ZENworks Linux Management 6.5 Administrator’s Guide

ZENworks Linux Management 6.5 Administrator's Guide

Table of Contents

Introduction

Welcome to the ZENworks Linux Management 6.5 Administrator's Guide. This manual is dated June 1, 2004. Please contact your sales or support representative if you believe that this documentation is out of date or in error, or simply if you have suggestions for its improvement.

How to Use this Manual

This document consists of the following parts

- An introductory guide to installation and configuration, with a sample rollout procedure to give you a feel for the tools.
- An in-depth guide to the software and its capabilities.
- Reference material, including a troubleshooting guide and glossary.

This guide covers both the web and command line interfaces for ZENworks Linux Management. However, the primary focus is on the web interface. For additional information on the command line tools, read the **rcman** man pages or the --help flag on a particular command. For example, **rcman channel-add --help** will display help information for the channel addition tool. Entering **rcman --help** will get you a complete list of available commands.

We will assume that you know or can look up basic information about UNIX and Linux command line software and the RPM package format. Shell scripting and custom RPM package building are not required, but may be used in combination with ZENworks Linux Management.

Where a single line of text at the command line gets too long, we will wrap it to the next line and indent it slightly for clarity. Print versions of this document may also scale some entries to fit on a single page.

In places where we ask you to replace text with specific relevant information we will surround it with square brackets. For example, **rug activate [your-key] [your-email-address]**.

More Help

Additional help can be found in the man pages for all command line tools **rug**, **rcd**, **rcman**, **rcmirror**, and the various rce scripts. Additional information on the RPM package format can be found at http://www.rpm.org .

For information about technical support, see the Section called *Contacting Support* in Chapter 13 .

ZENworks Linux Management: An Overview

The situation is familiar: you, as a system or network administrator, have limited time and a limited budget, and want to maximize security and convenience as you distribute software updates and new programs to your Linux network. You don't have the budget or time to develop your own tools, and the standard package management systems are aimed at individual computers rather than complete networks.

ZENworks Linux Management lets you manage dozens or thousands of individual client machines from a central server located within your firewall for maximum convenience and security. Using the web interface or the command line tools, you can create groups of individual machines, and channels of software, then distribute the software to the machines. You can delegate authority to multiple administrators, follow progress with the built-in reporting system, and maintain your network with a minimum of cost and effort.

ZENworks Linux Management 6.5 is a completely new release and offers a substantial number of new features, including:

Enhanced access-control lists

> Three levels of permission: View, Modify, and Grant, for an increased number of objects, allow you greater control over the way in which your network is managed. ACLs now work more closely with activation keys so that Administrators, Channels, Groups or Machine Notifications may now be associated with an activation key.

Ability to check pending updates

> An Administrator can now see machines that have updates pending as well as the importance of those updates. For each machine the full list of pending updates is provided with information about each of the packages.

Package sets

> Software packages can now be grouped into sets such as "Ximian Desktop 2" and then manipulated as single packages. The package sets behave like packages with versions and dependencies.

Ad-hoc groups or machine sets

> Various criteria can be used to create arbitrary sets of machines. For example an administrator can select a group of machines that have updates pending and have CPU speeds greater than 1GHz, and immediately construct groups and act on them.

Enhanced transactions

> Scheduling has been improved to allow immediate or future execution. You may also add scripts to run before and after the transaction, and set download times separately from transaction times to avoid network congestion. Rollback transactions may be set to work with multiple machines at once.

Set client permissions from the server

> New centralized management tools let you browse hardware and software profiles and control client configuration files.

Server-side dry runs

> A transaction dry run can now be conducted on the server or client side. The server can check package, package set, and channel dependency information itself, without conducting processor-intensive analysis on the client.

Support for machine comparisons

> The comparison tool lists software discrepancies between two or more machines, allowing administrators to ensure uniform software distribution.

Choice of database backend

ZENworks Linux Management 6.5 allows you the choice of an Oracle or PostgreSQL database, so you have greater control of the cost of deployment. Database selection is covered in Chapter 11.

Caching server for multiple subnets

For networks with subnets in different physical locations, or with slow or intermittent connections, an intermediary server running rccache can save time and bandwidth by caching software and authentication tokens. To learn more, read Chapter 12.

Enhanced mirroring tool

The **rcmirror** tool now automatically creates package sets from product descriptions provided by the mirrored server. See Chapter 10 for more information.

In addition, the client offers additional features, notably:

Multiple server support

Client applications can now access multiple ZENworks Linux Management servers. As a part of this feature the command **rug set host** has been removed. The new command is **rug service-add [service-URI]**. Configuration files are migrated automatically to the new system.

Time-based rollback

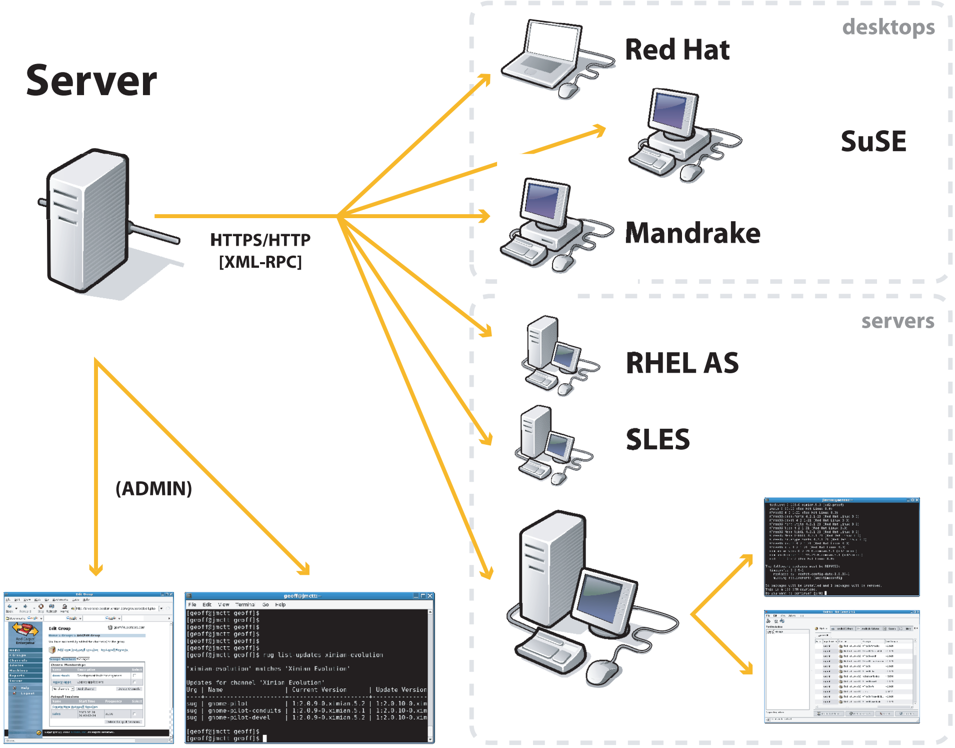
Rollback can be invoked locally through the **rug** command line interface, or centrally through the server administrative interfaces.

Expanded system information

Client machines can now send hardware, software and system information to the server, using the **rcd-modules** package. The **rcd-autopull** package has been removed.

ZENworks Linux Management Architecture

The simplest ZENworks Linux Management deployment consists of a single server and a single client machine. The client runs the Red Carpet Daemon, **rcd**, and fetches updates automatically, or at the user's direction, using the **rug** or **red-carpet** programs.



```
[geoff@jmctt geoff]$
[geoff@jmctt geoff]$
[geoff@jmctt geoff]$
[geoff@jmctt geoff]$
[geoff@jmctt geoff]$
[geoff@jmctt geoff]$
[geoff@jmctt geoff]$ rug list-updates ximian-evolution

'ximian-evolution' matches 'Ximian Evolution'

Updates for channel 'Ximian Evolution'
Urg | Name                  | Current Version       | Update Version
----+-----------------------+-----------------------+---------------
sug | gnome-pilot           | 1:2.0.9-0.ximian.5.2 | 1:2.0.10-0.xim
sug | gnome-pilot-conduits | 1:2.0.9-0.ximian.5.1 | 1:2.0.10-0.xim
sug | gnome-pilot-devel     | 1:2.0.9-0.ximian.5.2 | 1:2.0.10-0.xim

[geoff@jmctt geoff]$
[geoff@jmctt geoff]$
```

Figure 1. General System Architecture

The ZENworks Linux Management server uses an XML-RPC interface to communicate with a variety of clients and other systems. The XML-RPC interface feeds into the core components of the server, which work with Apache, a database, and the package repository.

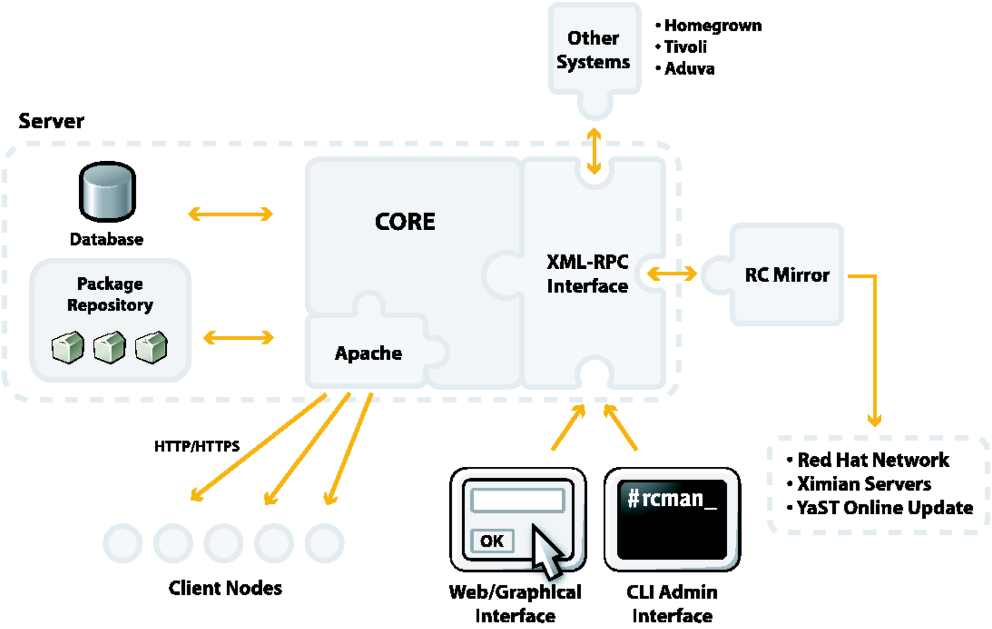


Figure 2. The ZENworks Linux Management Server

Similarly, the client daemon **rcd** uses an XML-RPC interface to accept commands from **rug** and **red-carpet**, remotely or locally. Additional modules provide the client with the ability to interact with the ZENworks Linux Management server for extended functionality.

