab, and click the **Update Errata** button. Click on the **Channels** tab to alter the errata's channel association. Click on the **Packages** tab to view and modify its packages.

To delete errata, select their checkboxes in the **Errata Management** page, click the **Delete Errata** button, and confirm the action. Note that deleting published errata may take a few minutes.



## Note

To receive an email when errata alerts are issued for your systems, go to **Your Red Hat Network => Your Preferences** in the Red Hat Network Management Website and select **Receive email notifications**. This is a useful setting for administrators of subscribed systems in your organization.

## 5.2. Assigning Packages to Errata

Follow this procedure to assign packages to errata.

1. After selecting an erratum to edit, click on the **Packages** tab then the **Add** subtab.
2. To associate packages with the erratum being edited, select the channel from the **View** dropdown menu that contains the packages and click **View**. Packages already associated with the erratum being edited are not displayed. Selecting **All managed packages** presents all available packages.
3. After clicking **View**, the package list for the selected option appears. Note that the page header still lists the errata being edited.
4. In the list, select the checkboxes of the packages to be assigned to the edited errata, and click **Add Packages** at the bottom-right corner of the page.
5. A confirmation page appears with the packages listed. Click **Confirm** to associate the packages with the errata. The **List/Remove** subtab of the **Managed Errata Details** page appears with the new packages listed.

Once packages are assigned to an erratum, the errata cache is updated to reflect the changes. This update is delayed briefly so that users may finish editing an erratum before all of the changes are made available. To initiate the changes to the cache manually, follow the directions to **commit the changes immediately** at the top of the page.

## 5.3. Publishing Errata

After adding packages to the errata, the errata needs to be published in order for it to be disseminated to affected systems. Follow this procedure to publish errata:

1. On the top navigation bar, click on **Errata** → **Manage Errata** on the left navigation bar.
2. Click on **Publish Errata**. A confirmation page appears that will ask you to select which channels you wish to make the errata available in. Choose the relevant channels.
3. Click **Publish Errata**. The errata published will now appear in the **Published** page of **Manage Errata**.

## 5.4. Cloning Errata

Errata can be cloned for easy replication and distribution as part of Red Hat Satellite. Only errata potentially applicable to one of your channels can be cloned. Errata can be applicable to a channel if that channel was cloned from a channel to which the errata applies. To access this functionality, click **Errata** on the top navigation bar, then **Clone Errata** on the left navigation bar. This button appears only for Red Hat Satellite customers.

In the **Clone Errata** page, select the channel containing the errata from the **View** dropdown menu and click **View**. Once the errata list appears, select the checkbox of the errata to be cloned and click **Clone Errata**. A confirmation page appears with the errata listed. Click **Confirm** to finish the cloning.

The cloned errata appears in the unpublished errata list. From there, verify the errata text and the packages associated with that errata. Once ready, publish the errata so it is available to users in your organization.

## Appendix A. Boot Devices

Automated installation (or *kickstart*) is an essential part of efficient system provisioning. This appendix describes how to prepare different types of boot media for use with kickstarting clients.

The Red Hat Enterprise Linux CD boot image **boot.iso** is a required prerequisite for creating boot devices. Make sure that this is available somewhere on the system and take note of its location.



### Important

Install the **syslinux** and **syslinux-extlinux** packages to use the following procedures.

```
# yum install syslinux syslinux-extlinux
```

The **syslinux** package installs files in **/usr/share/syslinux/** for Red Hat Enterprise Linux 6. If using Red Hat Enterprise Linux 5, substitute this directory with **/usr/lib/syslinux/**.

The **syslinux-extlinux** package installs tools for USB boot media creation.

### Procedure A.1. CD and DVD Boot Media



### Note

The backslash "**\**" is used below to represent a continuation of one line at the shell prompt.