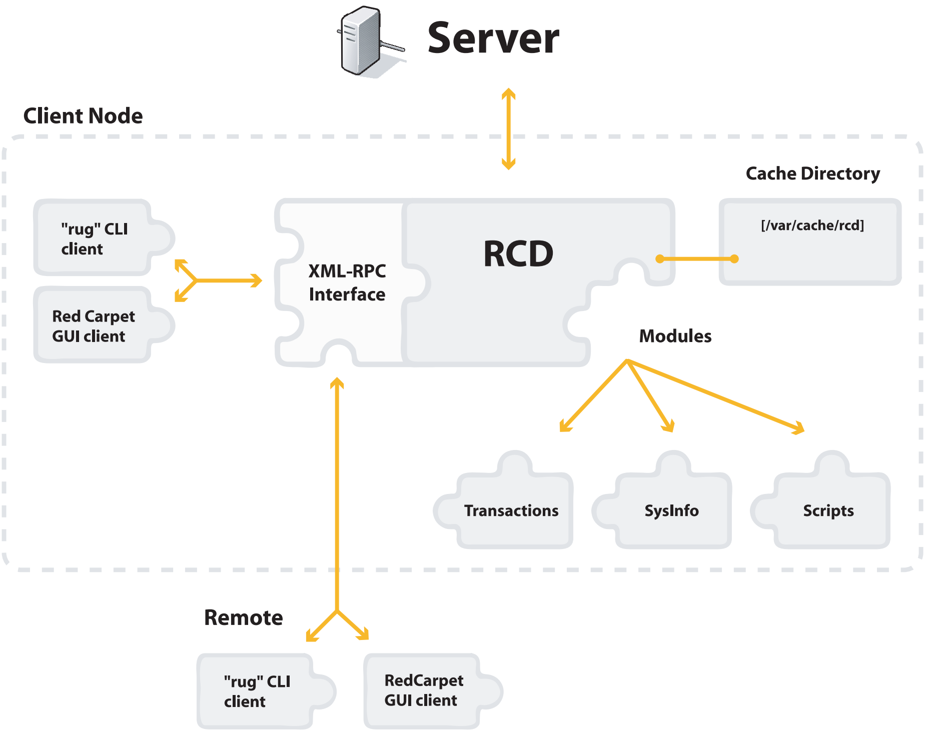


Figure 3. General Client Architecture

Ports and Listeners

The client listens on port 505 for server requests. The server listens on port 443. All communications are over SSL for security.

Chapter 1. Installing ZENworks Linux Management

This chapter will guide you through the process of installing the client and server portions of ZENworks Linux Management. Additional deployment information can be found in Chapter 2.

Installing and Configuring your ZENworks Linux Management Server

This section describes how to install a ZENworks Linux Management server. There are three steps in the server installation process: check the hardware and software requirements, install the packages, and run the initialization and configuration commands.

Server Software Requirements

A ZENworks Linux Management server has similar requirements to a web or database server, and should be run on a dedicated system running one of the following operating systems:

- SUSE Linux Enterprise Server 8, with Service Pack 3 or greater
- Red Hat Linux 7.3 or 9 with up-to-date kernel errata (kernel version of at least 2.4.20-28)
- Red Hat Enterprise Linux 2.1 AS, with up-to-date kernel errata
- Red Hat Enterprise Linux 3 AS or ES, update 1 or greater

Check the release notes for updates, notes, and changes to this list of supported operating systems.

We also require the following software versions:

Browser Versions

The web interface for ZENworks Linux Management works best with Internet Explorer 6.0 or greater, and Mozilla 1.4 or greater. Browsers based on supported versions of Mozilla, such as Galeon, Netscape, or Firebird, should work but are not guaranteed.

RPM Updates for Red Hat Systems

On both server and client systems, we suggest RPM updates. For Red Hat 8.0, we suggest rpm 4.1.1, and for Red Hat 9, we suggest rpm 4.2-1. Earlier versions have a relatively rare bug that can cause hangs during update, and subsequent database corruption.

Server Hardware Requirements

For an evaluation, with fewer than 100 clients, 2-3 Linux distributions, and a lab environment, we suggest:

- Pentium processor at 1 GHz

- 512 MB of RAM
- 10 GB disk space for package repository, 2 GB for database, and 5 GB for mirroring.

For a small deployment, with 100 to 500 systems, 3-5 Linux distributions, and some custom applications, we suggest:

- Pentium processor at 1.6 GHz
- 1 GB of RAM
- 25 GB of disk space for package repository, 5 GB for database, and 5 GB for mirroring.

You will need approximately two additional gigabytes of disk space for each additional operating system you plan to mirror and support. The default package storage location is /ximian, so in most cases / should be your largest partition. The /var partition holds the database, logs, and download caches for **rcmirror** and **rcd**. It will require less space than the package repository, but half to a quarter of the repository size is not unreasonable. If you plan to use **rcmirror** to any extent, you may wish to allocate 4 to 6 GB more.

If you prefer not to allocate so much space to /var, you can select a different cache directory in the **rcmirror.conf** file. See Chapter 10 for details.

Choosing a Database

You may use either PostgreSQL or Oracle 9i as a database. For the vast majority of deployments, PostgreSQL is more than satisfactory. However, if you already have an Oracle installation, ZENworks Linux Management can work with that database instead. See Chapter 11 for instructions.

Oracle database support is limited to the following configuration: an Oracle 9i database and either Red Hat 7.3 or Red Hat Enterprise Linux 2.1 Advanced Server (RHEL 2.1 AS). The recommended Oracle client is Oracle 9i client, Release 2.

Server OS Installation Options

We require a clean install of the server operating system. This section outlines the options we recommend during your operating system’s installation process.

SUSE Linux Enterprise Server OS Installation Recommendations

Install from the SUSE SLES 8 CD 1, rather than the unbranded UnitedLinux CD. The unbranded UnitedLinux CD will not work.

Within the YAST installer, select a *New Installation*.

When you reach the *Installation Settings* section, click *Software*, and confirm that the item *Default System* is selected. Do *not* select *Default System for United Linux*.

After completing the installation, install any required updates, errata, or service packs.

Red Hat 7.3 Installation Recommendations

When installing Red Hat 7.3 for use with Red Carpet Enterprise, select the following options:

- *Installation Type:* Server
- *Firewall Configuration:* No firewall
- *Package Group Selection:* Check at least *SQL Database Server* and *Web Server*

After installation, apply required errata, updates, or service packs.

Red Hat 9 Installation Recommendations

When installing Red Hat 9 for use with Red Carpet Enterprise, select the following options:

- *Installation Type:* Server
- *Firewall Configuration:* No firewall
- *Package Group Selection:* Under the *Servers* section, select at least: *Server Configuration Tools*, *SQL Database Server* and *Web Server*

After installation, apply required errata, updates, or service packs.

Red Hat Enterprise Linux 2.1 AS Installation Recommendations

When installing RHEL 2.1 AS for use with Red Carpet Enterprise, select the following options:

- *Installation Type*: Advanced Server
- *Firewall Configuration:* No firewall
- *Package Group Selection:* All required packages are installed by default.

After installation, apply required errata, updates, or service packs.

Red Hat Enterprise Linux 3 AS and ES Installation Recommendations

When installing RHEL 3 AS or ES for use with Red Carpet Enterprise, select the following options:

- *Firewall Configuration:* No firewall
- *Package Defaults:* Customize the set of packages to be installed
- *Package Group Selection:* Under the *Server* section, check at least the SQL Database.

After installation, apply required errata, updates, or service packs.

Server Installation

Upgrading from Earlier Product Versions

If you are upgrading from earlier product versions, be sure that your client systems are running the latest version of **rcd** before you begin. Consult the release notes for specific upgrade information.

You may install from a CD or online. If you install from a CD, your sales representative will provide you with a CD. If you install online, your sales representative will provide you with a license file and an activation key. You may wish to use the worksheet in Appendix B to keep track of the various bits of data you'll use as you deploy.

Client and Server Terminology

We will use the word "machine" to refer to a complete computer, most often to a client machine. In general, we will use the terms "client" and "server" to refer to whole computers, not just the software applications they run, which will be denoted as "client applications."

To Install from CD

This is the preferred method of installation. Before you begin the installation, make sure you have the server license file (`server.key`) ready, and become root with the **su** command.

1. Mount the CD according to your operating system's instructions.
2. Change directories to the location where you have mounted the disk.
3. Run the command **./rce-install**.
4. Read and agree to the terms of license.
5. When asked if you wish to use Red Carpet Express to install the software, select "No."

After you have run the installation program, proceed to the Section called *Start and Configure the Server*.

Additional information about the CD install may be found in the INSTALL file in the disc's root directory.

To Install Online

For this method of installation you will need a server license file and a client activation key. Your sales representative will supply you with both.

If you have not already done so, install the **rug** and **rcd** tools as follows:

1. Visit ftp://ftp.ximian.com/pub/redcarpet2/ and download the latest version of both **rug** and **rcd** for your operating system. Place the downloaded files in a directory by themselves.

2. Become root with the **su** command.
3. Enter the directory where you placed the files, and install the packages by using the command:

   ```
   rpm -Uvh *.rpm
   ```

Otherwise, upgrade the programs with the command **rug in rug rcd**.

Changes from Previous Versions

These installation instructions are geared to **rug** 1.5 and later. For earlier versions, you will need to use **rug set host** instead of **rug service-add**. To get the most from your ZENworks Linux Management deployment, use the latest stable version of **rug** and **rcd** for your operating system.

Once you have the latest versions of the **rug** and **rcd** packages installed, do the following:

1. Become root with the **su** command.
2. Start (or restart) the **rcd** service with the following command:

   ```
   /etc/init.d/rcd restart
   ```

3. If you do not plan to use signed SSL certificates, turn off certificate ID verification. This will also create the `/etc/ximian/rcd.conf` file.

   ```
   rug set require-verified-certificates false
   ```

   For information about SSL certificates, see the Section called *SSL Certificates*.

4. Point the client to your server. This will create the `/var/lib/rcd/services.xml` file:

   ```
   rug service-add [https://your-server.com/data
   ```

5. You may also wish to remove the old default server:

   ```
   rug service-delete http://red-carpet.ximian.com
   ```

6. If you are using a proxy, you will need to run the following commands as well:

   ```
   rug set proxy-url [http://hostname:port]
   rug set proxy-username [username]
   rug set proxy-password [password]
   ```

   ### Network Proxy Support

   If your corporate network uses a proxy server to connect to the Internet, or if your clients need to use a proxy to access your ZENworks Linux Management server, be sure to check the type of proxy. HTTP and HTTPS proxies, using basic or digest authentication, are supported. NTLM or SOCKS proxies such as Microsoft Proxy 2.0 are not supported.

7. Then, activate your system with the following command:

   ```
   rug activate [activationkey] [email]
   ```

8. You should receive the following message:

   ```
   System successfully activated
   Refreshing all channels
   Download complete
   ```

9. Check the list of channels with the **rug ch** command. Then, subscribe to the channels for `zlm-server` and `redcarpet`, as well as any others you wish to update. We suggest that you also subscribe to your operating system channel, so that installation can pull in packages as needed. For example, for Red Hat 9 you would type:

   ```
   rug sub redhat-9-i386
   rug sub ZENworks65
   rug sub redcarpet2
   ```

Your server is now ready for installation.

1. Run the following command to install the necessary packages. The -u flag permits installation from unsubscribed channels.