1. Create a working directory for the boot image:

```
# mkdir -p temp cd/isolinux
```

2. Mount the boot image to the **temp** directory:

```
# mount -o loop boot.iso temp
```

3. Copy the required files for a Boot Media device to the previously created directory:

```
# cp -aP temp/isolinux/* cd/isolinux/
```

4. Unmount the **temp** directory and change the permissions on the **cd** directory to be readable and writable to the user:

```
# umount temp
# chmod -R u+rw cd
```

5. Change to the **./cd** directory:

```
# cd ./cd
```

6. Copy the `/usr/share/syslinux/menu.c32` file to the `./cd` directory:

```
# cp -p /usr/share/syslinux/menu.c32 isolinux
```

7. Customize any boot parameters and targets in `isolinux.cfg` as needed for CD booting.
8. Use the `mkisofs` to create an ISO to burn to a CD or DVD.

```
# mkisofs -o ./custom-boot.iso -b isolinux/isolinux.bin -c
isolinux/boot.cat -no-emul-boot \
    -boot-load-size 4 -boot-info-table -J -l -r -T -v -V "Custom Red
Hat Enterprise Linux Boot" .
```

9. Burn the directory to a CD or DVD to complete the procedure.

## Procedure A.2. PXE Boot

1. Create a working directory for the boot image:

```
# mkdir -p temp pxe/pxelinux.cfg
```

2. Mount the boot image to the `temp` directory:

```
# mount -o loop boot.iso temp
```

3. Copy the required files for a PXE Boot device to the previously created directory:

```
# cp -aP temp/isolinux/* pxe/
```

4. Unmount the `temp` directory and change the permissions on the `cd` directory to be readable and writable to the user:

```
# umount temp
# chmod -R u+rw pxe
```

5. Change to the `/pxe` directory:

```
# cd ./pxe
```

6. Copy the `/usr/share/syslinux/menu.c32` file to the `/pxe` directory:

```
# cp -p /usr/share/syslinux/menu.c32 .
```

7. Move the `isolinux.cfg` file to `pxelinux.cfg/default`:

```
# mv isolinux.cfg pxelinux.cfg/default
```

8. Remove the temporary files:

```
# rm -f isolinux.bin TRANS.TBL
```

9. Copy the `/usr/share/syslinux/pxelinux.0` file to the `/pxe` directory:

```
# cp -p /usr/share/syslinux/pxelinux.0 .
```

- Open the **pxelinux.cfg/default** file in your preferred text editor, and customize any boot parameters and targets as needed for PXE booting.

### Procedure A.3. USB Boot Media



#### Warning

Be extremely careful when carrying out these commands as root (required for most critical parts). These commands access device files and using them incorrectly could irrecoverably damage your system. The example below uses **/dev/loop0** for mounting, make sure you use the correct device for your system. You can check which is the correct device using the **losetup -f** command.

- Create a working directory for the boot image:

```
# mkdir -p temp usb/extlinux
```

- Mount the boot image to the **temp** directory:

```
# mount -o loop boot.iso temp
```

- Copy the required files for a USB Media Boot device to the previously created directory:

```
# cp -aP temp/isolinux/* usb/extlinux/
```

- Unmount the **temp** directory and change the permissions on the **cd** directory to be readable and writable to the user:

```
# umount temp
# chmod -R u+rw usb
```

- Copy the **/usr/share/syslinux/menu.c32** file to the **./usb/extlinux/** directory:

```
# cp -p /usr/share/syslinux/menu.c32 ./usb/extlinux/
```

- Move the **usb/extlinux/isolinux.cfg** file to **usb/extlinux/extlinux.conf**:

```
# mv usb/extlinux/isolinux.cfg usb/extlinux/extlinux.conf
```

- Remove the temporary files:

```
# rm -f usb/extlinux/isolinux.bin usb/extlinux/TRANS.TBL
```

- Convert the **custom-boot.img** file and copy it:

```
# dd if=/dev/zero of=./custom-boot.img bs=1024 count=300000
```

- Discover the correct mounting location for the loopback device:



```
# losetup -f  
/dev/loop0
```

Set up the loopback device with the boot image:

```
# losetup /dev/loop0 ./custom-boot.img
```

10. Open the **fdisk** utility:

```
# fdisk /dev/loop0
```

Create one primary bootable partition on the device. This can be done by using the following key press combination: **n p 1 Enter Enter a 1 p w**