   ```
   rug in -u zlm-server zlm-server-mirror zlm-server-cli
   ```

2. The system will tell you what packages it will install, and what packages (if any) it needs to remove. Press **Y** to confirm the installation.
3. Once the install is complete you should see a "Transaction finished" message.
4. If you have an OEM or reseller customization and branding package, install it now.
5. Place the server license file in the directory `/etc/ximian/rcserver` with the file name `server.key`. Set the file ownership to `apache:apache` by running the command:

   ```
   chown apache:apache /etc/ximian/rcserver/server.key
   ```

6. Proceed to the Section called *Start and Configure the Server* below.

Start and Configure the Server

Now that you have installed the individual packages, you must set up the system database and start the ZENworks Linux Management server services. The **rce-init** or **rce-upgrade** process will configure and start the necessary services, such as Apache and PostgreSQL. Once you have all the accessory services up and running, you can control the services as you would a web server, using standard **/etc/init.d** or **/sbin/service** commands. This is not, however, generally necessary.

To begin, open a terminal and become root.

Run the command:

```
/usr/sbin/rce-init -U admin@address -P password -R "Real Name"
```

We suggest that you use the **`-U user@name`**, **`-P password`**, and **`-R "Real Name"`** options for **rce-init** to set the email address, password, and real name of your first administrator account. If you do not use the command line options, you must finish initialization as follows in the web interface:

1. Visit the ZENworks Linux Management server in your web browser (**`https://localhost`** if you are using the system directly).
2. Enter your email address, real name, and password.
3. Click the the **Initialize Database** button.

The Organization ID

When you log in to your server, it will display your Organization ID string, which you may wish to use later on in the client activation process. This number will be displayed every time you log in to the web interface.

The ZENworks Linux Management server is now ready to be administered with **rc-man** or the web interface.

Installation of Client Systems

Whether you are installing one client machine or hundreds, the process is similar:

1. Install client software packages.
2. Start daemon.
3. Configure or copy configuration files and restart **rcd**.
4. Activate client.

The following sections of this manual will show you how to install, configure, and activate client machines. First, we will explain in detail the process of installing and configuring a single machine, then discuss ways to automate the deployment process. Finally, we will cover activation and configuration of the clients.

Installing One Client Machine

To install the client software on a single machine:

1. Get the latest stable version of **rug**, **rcd**, and **rcd-modules** for the client's operating system. You can find the packages at ftp://ftp.ximian.com/pub/redcarpet/ or on your ZENworks Linux Management CD.

2. If you are downloading the files, place them in a directory by themselves. Otherwise, you can leave them on the CD and install from there.
3. Become root with the **su** command.
4. Install the packages by using the **rpm -Uvh** command. If you are not in the same directory with the packages, specify the complete path to the files.
5. Run the command **/etc/init.d/rcd start** to start **rcd**.

   Daemon Starts on Boot

   The client daemon **rcd** does not start as soon as it is installed. However, the daemon *does* add itself to startup services, and will run the next time the machine boots.

6. Unless you have purchased or generated an SSL certificate, turn off certificate checking. Changing a setting creates the file `/etc/ximian/rcd.conf`. To turn off certificate checking:

   ```
   rug set require-verified-certificates false
   ```

   See the Section called *Security, Signatures, and Certificates* for information about installation security and certificate checking.

7. Point the client to your server. This will create the `/var/lib/rcd/services.xml` file:

   ```
   rug service-add https://[your-server]/data
   ```

8. You may also wish to remove the old default server:

   ```
   rug service-delete http://red-carpet.ximian.com
   ```

9. If you are using a proxy, you will need to run the following commands as well:

   ```
   rug set proxy-url [http://hostname:port]
   rug set proxy-username [username]
   rug set proxy-password [password]
   ```

Upgrading and the Configuration File

rcd versions 1.5 and later support multiple servers, and store server information in the file `/var/lib/rcd/services.xml`. Earlier versions supported only one server and stored the information in `rcd.conf`. To get a list of currently available servers, enter the command **rug services-list**.

If `services.xml` does not exist, **rcd** will read the server from the `host` line of `rcd.conf` and create `services.xml`.

Once `services.xml` has been created, changes to `rcd.conf` will not affect the server accessed by the daemon.

Once you have installed and configured the machine, you must activate it against your server. See the Section called *Client Activation* for more information.

Installing Multiple Client Machines

There are two common situations for large numbers of installations: deploying completely new machines, and adding software to existing machines. If you are deploying completely new operating systems, you can install client software along with the operating system using your distribution's deployment automation tools (Kickstart or AutoYaST, for example). In that case, you will add the client packages **rug**, **rcd** and **rcd-modules** to the install set, and add the setup commands to the post-install scripts. Check the documentation for your install automation tools for details.

For client machines which are already deployed, you may find it more convenient to place the individual package files on a network shared volume and install them with a combination of shell scripts and command line package tools.

You can create an **rcd.conf** file with a text editor, but in most cases it is easier to run a script to configure the client software. If you edit the configuration file you must restart the daemon to force it to read the changes.

Client Activation

For ZENworks Linux Management to work, the server system must be able to recognize individual client machines. There are two different ways that you as a system administrator can register client machines with the server, each suited to different sorts of installations. They are:

- Activation with a key and an activation profile. This is the preferred method, especially for large deployments.
- By placing an organization ID token in the `/etc/ximian/rcd.conf` file for each client machine.

An activation profile consists of a key and a set of group, channel, and administration permissions. The permissions are automatically applied to each machine which connects to the server and presents the key. A machine can activate using either **rug** or **red-carpet**.

There are two types of keys: those which can be used only by a single client machine, and those which can be used by an unlimited number of clients (as long as you have client licenses for them). You can cancel an invitation if it has not yet been accepted, and you can cancel a multi-use key at any time. Any clients which have already registered with a key are not affected by cancellation.

You can learn to create keys and set the permissions associated with them in the Section called *Activations* in Chapter 5.

Once you have created one or more keys for your client machines, you can activate. To activate, use the command **rug activate** or the Red Carpet graphical client application. Using the command line allows you to write scripts that help automate the deployment process.

If you do not wish to use activation keys, you can set the organization ID token on each of your client machines. The organization ID is a unique identifier number created when the ZENworks Linux Management server is initialized. It is displayed every time you log in to the web interface. Any client presenting the organization

ID will be added to the list of available machines, but with no permissions set. The server administrator then decides whether to allow the client to update.

To set the token, copy the organization ID token from the web interface, and on each client run the command:

```
rug set org-id [your-org-id-value-here]
```

Alternately, edit the `rcd.conf` file to contain the line

```
org-id=[your-org-id-value-here]
```

ⓘ **Email Addresses and Organization ID**

If you register a system using the org-id method, it will not have an email address in the machine list (**rcman ml** or the *Machines* tab in the web interface). To add or change this information, you must use the **rcman machine-update** command or the web interface.

See below and Chapter 9 for more information about client machines and client settings.

Configuring Clients and the rcd.conf File

This section will cover client configuration in greater detail, and explain the configuration files for **rcd**.

If you prefer to use a configuration file rather than a series of commands to configure your client machines, you will need to create an `/etc/ximian/rcd.conf` file and a `/var/lib/rcd/services.xml` file. The first file is only a few lines long, and will look something like a Windows INI file. It has one or more sections, denoted with square brackets: `[Section]`. Each section contains one or more tokens, set to equal a particular value: `token=value`.

The **services.xml** file is not designed for hand-editing. If you prefer, you can create the file by configuring a single client, and then copy it to additional machines. You can use the commands **rug service-list**, **rug service-add**, and **rug service-delete** to display, add to, and delete from the list of servers your client will access.

ⓘ **File Formats**

The `rcd.conf` format is similar to the other configuration files you will encounter in the rest of the product. In particular, the configuration file for **rcmirror** is similar, although larger and more complex.

You can change the settings reflected in the rcd.conf file using the **rug set** and **rug get** commands. However, if you prefer to do things the old-fashioned way, you can edit the file yourself:

1. Open a text editor such as **emacs**, **vi**, or **gedit**.
2. Enter **`[Network]`** to designate a section about the network.
3. Enter the token **`require-verified-certificates`**. Set it to **`true`** if you have purchased or generated an SSL certificate for your ZENworks Linux Management server, and **`false`** otherwise (the Section called *Security, Signatures, and Certificates* has instructions on installing an SSL certificate.) Changing this line is the same as using the command **rug set require-verified-certificates false** (or true).
4. If you do not wish to use package signatures, set the token **`require-signatures`** to **`false`**. To use package signatures for packages not shipped by Ximian, you must add the public key to each client's rcd-specific keyring. This is generally impractical, as there is no convenient way to access the rcd keyring through **rug**. If you plan on shipping software that is not provided by Ximian, it is best to check package signatures before you add them to your server, and then rely on client authentication and SSL certificates to secure the communications between your clients and your server.
5. For maximum security, add a **`[System]`** section with the line **`remote-enabled=false`**. This forces all client applications to be run from the local machine. If you wish to allow remote connections, note that they occur over port 505 if initiated by the server, and over port 443 for standard SSL connections.

To check your current settings, run the command **rug get**, or check the rcd.conf man page for a list of all the settings you can set.

A very simple file could look like this:

```
[Network]
require-verified-certificates=false

[Cache]
enabled=true
```

A more complex file might look like this:

```
[Network]
require-verified-certificates=false

[Cache]
enabled=true

[System]
cache-cleanup=true
cache-size-in-mb=500
debug-level=1
repackage=false
syslog-level=1

[Server]
remote-enabled=false
```

Security, Signatures, and Certificates

ZENworks Linux Management uses SSL certificates and cryptographic signatures to verify the identity of the server and the packages it ships, respectively. All ZENworks Linux Management communications are encrypted, but there are several ways that you can increase your security with better cryptography management.

SSL Certificates

An SSL certificate is a way for your server to identify itself to clients and enable secure transactions. There are two major kinds of certificates: those signed by a certificate authority (CA), and those you generate and sign yourself.

Certificates signed by a CA are more secure, because the CA guarantees the identity of the server. However, signed certificates must be purchased. If you prefer, you can generate your own certificate and sign it yourself. A self-signed certificate does not provide the same guarantee of identity, but it still prevents others from eavesdropping on client-server communications.

To purchase a certificate, consult a certificate authority such as Thawte or Verisign. To generate a certificate and sign it yourself, use the **openssl** command or another cryptography tool. See the man pages for **openssl** for more details.

Once you have a certificate, you must install it. Place the private portion of the certificate in the same location you would for Apache, which will be /etc/httpd/conf/ssl.crt/server.crt (for Red Hat) or /etc/apache2/ssl.crt/server.crt (for SUSE). Put the public portion of the certificate in `$prefix/share/rcd/ca` for each client. The value of the prefix will change depending on the operating system.

If you have purchased a CA-signed certificate, set the **require-verified-certificates** option to **true** in the client machines:

```
rug set require-verified-certificates true
```

An `rcd.conf` file set that way would have these lines in it:

```
[Network]
require-verified-certificates=true
```

If you have generated and signed your own certificate, set that token to "false."

⚠ **Security Fences**

If you have an improperly installed or unsigned SSL certificate, and the **require-verified-certificates** token on your clients is set to **true**, every transaction will fail.

Package Signatures

Ximian packages come signed by Ximian, and the client applications recognize the Ximian GPG keys. However, non-Ximian packages, including those generated by you or by third parties, may generate spurious warnings if the keys are unrecognized. To prevent spurious warnings or rejected installs, add your public keys to the keyrings on the client machines.

On systems using RPM 4.1 or later, use the command **rpm --import filename**, where "filename" is a file containing your key or keys. For systems using RPM versions earlier than 4.1, use the command **gpg --import filename**.

Signing packages, like building them, is not within the scope of this manual. To learn how to sign packages, see the manual pages for the **rpm** tool.

Chapter 2. Example Rollout Procedure

In this example, we will use ZENworks Linux Management to maintain a network of Linux systems that includes servers, a number of highly-skilled engineers, and a few non-technical users in administrative and operations roles. For simplicity, the example will be based entirely on Red Hat Linux 8.0 and will focus on the command line rather than the web interface.

This example will not be identical to your deployment, but it should give you an outline of the choices you have. You can find additional examples in the Section called *An Example rcmirror Deployment* in Chapter 10. For an example of how to use package sets, see the Section called *Examples of Using Package Sets* in Chapter 4.

Planning

The first step in deployment is planning. Although many of the decisions you will make at this stage can be altered or adjusted in the future, it will help to have a clear set of actions laid out beforehand.

Group and channel membership

Decide on the number of groups you will need and how you wish to divide them. Try to predict the sorts of software you will need to distribute.

In our example network, there are three general types of machines to take care of: servers, which require remote administration, engineering workstations, which are administered, sometimes recklessly, by their users, and non-technical workstations which are used but generally not administered by their users. For all machines, we want security and operating system updates installed automatically whenever they are available. Office workstations will also get access to optional updates and additional tools. The engineering team will want developer tools, as well.

Later, it may be advisable to grant administrator access to some of the engineering team members, so they can distribute their own software to each other.

Our example will use three groups, one for each type of machine. The servers will get security updates only; the workstations will get security updates and the optional software channel, and the engineering workstations will get those two, plus a development channel.

Activation method

For most deployments, multi-use keys are the best activation tool, and we will use them in our example. Invitations might be good for the engineering workstations, and the organization ID method would be acceptable for the servers, if there were only a few of them and we wanted to give them individual attention. However, for maximum automation and convenience, the multi-use key is the obvious choice.

Install process for existing client machines

You will need to decide on a procedure for installing the software on Linux clients that already exist. Our example uses a shell script and an NFS shared drive.

Install process for new client machines

Even if you are not deploying to new machines now, it will help to have a plan for the addition of new machines to your existing ZENworks Linux Management network.

If you are using OS installation tools like AutoYaST or Kickstart you can add the required packages to the install set, then have post-install scripts run configuration commands and start services as needed. If you are installing custom machines or adding them one by one, it may be simpler to do the client software installs by hand as well.

Database type

In most cases, PostgreSQL is more than adequate. Environments with an existing Oracle deployment, however, may find it convenient to use Oracle instead. Our example will use the default PostgreSQL database.

Administrator accounts

How many administrators do you plan to have, and what privileges will you give them? For this example, there will be only one.

Software mirroring goals

Decide now how many operating systems you wish to support, and how many different channels you wish to use. You will need at least an additional 2GB of disk space per OS supported. You may wish to include future versions of operating systems in your planning, although it is simple enough to add disk space later. For more information about software mirroring, see Chapter 10.

Security policies

Choose your policies on package signatures, SSL certificates, and remote access. Especially with package signatures, you will want them in place before you need to ship software.

The Installation Worksheet

You may wish to use the worksheet in Appendix B to keep all your installation data organized.

Server Install

Now that your plan is complete, you can begin the deployment. The first step will be to provision and install the server. Our example administrator starts with a fresh install of Red Hat 8.0, with 10GB of disk space. She gets the ZENworks Linux Management license file **server.key** from her email Inbox and saves it to the hard disk. Then, she inserts the CD and enters:

```
mount /mnt/cdrom
cd /mnt/cdrom
./rce-install
```

Note that on SUSE distributions, the mount point will be **/media/cdrom** rather than **/mnt/cdrom.**

After agreeing to the terms of license, she enters the path to the license file and her activation code, so that she can install any necessary updates that were not on the CD. Then, she waits for the install to finish.

Now she initializes the server:

```
/usr/sbin/rce-init -U username@email -R "Real Name" -P password
```

With the software installation part complete, she proceeds to configure it and prepare it for clients.

Server Configuration

The client install process will end with clients contacting the server, and the server needs to be prepared for that contact, and ready to sort the clients into groups, assigning them group membership and channel access as appropriate. That means we need to create groups, channels, and activation profiles.

To create groups, our administrator enters the following commands:

```
rcman group-add --desc="Engineering Department" engineers
rcman group-add --desc="Server Farm" servers
rcman group-add  --desc="Office and Operations Team" office-ops
```

While she is creating groups, she sets notifications for them. A notification allows her to be alerted by email for transaction failures or for all transactions performed by a machine or a group of machines. She chooses to be alerted to all server transactions, and to failed transactions from the office workstations group:

```
rcman group-addemail servers [my@address] All
rcman group-addemail office-ops [my@address] Failure
```

Another way to check on system performance is to use reports, which are covered in Chapter 7.

Then, she creates channels:

```
rcman channel-add --desc="Security and OS updates everyone needs" "Security" security
rcman channel-add --desc="Optional updates and extra software" "Extra Software" extras
rcman channel-add --desc="Software development toolkits" "Developer Tools" devel
```

With channels and groups in place, she now needs to create activation profiles to put the group and channel privileges onto the right machines as they activate. To create activation profiles, she enters these commands:

```
rcman act-add --key=server-key
rcman act-add --key=office-key
rcman act-add --key=engineering-key
```

She has chosen her own activation keys so that they are easy to remember. Each key is unique and also serves as a name for the profile.

Activation profiles, when first created, are nearly identical: they do not provide access to anything. To change that, our administrator adds groups:

```
rcman act-addgroup server-key servers
rcman act-addgroup office-key office-ops
rcman act-addgroup engineering-key engineers
```

Activation Changes

Remember that changes to an activation do not affect machines that have already activated. An activation is like a stamp applied to a machine a single time. If you change the activation, machines you activated earlier are not affected, so you must re-activate them to apply your changes.

Then, she adds channels. All three profiles get the security channel:

```
rcman act-addchannel server-key security
rcman act-addchannel office-key security
rcman act-addchannel engineering-key security
```

The office and engineering groups get the extras channel:

```
rcman act-addchannel office-key extras
rcman act-addchannel engineering-key extras
```

Finally, only the engineers get the developer tools channel:

```
rcman act-addchannel engineering-key devel
```

Preparing Transactions

The next step is preparing a transaction to ship a package:

1. Add one or more new packages to a channel.
2. Create a transaction and set its start time and repeat frequency.

3. Add the channel or package to the transaction, and decide on an action type (update or install).

First, our administrator adds a package to the security channel. The package file is currently on the NFS shared disk:

```
rcman channel-addpkg --targets=redhat-80-i386 --importance="suggested"
          --desc="Testing the Security Update Mechanism" security
          /mnt/nfs/admin/rce/frobulator-1.0.1-i386.rpm
```

She has chosen this particular update as the first one because it is non-critical. Most of the software shipped through this channel will be much more important, but for a first transaction, this minor update is ideal.

Packages and Channels in Transactions

Transactions may ship individual packages or whole channels. For our example, a channel makes more sense. Once it is in place, an administrator could put any security update into the channel and have it shipped without having to alter the transaction. Having a package-based transaction is useful for tighter control of the shipped software, or when you need to update the same packages frequently, such as when building regular beta versions during software development.

Then, she creates a transaction. Because it is a test transaction, she adds the **`--dry-run=1`** flag and sets it to start at 4:30 PM. She can watch it happen, then update it when she is ready to run it as a ’live’ transaction.

```
rcman trans-add regular-security-update --dry-run=1 starttime="2004-02-23 16:30"
```

The next step is to associate the channel with the transaction, and select an action type. Our transaction name is "regular-security-update" and our channel is "security." The action argument for this transaction will be "U," an update transaction.

```
rcman trans-addchannel regular-security-update security U
```

Transaction Order

In this example we create a transaction first, and then add clients to it. You can also set up clients, group them, and add them to a transaction. In the web interface, this is most easily accomplished with the machine set, as covered in the Section called *Working with the Machine Set* in Chapter 5 .

Client Install

Once the server is ready, it's time to install the clients. The process will be:

1. Install **rcd** and **rug**.
2. Start the daemon.
3. Configure the daemon.
4. Activate the machine, so that it has the appropriate permissions on the server.

All of these items can be done at once, with a script, and all the machines will have the same settings, although they will use different activation keys depending upon their group membership.

Our example administrator has already decided to install the packages from an NFS mounted disk, which conveniently happens to be available to all clients as the central network storage device. She copies the files she will need to their own directory there: packages for **rcd**, **rug**, and **rcd-modules**.

For every client machine, she will perform the following steps, either by hand or with a script:

1. Log in remotely.
2. Install the packages with the **rpm** tool:

   ```
   rpm -Uvh /mnt/nfs/admin/rce/*.rpm
   ```

3. Start the daemon:

   ```
   /etc/init.d/rcd start
   ```

4. Disable SSL certificate checking:

   ```
   rug set require-verified-certificates false
   ```

5. Change the services available to add the new ZENworks Linux Management server, and remove the default server:

   ```
   rug service-add [https://my-server.company.com/data]
   rug service-delete http://red-carpet.ximian.com
   ```

More Information on Settings

For details about settings and the configuration file `rcd.conf`, read the Section called *Configuring Clients and the rcd.conf File* in Chapter 1 , the Section called *Preferences and Preference Management Tools* in Chapter 9 , or the rcd.conf man page.

Before she logs out of the machines, she has one more step. The final step varies because the three groups of machines will use three different activation keys.

- For the server machines:

  ```
  rug activate server-key [admin@company.com]
  ```

- For the office machines:

  ```
  rug activate office-key [admin@company.com]
  ```

- For the engineering workstations:

  ```
  rug activate engineering-key [admin@company.com]
  ```

As the machines activate, they acquire the appropriate group memberships and channel access privileges. The client deployment is complete, and all that remains is to associate the clients with the transaction we created earlier.

She can do this using groups:

```
rcman trans-addgroup regular-security-update engineers
rcman trans-addgroup regular-security-update servers
rcman trans-addgroup regular-security-update office-ops
```

Note that the connection made here is between individual machines and the transaction. All the current members of the groups will conduct the transaction, but future additions to the groups will not. If our administrator adds to the groups, she will have to add the individual machines to the transaction, or add the whole group again.

At 4:30 PM, the server will contact individual machines, instructing them to refresh their package lists and update schedules. The machines which have old versions of the "frobulator" package will calculate dependency information and prepare to download the package and any available dependencies. Then, they will stop. The transaction reports will note that the transaction was a dry run, and whether it would have failed due to unavailable dependencies.