11. Copy the master boot record (MBR) to the loopback device:

```
# dd if=/usr/share/syslinux/mbr.bin of=/dev/loop0
```

12. Add partition maps to the loopback device:

```
# kpartx -av /dev/loop0
```

13. Create the file system:

```
# mkfs.ext2 -m 0 -L "Custom Red Hat Enterprise Linux Boot"  
/dev/mapper/loop0p1
```

14. Mount the device:

```
# mount /dev/mapper/loop0p1 temp
```

15. Delete temporary files:

```
# rm -rf temp/lost+found
```

16. Copy the **usb/extlinux/** directory to a temporary location:

```
# cp -a usb/extlinux/* temp/
```

17. Install the bootloader in the temporary location:

```
# extlinux -i temp
```

18. Unmount the temporary location:

```
# umount temp
```

19. Delete the partition maps on the loopback device:

```
# kpartx -dv /dev/loop0
```

20. Delete the loopback device:

```
# losetup -d /dev/loop0
```

Synchronize the file system changes:

```
# sync
```

21. Open the **extlinux.conf** file in your preferred text editor, and customize any boot parameters and targets as needed for USB booting.
22. Transfer the image to a USB device to complete the procedure. Insert the device, and run the **dmesg** command to check the mounting location. In this example, it is **/dev/sdb**.

Unmount the USB device:

```
# umount /dev/sdb
```

Copy the image to the USB device:

```
# dd if=./custom-boot.img of=/dev/sdb
```

## Appendix B. Custom Package Management

This chapter provides an overview of how to build packages for successful delivery via Red Hat Network. Topics covered include why RPM should be used, how to build packages for Red Hat Network, and how to properly sign packages.

### B.1. Building Packages for Red Hat Network

Red Hat Network uses the *RPM Package Manager* (RPM) technology to determine what software additions and updates are applicable to each client system. Packages retrieved from Red Hat Network are usually in RPM format. Entire ISO images, however, are available through the **Software** tab of the Red Hat Network website, but are not available in Red Hat Satellite installations. If the Satellite server has Solaris support enabled, use Red Hat Network Push to upload Solaris packages to custom channels used by Solaris clients.

RPM is a tool that provides users with a simple method for installing, uninstalling, upgrading, and verifying software packages. It also allows software developers to package the source code and compiled versions of a program for end users and developers.

#### B.1.1. RPM Benefits

RPM provides the following advantages:

##### Easy Upgrades

Using RPM, you upgrade individual components of a system without completely reinstalling. When Red Hat releases a new version of Red Hat Enterprise Linux, users do not have to reinstall in order to upgrade. RPM allows intelligent, fully-automated, in-place upgrades of the system. Configuration files in packages are preserved across upgrades so users do not lose customizations. There are no special upgrade files needed to update a package because the same RPM file is used to install and upgrade the package.

##### Package Querying

RPM provides querying options that allows a search through the entire RPM database for all packages or just for certain files. RPM can also find out what package the file belongs to and where the package came from. The files contained in the package are in a compressed archive, with a custom binary header containing useful information about the package and its contents. RPM queries the headers quickly and easily.

##### System Verification

Another feature is the ability to verify packages. If there are concerns that a file related to a package was deleted, verify the package to check the status of the files it provides. The verification notifies you of any anomalies. If errors do exist, the files are reinstalled easily. Modified configuration files are preserved during reinstallation.

##### Pristine Sources

A crucial design goal of RPM is to allow the use of *pristine* software sources, as distributed by the original authors of the software. With RPM, the pristine sources can be packaged, along with any patches that were used, plus complete build instructions. This is an important advantage for several reasons. For instance, if a new version of a program is released, it is unnecessary to start from scratch to make it compile. Looking at the match may allow you to see what you *might* need to do. All the compiled-in defaults and changes made to get the software to build properly are easily visible using this technique.

Keeping sources pristine may seem important only to developers, but it results in higher quality software for end users as well.

### B.1.2. Red Hat Network RPM Guidelines

The strength of RPM lies in its ability to define dependencies and identify conflicts accurately. Red Hat Network relies on this aspect of RPM. Red Hat Network offers an automated environment, which means that no manual intervention can take place during the installation of a package. Therefore, when building RPMs for distribution through Red Hat Network, it is imperative to follow these rules:

1. Learn RPM. It is crucial to have a fundamental understanding of the important features of RPM to build packages properly. For more information about RPM, start with the following resources:

✎ [http://docs.fedoraproject.org/en-US/Fedora\\_Draft\\_Documentation/0.1/html/RPM\\_Guide/index.html](http://docs.fedoraproject.org/en-US/Fedora_Draft_Documentation/0.1/html/RPM_Guide/index.html)

✎ [http://docs.fedoraproject.org/en-US/Fedora\\_Draft\\_Documentation/0.1/html/Packagers\\_Guide/index.html](http://docs.fedoraproject.org/en-US/Fedora_Draft_Documentation/0.1/html/Packagers_Guide/index.html)