Satisfied that the update will work as expected, our administrator updates the transaction. She sets the start time to one which will not interfere with the daily use of the machines, adds a frequency of two days, and sets the dry-run value to 0:

```
rcman trans-update --starttime="2004-02-23 00:30"--frequency="2 days"
        --dry-run=0 regular-security-update
```

The transaction is ready. At 12:30 AM, all client machines will check for updates in the "security" channel, and, if they have an outdated version of "frobulator," install a new one.

In addition, the machines will continue to check for updates every 48 hours, so any packages added to the channel will be updated without additional transaction work.

Chapter 3. Administrators and Permissions

ZENworks Linux Management 6.5 includes a more complex permissions and security mechanism than previous versions did. The new mechanism is based on three levels of access: Grant, Modify, and View.

Permissions can be granted for different portions of the organization, which we will refer to as objects: administrator accounts, channels, groups, machines, packages, and activations.

Organization Administrator Privileges

The first administrator is an organization administrator, and may grant organization administrator status to other administrator accounts. An organization administrator gets all permissions for all objects in the organization, no matter what the actual permissions settings say.

For multiple-organization deployments, a superadministrator account can be created, which is equivalent to organization administrator status in all organizations.

Grant permission allows you to grant access to an object. By default, you get the Grant permission for any object you create. Grant permission includes the Modify and View permissions.

Modify permission allows you to edit a given object. For example, if you have Modify access to a channel, you can add packages to it. Modify permission includes View permission.

View permission is read-only access to a given object. For example, an administrator could be given Grant permission for channel A, but only View permission for channel B. She will then know what software is available in channel B, and won't bother to ship it in channel A. However, she doesn't need Modify or Grant access to the channel.

Granting Access Requires Two Permissions

To change two objects at once, an administrator must have appropriate permissions for both objects. For example, if Anne wants to have John administer channel A, she must have Grant permission for the channel, and Modify permission on John's account.

Controlling Account Permissions

The permissions editing page allows you to change the way a particular administrator accesses groups, channels, and activations, and the way that other administrators access the account you are editing. To find the administrator editing page, click the *Admins* section, then the name of the administrator whose permissions you wish to edit. There are two tabs of information about an account that you may edit: *Basic* information, such as name and email address, and *Permissions*. Click the *Permissions* tab.

The *Permissions* tab consists of a series of tables describing the objects which the account may access, and what permissions the account has on the objects. At the bottom of each table you will see a drop-down list, from which you may add additional items to the tables of accessible objects.

Other Ways to Set Permissions

You can also change these settings in the individual pages for each object. For example, to add Jane Smith to your list of administrators for the "Workstations" group, you could click on Jane's account in the Admins page, or click the Workstations group in the Groups page.

Groups

Set the groups that this account may manage.

Channel Memberships

Sets the channel permissions for this account.

Administrator Accounts

Change the accounts this administrator may access.

Activations

Set the account's access to activation profiles.

Managing Administrator Accounts with the Command Line Tools

You can control administrator access to your server using rcman as well as the web interface. The commands for administrator account management are named in a similar fashion to those for package, channel, and group management. They are:

admin-add (aa)

Add an administrator. The arguments for this command are the email address, password, and real name of the administrator. This administrator has no permissions yet: you must grant them later by adding the account to various objects.

admin-del (ad)

Delete an administrator account. Takes a single argument: the email address of the administrator.

admin-list (al)

Lists all administrators in your organization.

admin-listacts (alv)

Lists the activations available to your administrator account. Use the the **--admin=** flag to specify an administrator other than yourself. For other information, use the commands **admin-listchannels** (alc), **admin-listgroups** (alg), and so forth.

admin-addadmin (aaa)

Use this command, along with **act-addadmin** (vaa), **channel-addadmin** (caa) and **group-addadmin** (gaa), to edit permissions for an administrator account. The argument order here is particularly important: in the following example, the administrator "admin1" is granted permission to modify the account "admin2":

```
rcman aaa admin1 admin2 modify
```

adminupdate (au)

> Update the information provided at account creation: name, email address, and so forth.

Check the man page and the output of the commands **rcman --help** or **rcman commandname --help** for additional details on these commands.

Chapter 4. Organizing Software: Channels and Packages

This chapter covers software management with ZENworks Linux Management, including control of channels, packages, and package sets.

Introducing Channels and Packages

Channels are groups of software packages that you want to ship together. To see a list of channels you have created, click the *Channels* item in the menu on the left side of the ZENworks Linux Management web interface. To list channels at the command line, use the **rcman channel-list** command.

You can edit an existing channel by clicking on its name or with the **rcman channel-update** command.

To create a new channel using the web interface:

1. Click the *Create New Channel* button.
2. Enter a name and, optionally, a short description for your channel.
3. Choose a nickname, or alias, for your channel. The alias is used when manipulating the channel with the command line tools. It should be unique and short, and may not contain any spaces.
4. If you wish, choose a channel icon. The server will provide a default icon if you do not select one. The channel icon is displayed in the GUI client application.
5. Select a channel priority number. Channel priority affects the choice of channels when a client is selecting software to fulfill dependencies. The higher the number, the more important the channel is considered. For the most part, you can leave this value at 0.
6. When you are ready, click *Save*.

To add a channel using the command line, use the command **rcman channel-add** tool with the name and alias of the channel as arguments. A description may be added with an option flag. For example:

```
rcman -U username channel-add "Developer" dev --desc="Developer tools"
```

Using Channels from Ximian

Ximian provides channels such as the operating system and Ximian products, which you can use in addition to the software you ship in your own channels. Check Chapter 10 for information on using channels provided by Ximian.

For a complete list of the options for channel creation, use the command **rcman channel-add --help**. There is an additional option available at the command line that is not available in the web interface: Base Path.

Using Channels with Packages

Once you have created a channel, you are ready to put it to work, adding packages and shipping them.

Every package you distribute must be in a channel. To work with packages, enter the channel editing page and then click the *list packages* link. The package list shows the name, version, and the package summary for each package available. An equivalent list is available with the **rcman channel-listpkgs** command.

You can get more information about a package by clicking its name, or remove it from the channel by clicking the check box on the row for the package and then pressing the *Delete* button. The equivalent command for deletion is **rcman channel-delpkg**.

Multiple Target Platforms

You may choose more than one target platform for a given package if you wish. For example, you might ship the file "manedit-0.5.6-1mdk.i586.rpm" for Mandrake Linux 8.1 and 8.2 on x86 machines, but ship "xslide-0.2-363.noarch.rpm" for all RPM-based machines.

Adding Packages

To add a package to a channel, click the *Add Package* button. This presents a form where you can select the package file to add from your local file system by pressing the *browse* button. In addition, specify:

- The target platforms supported by the package. Select from the list of available platforms on the left and click the *Add* button to move them across to the list of target platforms for this package. To remove an item from the list on the right, select it there and click the *Remove* button.
- The package importance. From most important to least, choose from urgent, suggested, feature, or minor.
- Optionally, a note indicating why you are shipping this package.

Descriptions and Summaries

Package shipped in ZENworks Linux Management have three pieces of descriptive information:

Package Summary

Short description of the package contents provided by the software packager. For example, the package summary for the Evolution package is "Ximian Evolution integrates email, calendaring, meeting scheduling, contact management, and task lists, in one powerful, fast, and easy-to-use application." This information is displayed in the detailed package information page in the web interface, as the output of the command **rug info [packagename]**, and in the GUI client application.

Package Description

A more detailed description of the package contents, provided by the software packager. For example, the description of the httpd package is "Apache is a powerful, full-featured, efficient, and freely-available Web server. Apache is also the most popular Web server on the Internet." This information is displayed in the detailed package information page in the web interface, as the output of the command **rug info [packagename]**, and in the GUI client application.

Update Description

Additional comments provided by the ZENworks Linux Management administrator, usually describing the reasons for shipping the package. This information appears in the *Basic* package information tab, and is not displayed to clients.

When you have filled in all the fields, press the *Add* button to upload the package.

Packages with User Licenses

Most software packages include a license file, and some set a specific license display flag. When that flag is set, installation tools will ask the user to agree to the terms of license before installation. The license flag is set at build time and not by ZENworks Linux Management. To learn about building packages with license flags, see the man page for the **rpm** command.

If you use the license display flag for a package, that package must be installed interactively.

If you are working with more than a few packages, the web interface becomes cumbersome. Using **rcman channel-addpkg** makes it much easier to add several packages at once, because at the command line you can use wildcards to specify a whole directory of software.

Upgrading the Kernel

The ZENworks Linux Management server will detect kernel packages and handle them properly: they are marked for parallel installation rather than upgrade, and the necessary scripts to update the bootloader are run by the daemon.

The **rcman channel-addpkg** command uses the **--targets** flag to list target platforms, and the **--desc** flag to specify a short description of the update. Following those flags, enter the alias of the channel to which you will add the package, and the filename of the package itself. For example:

```
rcman -U my@username.com channel-addpkg --targets=suse-81-i586,suse-82-i586
            --desc="Security update" mychannel *.rpm
```

will add all RPM packages in the current directory to the "mychannel" channel. The targets will be SUSE Linux 8.1 and 8.2, and the priority will be the default "suggested."

Target Platform Compatibility

Be aware that although ZENworks Linux Management will let you ship a package for many platforms, you are responsible for determining whether it will actually work on those platforms.

One particularly tricky issue is that a small number of very low-level packages actually require sub-architectures. For example, glibc-2.2.5-43-i686.rpm will not install properly on a Transmeta-based computer, even though Transmeta is an x86 architecture.

Changing Package Information

If you have already added a package, and wish to ship it for additional target platforms, you can edit its information with the web interface, or add it again at the command line, using different information in the **--targets** flag. To change target platforms for a package using the web interface:

1. In the channel administration page, click the *List packages* link, and then click the name of the package itself.
2. In the *Basic* tab, change the list of platforms as you like by selecting items from the list of available platforms on the left and adding them to the list of platforms on the right.

Deleting Packages

To delete packages from the server, enter the channel and list its packages in the *Packages* tab. Use the check boxes at the right of the package listing to select the packages you wish to delete, and then click the *Delete* button.

Alternately, use the **rcman channel-delpkg** command. The package deletion command takes the arguments, in order, of channel, package, and target. For package deletion, you must enter one target at a time. For example, to remove the Emacs package from the Developer Channel on both Red Hat 8.0 and Red Hat 9, you would do the following:

```
rcman -U name@mycorp.com channel-delpkg DevChannel emacs redhat-80-i386
rcman -U name@mycorp.com channel-delpkg DevChannel emacs redhat-9-i386
```

Virtual Packaging with Package Sets

A package set is a group of packages that are treated like a single package and have similar attributes to packages. For example, you might create a package set called "JavaTools," containing all the tools your engineering team needs to develop Java software. A package set provides functionality, just like a package does, and has the same sorts of conflicts and dependencies that a package does. It is, essentially, a convenient handle for a number of packages you always or nearly always manipulate at once.

Task Packages

> Those familiar with the Debian distribution will recognize package sets as similar to Debian's "task" metapackages. Package sets are also similar to the dummy "product component" packages created when using the [product] feature of **rcmirror**.

In many ways, package sets behave like individual packages:

- A package set must be in a channel.
- Package sets have version numbers. Any change to a package set require a change in the version number.
- Package sets can be installed or updated as though they were packages.

However, package sets are not packages, and they have a number of important differences from packages and from the channels in which they exist:

- Since package sets are created by the server administrator, it is the server administrator, rather than the software vendor, who must change the version number for a package set.
- Package set versions are always integers, although you may designate epochs as you would with a package.
- The packages in a package set may be spread across multiple channels, but this is not recommended. If a package set includes packages from two channels, clients must have access to both of them to be able to install the package set.
- Packages within a package set may not conflict with each other, although packages within a channel may.
- If a package is part of a package set it cannot be deleted unless it is removed from the package set or the package set itself is deleted.
- Package sets override package versions: If you install a package set with xsane 9.0 on a system, and xsane 9.1 becomes available in the channel, it will not be installed, even if a scheduled transaction would otherwise install it. To have the update performed, you must add a new version of the package to the package set, increase the package set's version number, and ship the new package set as an update. Therefore, be careful adding a package to a package set if you might later want to update that package alone.

Creating a New Package Set

To create a new package set using the web interface, click the *Channels* page in the navigation bar, then select the channel in which you wish to create the package set. There, click the *Add Package Set* link, and follow the instructions to provide the requested information. After each step, click *Next*, and at the end, click *Save* to complete the package set creation.

1. Choose a name and, optionally, a short summary of your package set's contents.
2. Select packages for the set. Select a channel and search for the packages you wish to add. Then check the box next to one or more packages and click *Save* to add them to the set. The packages that you add to a package set must already exist on the server.
3. Enter additional package set dependency and conflict information, if you wish. Conflict and dependency information for the package set supplements the in-

formation for the individual packages within it, so you do not necessarily need to add more information here.

You may set the following dependency information:

Obsoletes

Packages or package sets replaced by this package set. If you do not specify a relation and version number (for example, "greater than or equal to 1.0.4"), your set will render obsolete all versions of the package you enter.

Conflicts

Packages which must be removed before the package set can be installed. If you do not specify a relation and version number (for example, "greater than or equal to 1.0.4"), your set will conflict with all versions of the package you enter.

Provides

Applications or libraries which are available in the package set.

4. Select a target platform. This is identical to selecting target platforms for individual packages.
5. Set the version and release number for your package set. The first number is the major version, the second is the minor version, and the third is the release number. You may choose any integer for each value.

   In the package frobulator-1.3-5.i386.rpm, the major version is 1, the minor version is 3, and the release is 5. Major versions override minor versions, and minor versions override releases. Version numbers are not normal decimal numbers: After version 1.9, expect 1.10, pronounced "one point ten."

Epochs and Versions

You may also set an epoch, but it is suggested that you do so only rarely. To that end, epochs may be set only at the command line, with the command **rcman channel-updatepkgset** (cups). Epochs override version numbers: a package numbered 1:2.0 is newer than 0:3.0.

Once you have created the package set, it will appear in the channel like an ordinary package. If you click on it in the package list, you can update it. The update process is identical to the creation process.

Package Sets and "Requires"

Package sets do not have "requires" dependencies like normal packages do. Packages you require are simply the packages in the package set.

The command line tools for package sets may be confusing to those who are not familiar with dependency syntax. To add a dependency line to a package set, use **rcman pkgset-adddep** with arguments for type and value of the dependency. For example:

```
rcman pkgset-addep "conflicts" "j2re < 1.0"
rcman pkgset-addep "obsoletes" "netscape-common (any)"
```

Changing or Updating a Package Set

When you change a package set you must increase the version number, or client machines will not update it.

Examples of Using Package Sets

Consider package set P, packages A, B, and C, and the client machines system1 and system2. All three packages are already installed on system1, but system2 has only package A.

We start by creating package set P so that it contains packages A and B. If we ship the package set to both clients, it needs to be as a new install. System1 may already have all the required packages, but it does not yet have package set P. After the transaction, system1 will have package set P and packages A, B, and C, and system2 will have package set P and packages A and B.

Suppose there is now a new version of package A, which we'll call A-1.1. If we add A-1.1 to our distribution channel and tell the clients to update, they will not upgrade, because that would break package set P. To get them to upgrade, we need to create package set P-1.1, containing packages A-1.1 and B. When we ship that as an update, then the clients will install.

What if we add package C to the package set? To do so, we need to increase our version number again, to P-1.2. Then, we ship the package set as an update. System1 already has package C, so the only difference on that client is the package set version number. Note that System2 will also install package C, even if the transaction is not specifically an install transaction. It installs the new package because that is required by the upgrade. This is similar to the way that an upgrade from Evolution 1.4 to Evolution 2.0 will pull in junk-mail filtering software required by the new version.

In contrast, packages removed from package sets are not necessarily removed from the client machines. For example, we might change our package set again: P-1.3 contains packages A-1.1 and C, but not B. When the client machines upgrade to P-1.3, they are given no instructions about package B: it is no longer mentioned. Similarly, if you remove Evolution 2.0, the junk-mail filtering software it required is not removed. On the other hand, a new system installing P-1.3 would not install package B, again because package B is simply not mentioned.

To force the removal of package B from systems upgrading from P-1.2 to P-1.3, we would need to create an explicit conflict in the package set. Then, upgrading to P-1.3 would require the removal of package B. Remember, though, that unless removals are explicitly permitted, they will cause transaction failures, so if you want packages removed, you must not only create the conflict, but also indicate that package removals are permissible for this transaction.

Shipping a Package Set

Shipping a package set is identical to shipping a package, with one exception: channels. A package set may span multiple channels. If a client machine cannot access one of those channels, then the package set installation will fail for that client. If you spread package set contents across multiple channels, be sure that all the relevant client machines can reach all the right packages.

Chapter 5. Machines and Groups of Machines

This chapter will explain the four ways that ZENworks Linux Management interacts with client machines. They are:

Machines

Individual client machines are the objects of nearly every ZENworks Linux Management action. A client machine is assigned one or more administrators and notification recipients, and may also belong to groups and have access to channels.

Groups

Groups of individual client machines. A group is assigned one or more administrators, and may have report notifications, channel access, and transactions.

Activations

An activation is a password that grants a client a certain set of privileges. When a client uses a particular activation key (usually with the **rug activate** command) it can be assigned group membership, channel access, administrators, and notification recipients. The set of privileges assigned is called an activation profile, and is controlled by the administrator.

The Machine Set

The machine set is an ad-hoc grouping of individual machines. Transactions and permissions changes are performed through the machine set. For example, you would add a group to the machine set, then grant channel access to the machine set.

One Machine at a Time

To work with client machines, click the *Machines* button in the left hand navigation menu. There, use the search tools to find a list of machines you want to work with. If you want to see a list of all your machines, just submit an empty search. You can access any machine which has contacted the server and presented a valid activation code or an org_id token, as long as you are logged in with sufficient permissions.

The list of client machines shows one row per machine, with the machine's alias, hostname, description, and time of last contact. To edit the settings for an individual machine, click the machine name in the left-most column, just as you would for a group, channel, or administrator. This brings you to a detailed view / edit screen for that machine.

Color Coding in the Date Field

The date of last contact for each machine is color coded: a green background indicates that the machine last contacted the server some time in the past 24 hours; orange indicates contact in the last week, and red indicates that the machine has not contacted the server in at least seven days.

Once a machine is known to the ZENworks Linux Management server, you can edit it in any way you see fit, provided that you have the appropriate permissions. For example, if the machine dev1.company.com used a key to activate and join the group

"Developers," you can still add it to the "Workstations" group without sending another key. It does not matter whether it registered using an invitation, a multi-use key, or whether it contacted the server and requested activation using the organization ID token.

Click on a machine name to get more detailed information about it. There are five tabs in the detailed machine information page: *Basic*, *Permissions*, *Hardware*, *Software*, and *Settings*.

Basic Machine Information

Here, you can see the email address, hostname, alias and description associated with the machine. To save changes, click the *Save* button. You can also remove the machine from the list of available machines by clicking the *Delete* button. A deleted machine will not be able to perform any more updates, but it will reappear in the list of machines if it contacts the server again and presents a valid activation key or organization ID token.

The *Basic* tab is equivalent to the **rcman machine-update** command. For example, to set the description and alias on the machine "imap.company.com" you would enter this command:

```
rcman machine-update --desc="Mail server" --alias=imap imap.company.com
```

Machine Permissions

The *Permissions* tab allows you to adjust machine group membership, channel access, administration privileges, and notification addresses.

To add the machine to a group, select the group from the drop down list and press the *Add Group* button. To remove the machine from a group, select the checkbox next to the group and press the *Delete Groups* button. Use the same procedure to add and remove channel access for the machine. You will need to have Grant access to the group or channel, and Modify access to the machine, to change group and channel settings.

To choose an administrator for the machine, select the administrator account name from the list in the *Administrators* section of the page. Then, choose a permission setting. You will need Grant access to the machine, and Change access to the administrator account, to do this.

To set a notification, click the *Create New Machine Notifications* link and enter an email address. Then, select the type of notification: *Failure* or *All*. If you select *Failure*, the server will alert the email address you provide when a transaction fails. If you select *All*, the server will send a message at the end of every transaction, noting its success or failure.

Return Address on Notifications

To set the return address on server messages, edit the **rcserver.conf** file and change the **server_contact** address.

The *Permissions* tab is equivalent to using the commands **rcman group-addmachine** and **group-delmachine**, **machine-addemail** and **machine-delemail**, and so forth.

Finding the Right rcman Command

If you can't find the right command to use, be sure to look for commands starting with both objects in question. For example, there isn't a **machine-addgroup** command, because **group-addmachine** does the same thing.

Hardware Information for Client Machines

This page displays a hardware profile for the machine in question. You can use the hardware information to do things like ship video drivers appropriate to the video cards in each machine. This page provides the same information as the **rcman machine-listhardware** command.

Hardware information is updated whenever rcd restarts, and when you use the **rug sa** command to add a service. If you add new hardware to a system, it will be reflected at the next daemon restart. Usually, this will be the same as rebooting.

Machine Software Profiles

The *Software* tab in the machine display allows you to check the packages in place on a machine, regardless of whether they were installed by ZENworks Linux Management.

The *Software* tab provides the same information as **rcman machine-listpackages**, and both are equivalent to logging into the machine in question and running the command **rpm -qa**.

Machine Settings

The *Settings* tab lets you change the same client preferences you can set by using the **rug set** command or editing the `/etc/ximian/rcd.conf` file. the Section called *Preferences and Preference Management Tools* in Chapter 9 has more information about these options. To change one or more of the values, enter it in the space provided, then click the *Save* button or press **Return**. You can adjust machine settings with the **rcman machine-setpref** command, by logging into the client machine and using the **rug set** command, or by editing the `/etc/ximian/rcd.conf` file on the client.

Below the main settings list is the list of users authorised to run the **rug** or **red-carpet** applications on the machine. You can create new accounts by clicking *Create New User*, or edit an existing user by changing the user name and privileges. For more information about adding users to your client, check the rug(1) man page or the Section called *User Management Commands* in Chapter 9. Note that accounts created this way are only **rcd** accounts, not actual login accounts. To work with client users at the command line, log into the client machine and use the commands **rug user-add**, **rug user-delete**, **rug user-list**, and **rug user-edit**.