✎ <http://www.gurulabs.com/GURULABS-RPM-LAB/GURULABS-RPM-GUIDE-v1.0.PDF>

2. When building an RPM for a child channel, build the package on a fresh install of Red Hat Enterprise Linux of the same version as the child's base channel. Be sure to apply all updates from Red Hat Network first.
3. The RPM package must install without using the **--force** or **--nodeps** options. If an RPM cannot be installed cleanly on a build system, Red Hat Network cannot install it automatically on a system.
4. The RPM package filename must be in the NVR (name, version, release) format and must contain the architecture for the package. The proper format is **name-version-release.arch.rpm**. For example, a valid RPM package filename is **pkgname-0.84-1.i386.rpm**, where name is *pkgname*, version is *0.84*, release is *1*, and arch is *i386*.
5. The RPM package should be signed by the maintainer of the package. Unsigned packages may be distributed through Red Hat Network, but the **yum** updater must be forced to accept them. Signing packages is highly recommended and is covered in [Section B.2, "Digital Signatures for Red Hat Network Packages"](#).
6. If the package is changed in any way, including changing the signature or recompiling, the version or release must be increased incrementally. In other words, the NVRA (including architecture) for each RPM distributed through Red Hat Network must correspond to a unique build to avoid ambiguities.
7. No RPM package may obsolete itself.
8. If a package is split into separate packages, be extremely careful with the dependencies. Do not split an existing package unless there is a compelling reason to do so.
9. No package may rely upon interactive pre-install, post-install, pre-uninstall, or post-uninstall scripts. If the package requires direct user intervention during installation, it cannot work with Red Hat Network.
10. Any pre-install, post-install, pre-uninstall, and post-uninstall scripts should never write anything to stderr or stdout. Redirect the messages to **/dev/null** if they are not necessary. Otherwise, write them to a file.
11. When creating the spec file, use the group definitions from **/usr/share/doc/rpm-<version>/GROUPS**. If there is not an exact match, select the next best match.

12. Use the RPM dependency feature to make sure the program runs after it is installed.



### Important

Do not create an RPM by archiving files and then unarchiving them in the post-install script. This defeats the purpose of RPM.

If the files in the archive are not included in the file list, they cannot be verified or examined for conflicts. In the vast majority of cases, RPM itself can pack and unpack archives most effectively anyway. For instance, don't create files in a %post that cannot or will not be cleaned up in a %postun section.

## B.2. Digital Signatures for Red Hat Network Packages

All packages distributed through Red Hat Network should have a *digital signature*. A digital signature is created with a unique private key and can be verified with the corresponding public key. After creating a package, the SRPM (Source RPM) and the RPM can be digitally signed with a GnuPG key. Before the package is installed, the public key is used to verify the package was signed by a trusted party and the package has not changed since it was signed.

### B.2.1. Generating a GnuPG Keypair

A GnuPG keypair consists of the private and public keys. To generate a keypair:

1. Type the following command as the root user on the shell prompt:

```
gpg --gen-key
```

GPG Keypairs should not be created by non-root users. The root user can lock memory pages which means the information is never written to disk, unlike non-root users.

2. After executing the command to generate a keypair, an introductory screen containing key options similar to the following will appear:

```
gpg (GnuPG) 2.0.14; Copyright (C) 2009 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
```

```
Please select what kind of key you want:
  (1) RSA and RSA (default)
  (2) DSA and Elgamal
  (3) DSA (sign only)
  (4) RSA (sign only)
Your selection?
```

3. Choose the option **1** and then press **Enter**.
4. Choose the key size, which is how long the key should be. The longer the key, the more resistant against attacks the messages are. Creating a key of at least 2048 bits in size is recommended.
5. The next option will ask to specify how long the key needs to be valid. When choosing an expiration date, remember that anyone using the public key must also be informed of the expiration and supplied with a new public key. It is recommended to not select an expiration date. If an expiration date is not specified, you are asked to confirm your decision:

Key does not expire at all Is this correct (y/n)?

6. Press **y** to confirm your decision.
7. Provide a User-ID containing your name, your email address, and an optional comment. Each of these is requested individually. When finished, you are presented with a summary of the information you entered.
8. Accept your choices and enter a passphrase.



### Note

Like your account passwords, a good passphrase is essential for optimal security in GnuPG. Mix your passphrase with uppercase and lowercase letters, use numbers, and/or include punctuation marks.