Machine System Information

The *System* tab displays information about transactions that the machine has conducted:

- Pending actions since: If there are overdue transactions, this is the date when they should have run. If there are no overdue transactions, this is blank.
- Number of pending actions: The number of transactions that the machine has scheduled in the future.
- Transactions in the last 24 hours: number of transactions attempted in the last 24 hours.
- Failed transactions: Total number of failed transactions.

Activations

The activation object is new in ZENworks Linux Management 2.0. It represents a permissions profile that can be conveniently applied to one or more machines. When you create an activation, you create a key that a client machine can use to activate with the server. A machine that uses the key gets all the attributes associated with the activation key. You can use activations to set:

Channel Access

Permission to install software from one or more channels.

Group Membership

Membership in one or more groups.

Administrators

An administrator given permissions to view, modify, or grant access on an activation has those privileges on all client machines activated with that key.

Notification

Email addresses of individuals who can be notified of transactions or transaction failures for the machines using this activation key.

Note that these notifications do not tell you when machines have activated, only when the activated machines conduct transactions. To check which machines have activated, visit the *Reports* page.

There are two types of activation: a single-use activation, which is tied to a specific email address, and a multi-use activation, which is not tied to any particular email address and can be used as many times as you like. The single-use key is best for relatively small groups, or for software whose distribution you need to control precisely.

To work with activations in the web interface, click the *Server* button in the left hand navigation bar. Then, click the *Activations* link.

To create a new activation using the web interface:

1. Click *Create New Reusable Activations* or *Create New Single-Use Activations*.

2. For reusable activations, you may enter a key name and description; both are optional. For single-use activations, you must enter an email address, and you may specify a key and description. If you do not specify a key, one will be generated for you. If you do not specify a description, it will remain blank.
3. Click the *Save* button to continue to the profile editing page.
4. Adjust the profile associated with the key you have just created. Click the *Save* button when you are finished.

You can return to the profile editing page page by clicking the activation key in the activation list.

You can have the ZENworks Linux Management server email single-use activation keys to the appropriate email addresses. To do so, select one or more single-use keys from the list and click the *Email Selected Activations* button. You can also distribute the keys yourself, if you prefer.

To work with activations at the command line, use the commands **rcman act-add**, **rcman act-update**, and so forth. To see the list of activation commands, enter **rcman --help |grep act**.

Once you have created the activation profile on the server, you can return to the client and activate there. If you are installing clients using a script-based system such as AutoYaST, you can add a single line to the end of your client installation script:

```
rug activate key email@address
```

By default, an activation key is a set of four hexadecimal numbers. However, you can choose any string shorter than 64 characters. Simply enter it in the provided space for your activation profile, or set it with the **--activation** flag for the **rcman act-update** command. Note that a profile can only have one activation key at any given time, so when you set the key you will stop new registrations from clients using the old key.

To alter the access attributes for an activation, click on it in the list of activations or use the rcman tools **act-addchannel**, **act-delchannel**, and so forth. See **rcman --help** for more information.

Be aware that the settings in an activation are applied only when the system activates. In other words, if you activate a machine, then change the profile of the activation, the activated machine is unchanged unless you activate it again. This is in contrast to groups: if you change the permissions for a group, then add a new machine to the group, both the old machines and the new machines will reflect the changes you made.

Also, activation profile settings are additive: if you activate a machine and give it access to channel A, then activate again with a new code that grants access to channel B, the machine will have access to both channels. You can activate a machine with multiple keys, one after another, to provide it with a varying set of permissions. To remove privileges, use the individual machine, channel, or group editing tools.

Security Tip: Use Reports

Check the reports to see whether invited clients have signed up, and whether individuals who have multiple-use keys have activated them. Especially with multiple-use activation keys, make sure that no unauthorized clients have appeared and begun updating.

Managing Groups of Client Machines

In contrast to earlier versions, ZENworks Linux Management 2.0, places a greater emphasis on individual machines and does not require that you use a group if you are updating a single machine. Groups do, however, make most operations much easier, and tend to prevent mistakes.

Group Settings in the Web Interface

To work with groups, click the *Groups* tab in the left hand navigation bar. The Groups page lists the groups that you have created, and has a link for *Create New Group*.

Click on the name of any existing group to see more detail and make changes. If you haven't created any groups yet, click the *Create New Group* link. All you need to create a group is a name. Once you've entered that information, submit the form, and you'll be taken to the view / edit details page for the new group. Editing a group uses approximately the same interface as adding one: just adjust the basic data about the group and click *Save*.

When you create a group, it is assigned an administrator (you), but it does not yet have any members or notification settings. Those you can set in the group editing page, which has four tabs: *Basic*, *Permissions*, *Members*, and *Transactions*.

In the *Basic* tab you will find only the name and description of the group.

In the *Permissions* tab you will find tables to set:

Administrators

> To delete a current administrator, check the box to the right of the administrator's name name and click the *Delete Checked* button. To add an administrator, select that administrator's name from the drop down list of available administrators and click *Add Administrator*.

Channels

> Set the channels that may be accessed by group members.

Notifications

> Add an email address to the list of people notified by particular events. Notification may be sent for all events or only for transaction failures.

In the *Members* tab, you will see a list of current group members and a box where you can add members from the list of available machines. You can add new machines or delete existing ones as you would administrators. To add a machine to a group, you must have Grant permission for the group and Modify permission for the machine.

The *Transactions* tab shows the list of transactions that are to be performed by members of the group. You can add transactions as you would machines or administrators. You can also create a new transaction by clicking *Create New Group Transaction*. This transaction will be performed by the current group, rather than on the members of the machine set. You can also add groups to your transactions in the transaction editing tool, covered in Chapter 6.

Transactions and Groups

Transactions created in the *Groups* page are applied to all the members of the group. Transactions created in the *Transactions* page are applied to the current members of the machine set. For more information on the machine set, see the Section called *Working with the Machine Set* .

Managing Groups with the Command Line Tools

To create a new group, you will use the "group-add" action in a similar manner to the channel addition tool:

```
rcman -U username group-add [Name of group] [group-alias]
   --desc=[description of group]
```

You can update basic group information using the **rcman group-update** command and the option flags **--name** and **--desc**. The name should be a single word consisting of normal characters, and the description a short phrase surrounded by quotation marks to protect its spaces from the shell.

To change group membership, use the command **rcman group-addmachine** or **group-delmachine**. For example, to add "workstation1" to the group "Engineering" and then display all machines in the group, you would enter these commands:

```
rcman group-addmachine Engineering workstation1
rcman group-listmachines Engineering
```

The machine set is not applicable at the command line, so transactions on groups are slightly different. You will use the **rcman group-addtrans** command to add groups to a transaction. For example, to have the Engineering group perform the transaction "my-update," you would run:

```
rcman group-addtrans my-update Engineering
```

Working with the Machine Set

The machine set is a special sort of grouping of machines. There is only one, and it consists of whatever machines you wish to manipulate at any given time. When you create a transaction in the *Transactions* section of the web interface, it is assumed that you will be applying it to the current machine set.

The machine set is only used in the web interface. For the command line, use the **rcman** transaction commands to create associations between machines and transactions.

To understand the difference between groups, activation profiles, and the machine set, picture a room with a variety of buckets: each bucket contains some machines, and there are other machines loose on the floor. When a new machine arrives in the room, your assistant tags it with a sticker and places it in a bucket.

Each bucket is a group, your assistant sorting the new machines is an activation profile, and the machines you’re holding are the machine set. If you pick up a bucket, then the machine set consists of all the machines in the bucket. If you pick up another bucket and a pair of loose machines from the floor, then the machine set consists of all the members of both groups, plus the two loose machines.

Before you work with the machine set, you must add machines to it. You can do that in several ways:

- Visit a group and add all machines in the group to the current set by clicking the *Add Group to Set* link in the far upper right of the page.
- Add one machine at a time from the *Machines* page, by checking the boxes in the *In Current Set* column and clicking the *Update Set* button.
- When editing an individual machine, click the *Add Machine to Set* link in the far upper right of the page.

You can remove machines from the set in the machine list page or in the machine set *Members* tab, by unchecking the boxes in the *In Current Set* column and clicking the *Update Set* button.

Once you have your desired machines in the current machine set, you can work with the set as though it were a single machine. The machine set page has four tabs:

Members

Displays the machines currently in the set. You can remove machines from the set here.

Permissions

This is identical to a single machine's *Permissions* page, except that changes made here apply to all machines in the set.

Settings

This is identical to a single machine's *Settings* page, except that changes made here apply to all machines in the set.

Compare

Displays a table of software differences between machines in the set. This is equivalent to using the **rcman machine-comparepkgs**, although you must specify the individual machines at the command line.

When you are satisfied with the membership and configuration of the current set, you can move on to creating transactions for it.

Chapter 6. Transactions: Making Things Happen

A transaction can be anything from a simple update of a single machine, to a complex scripted interaction for multiple machines. ZENworks Linux Management transactions let you perform various tasks on the client machines: simple or repeated updates, installations of new software, pre-installation downloads, script running, and rollback to earlier software installations.

Change from ZENworks Linux Management 1.4

Users of ZENworks Linux Management 1.4 will note that the "Autopull" and "Instapull" features have been folded into the transaction object. Transactions may be applied to groups or to individual machines.

Creating Transactions

You do not need to have a machine in the machine set to create a transaction. If your machine set does contain machines or groups, then transactions you create in the *Transactions* page will apply to them. You can change the list of machines and groups conducting a transaction later, in the transaction editing *Members* tab.

Groups and Transactions

You can also add groups to transactions by going to the group editing page and clicking the *Transactions* tab there.

To create a transaction with the web interface, click the *Create New Transaction* link at the top of the transaction list. Then:

1. Select a name for your transaction.
2. Choose a start time and date for the transaction. The default value is the current time and date. The time you enter is interpreted as being in the same time zone the server is set to; it is then translated into UTC and stored that way. See Appendix C for details on the way that the server handles time.
3. Some transactions may require the removal of software. If you are sure that you want to allow the transaction to remove software, click the *Remove conflicting packages* box. This is equivalent to the **--allow-removals** flag in **rcman**.
4. Select the type of transaction you will conduct:

   Once Only

   A basic transaction which happens once at the time you specify.

   Rollback

   Rollback transactions occur only once. When you select a rollback transaction, you also select the date to which you will revert all the software on the client machines.

Recurring

To have a transaction repeat, select this option and choose a repeat frequency.

Pre-Position

To have software downloaded in advance, choose this option and select a time and date for the downloads to begin. Pre-positioning allows you to time installations more exactly, without worrying about network traffic. The software will be downloaded at the time and date you specify as the pre-position time, and then installed at the transaction start time. Be sure to select a pre-position time that is before the start time.

Dry Run

Select this option to test the transaction without performing it. If you choose *Server Side Dry Run*, the server will simulate the transaction without involving the client. Client side dry runs download packages and resolve dependencies, but do not install or remove anything.

5. Click *Next*. If you selected *Rollback* for your transaction type, you are done. If you selected other transaction types, continue on to select software for the transaction.
6. Select one or more channels for your transaction. You do not need to choose a channel, if you prefer to work with individual packages. If you do not wish to select channels, click *Next*.