9. Once you enter and verify your passphrase, the keys are generated. A message similar to the following appears:

```
We need to generate a lot of random bytes. It is a good idea to
perform some
other action (type on the keyboard, move the mouse, utilize the disks)
during the prime generation; this gives the random number generator a
better chance to gain enough entropy.
```

```
+++++.+++++.+++++. . .+++++.+++++.+++++.+++++. +++.
+++++.+++++.+++++. . .+++++.+++++.+++++.+++++. +++++.
```

When the activity on the screen ceases, your new keys are placed in the directory **.gnupg** in root's home directory. This is the default location of keys generated by the root user.

To list the root keys, use the command:

```
gpg --list-keys
```

The output is similar to the following:

```
gpg: key D97D1329 marked as ultimately trusted
public and secret key created and signed.

gpg: checking the trustdb
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
gpg: depth: 0 valid: 3 signed: 0 trust: 0-, 0q, 0n, 0m, 0f, 3u
gpg: next trustdb check due at 2013-08-28
pub 2048D/D97D1329 2013-08-27 [expires: 2013-08-28]
    Key fingerprint = 29C7 2D2A 5F9B 7FF7 6411 A9E7 DE3E 5D0F D97D 1329
uid                               Your Name<you@example.com>
sub 2048g/0BE0820D 2013-08-27 [expires: 2013-08-28]
```

To retrieve the public key, use the following command:

```
gpg --export -a 'Your Name' > public_key.txt
```

The public key is written to the file **public\_key.txt**.

This public key is quite important. It's the key that must be deployed to all client systems that receive custom software through **yum**. Techniques for deploying this key across an organization are covered in the *Red Hat Network Client Configuration Guide*.

### B.2.2. Signing packages

Before signing packages, configure the `~/.rpmmacros` file to include the following:

```
%_signature gpg
%_pgp_name B7085C8A
```

Replace the `_pgp_name` key ID value of `B7085C8A` with the key ID from your GPG keyring that you use to sign packages. This value tells **RPM** which signature to use.

To sign the package `package-name-1.0-1.noarch.rpm`, use the following command:

```
rpm --resign package-name-1.0-1.noarch.rpm
```

Enter your passphrase. To make sure the package is signed, use the following command:

```
rpm --checksig -v package-name-1.0-1.noarch.rpm
```



#### Note

Before running the `rpm --checksig -v` command, import the gpg key. See [Section B.2.3, "Importing Custom GPG Keys"](#) in the next section for more information.

You should see the phrase Good signature from "Your Name" in the output, with *Your Name* replaced with the name associated with the signing key.

### B.2.3. Importing Custom GPG Keys

For customers who plan to build and distribute their own RPMs securely, it is strongly recommended that all custom RPMs are signed using GNU Privacy Guard (GPG). Generating GPG keys and building GPG-signed packages are covered in the [Section B.2.1, "Generating a GnuPG Keypair"](#).

Once the packages are signed, the public key must be deployed on all systems importing these RPMs. This task has two steps: first, create a central location for the public key so that clients may retrieve it, and second, adding the key to the local GPG keyring for each system.

The first step is common and may be handled using the website approach recommended for deploying Red Hat Network client applications. To do this, create a public directory on the Web server and place the GPG public signature in it:

```
cp /some/path/YOUR-RPM-GPG-KEY /var/www/html/pub/
```

The key can then be downloaded by client systems using **Wget**:

```
wget -O- -q http://your_proxy_or_sat.your_domain.com/pub/YOUR-RPM-GPG-KEY
```

The **-o-** option sends results to standard output while the **-q** option sets **Wget** to run in quiet mode. Remember to replace the *YOUR-RPM-GPG-KEY* variable with the filename of your key.

Once the key is available on the client file system, import it into the local GPG keyring. Different operating systems require different methods.

For Red Hat Enterprise Linux 3 or later, use the following command:

```
rpm --import /path/to/YOUR-RPM-GPG-KEY
```

Once the GPG key has been successfully added to the client, the system should be able to validate custom RPMs signed with the corresponding key.



### Note

When using custom RPMs and channels, always create a custom GPG key for these packages. The location of the GPG key also needs to be added to the Kickstart profile.

The custom GPG key needs to be added to the client systems or the Kickstart installation may fail.



## Appendix C. Revision History

Revision 1.1-0	Wed Feb 1 2017	Satellite Documentation Team
Initial revision for the Red Hat Satellite 5.8 release.		