   To select a channel, pick one from the drop-down list of available channels, and choose either *Update* or *Install* for the action performed with packages in the channel.

   When you are done selecting channels, click *Next*.

Updates, Installs, and Blocked Transactions

"Update" means that software will only be installed if an older version is already present on the client machine. "Install" means that the software will be installed even if it is new to the machine.

Neither updates nor installs will occur if a newer version is already installed, or if the transaction would break a package set. This is not considered an error and will not trigger a failure report. See the Section called *Virtual Packaging with Package Sets* in Chapter 4 for more information about package sets.

7. Select one or more packages for your transaction. You do not need to choose a package, if you prefer to work with whole channels instead.

   To display the list of available packages, select a channel, enter one or more letters from the name of the package in the text entry box provided, and click the *Search* button. Then, click the checkboxes next to the names of the packages you wish to work with, choose *Install* or *Upgrade*, and click *Save*.

   When you are done selecting packages, click *Next*.

8. If you wish, select or upload pre-transaction and post-transaction scripts. Scripts are entirely optional. For information about ZENworks Linux Management transaction scripts, see the Section called *Transaction Scripts*. When you are done, click *Next*.
9. Select additional machines or groups for the transaction. The machines in the machine set will already be selected for you. When you are done, click *Next*.

Once the transaction is created, you can edit it by clicking its name in the main *Transactions* page. Each step of the creation process is available as a tab: *Basic*, *Channels*, *Packages*, and *Scripts*. In addition, you can use the *Members* tab to adjust the machines or groups which will conduct the transaction.

If you prefer, you can add a transaction at the command line, using the **rcman trans-add** command. Use the option flag **--starttime** to set the start time (see Appendix C for information about time and date handling). You may also use one of the three flags **--frequency**, to set the repeat frequency for the transaction; **--dry-run**, set to 1 if the transaction is just a test; and **--preposition**, set to the time that downloads should occur. Note that you may only set one of the three flags at a time. To remove one of the flags, set it to 0 with the **trans-update** command.

Rollback in the Command Line

In contrast to the web interface, rollback is not handled at transaction creation time. First create the transaction, then use the command **rcman trans-addrollback**, covered in the Section called *Rollback: Reverting to a Previous Installation* .

This newly created transaction does not specify any actions, but you can add them with additional commands such as **rcman trans-addchannel** or **rcman trans-addpkg**.

When you add a channel or a package to a transaction, you must also specify whether you wish to install (I) new software, or just update (U) the software if an earlier version is in place.

For example, to add a transaction named "security-transaction," and have it set to update the software in the channel "update-channel", use these commands:

```
rcman trans-add security-transaction
rcman trans-addchannel security-transaction update-channel U
```

Transaction Time Precision

To avoid hitting the server all at once, clients begin transactions at random times within fifteen minutes of the time you specify as the start time.

Transaction types are somewhat more complex at the command line, because they are set, for the most part, with option flags which override each other.

To set a transaction as a rollback transaction, use the command **rcman trans-addrollback**, with the target date as an argument. You cannot set the rollback date for transactions that have the frequency, pre-position, or dry-run flags set. To delete the rollback from a transaction, use **trans-delrollback**.

Other transaction types are handled with option flags during creation or with the **rcman trans-update** command:

Dry Run

> To set a transaction as a dry run, use the flag **--dry-run=1**. To remove the dry run status, set the value to 0. You cannot set recurrence, pre-positioning, or rollback for a transaction that has the dry-run value set to 1.

Recurrence Frequency

> Set the **--frequency=N** flag to make a transaction repeat on a regular basis. If unmodified, the number is assumed to be seconds, but you can use time units as described in the man page for **rcman**. To turn off recurrence, set the flag to 0. You cannot add dry-run status, pre-position times, or rollbacks for a transaction that repeats.

Pre-position

> Set a pre-positioning time. Rollback, recurring, and dry-run transactions cannot have pre-position times set. To turn off pre-positioning, set the value to 0.

For all transactions, you can use the **--starttime** flag to set the time that the transaction will start.

When entering a time value, most time formats are acceptable: you can use "12 hours" or "12:00" for the frequency, and "2004-01-23 24:00" or "January 23 2004 12:00am" for the pre-position or start times.

Scripting ZENworks Linux Management

ZENworks Linux Management command line utilities can be scripted and automated. There are numerous ways that scripts can enhance your ZENworks Linux Management usage, but the topic is far beyond the scope of this guide. If you develop a particularly useful script and would like to share it, send it to your sales or support representative and we will include it in future versions of this guide.

Advanced Transaction Work: Scripts and Rollback

You can use transactions to do more than just install or update software. This section will help you take full advantage of the ZENworks Linux Management transaction tools.

Transaction Scripts

You may attach a script to a transaction and have it executed on the client. Typically, they are shell scripts, but you may use other languages if you know they are installed on the client machine. Script creation is beyond the scope of this guide, but the **bash** man page should provide you with a good starting point.

When you add a script, you will designate it as a "pre" or "post" transaction script, having it run before or after the actual package transactions.

Scripts Run As Root

These scripts run as root. Be sure to test them carefully.

You can add a script to a transaction during the transaction creation process in the web interface, or with the command **rcman trans-addscript**. There is a maximum of one "pre" and one "post" script for each transaction.

Rollback: Reverting to a Previous Installation

Instead of an update or install transaction, you can also revert to a desired software configuration by date. To do so, click the *Create New Transaction* link on the machine set page. Then, check the box for *Rollback* and select your desired date for reversion.

Transactions must be either rollback or install/update transactions, not both. If you edit a transaction to add rollback, the update and installation actions will be canceled. Rollback does not interfere with transaction scripts, however.

There are several conditions that must be met before you can use rollback:

1. The "rollback" option must be enabled on the client machines. Use the command **rug set rollback true** to allow rollback for clients.
2. The desired rollback date must be some time *after* the client was installed on the client machine with the "rollback" option enabled.
3. The software versions to which you downgrade must still be available on the ZENworks Linux Management server.

To set up rollbacks at the command line, use the command **transaction-addrollback** to set the date you wish to go back to. If you add a rollback action to an existing transaction, it will override any installations, updates, or other rollback actions you might have set for that transaction previously. You can also use rollback from a command line client with the **rug rollback** command. See the **rug** man page or the Section called *Rollback to Previous Configurations* in Chapter 9 for details.

Chapter 7. Generating Activity Reports

ZENworks Linux Management server allows you to track software updates with web based reports and with the command line tools. There are several default reports, and you can also create your own reports.

Reports are presented as a table, in which each row represents a single client machine action: a script run, a package installed, removed, or updated. A transaction as created with **rcman** or the web interface may result in several actions on each machine it affects, and each of those actions will be represented by one row in a report. Each column of the report then represents some aspect of that single action.

To begin working with the report-generation features of the ZENworks Linux Management web interface, click the *Reports* button in the left side navigation menu. The *Reports* page contains a list of available reports. You can display the existing reports by clicking their names, and edit or delete them by clicking the *Edit* or *Delete* links beside them.

At the command line, reports may be displayed, but not created. Use the command **rcman report-list** to list the available reports, and **rcman report-get [reportname]** to display a particular report.

Displaying Basic Reports

ZENworks Linux Management provides a series of pre-built reports, which you can use as-is or edit to better suit your needs. They are:

Failures

> Displays data about transactions that may have failed. Failures most often occur when a transaction requires the removal of software, but software removal is not permitted. They can also occur when the client requires packages that are unavailable, or when the server cannot reach the client.
>
> The report displays the time of the transaction, the machine updating and the client software it used (**rug** or **red-carpet**), and any messages that the transaction may have produced.
>
> Transactions which do not occur due to package locks, package set overrides, or client unavailability are not considered failures and do not appear in these reports. To check machine contact dates, visit the *Machines* page.

Last Transaction

> Displays information about the last transaction for each machine in your organization. Displays the name and time of the transaction, the machine updating, the client software it used, and any messages that the transaction may have produced.

Per Machine

> Information about all transactions performed by all machines. For each machine the report displays the success and failure count, and status (failure or success) and time of the last transaction. You can see similar information by clicking on the *System* tab in the *Machines* information page.
>
> The per-machine report lets you search by hostname or partial hostname, so that you can, for example, see the reports for all the machines in a given subnet.

Pending Transactions

The list of transactions whose first occurrence time is still in the future. Does not include any repeated transaction occurrences. Transactions disappear from this report once their scheduled start time passes, regardless of whether they have actually occurred.

Past 24 Hours

For all transactions in the past 24 hours, displays the end time, the alias for the machine, the client software, and transaction messages.

Disk Usage

Displays information for machines that are using more than 90 percent of their local disk space.

Disk Checking Methods

Disk usage is checked with the **df** command, and checking it does not use significant system or network resources.

Sort Order

Click the column header (Time, Machine, Client, etc.) to sort the data by that column.

Creating Customized Reports

To create your own reports, click the *Create New Report* link in the Reports main page. Report creation involves three steps: naming the report, choosing columns of data to display, and then creating filter rules to select from that data.

For example, to find out which packages were updated recently by machines in the Engineering group, you would create columns for the display of machine names, package names, and transaction times, then create filter rules to limit the display to recent transactions from Engineering group machines.

Transaction Display

Transactions will appear in your reports if they have data for one or more of the columns you create and if they match your filter specifications. Transactions that do not match any of the data will not display. For example, if your report displays only information about transaction scripts, transactions with no scripts will not have any rows in the report.

Adding Columns

Each column in a report provides a specific piece of information about a *machine*, *transaction*, a transaction script, or a *transaction package*. Each of those objects has different attributes. For example, a column might show the host name attribute for machines, or the time attribute for transactions.

For machines, you may display the following data:

- Email: the email address associated with the machine.
- Hostname: the hostname for the machine.
- Description: the description of the machine, as set in the *Machines* tool.
- Alias: the machine alias.
- Added date: date that the machine first contacted the server.
- Last contact: the last time the machine contacted the server. This could be a queue or channel refresh rather than a transaction.
- Groups: The list of groups in which the machine has membership.
- Channels: List of channels available to the machine.
- Administrators: List of administrators who have access to the machine.
- Transactions: number of transactions attempted by the machine.
- Packages: The complete list of packages installed on the machine. This is normally a very long list and should be used with care.
- Has Updates: Whether there are updates available for the machine in question. If the server has a newer version of a package than the machine does, this value will be "True."
- Success Count: Number of successful transactions completed by the machine.
- Failure Count: Number of failed transactions for this machine.
- CPU: The machine CPU.
- Memory: Amount of RAM installed on the machine.
- Last Status: Whether the last transaction conducted by the machine succeeded or failed.
- Last Transaction: Date of the last transaction performed by this machine.

For transactions, you can display:

- Name: The transaction name, as set in the *Transactions* tool. Because each row represents a single package or script action on a single machine, many rows may have the same transaction name.
- Status: Whether the transaction was successful.
- End Time: Time that the machine completed the transaction and sent a success or failure message to the server.
- Target: The operating system and hardware running on the machine conducting the transaction.
- Message: A message from the server or client about the transaction. May be blank.
- Client: The client program initiating the transaction. This will be `Red Carpet Command Line Client` for **rug** transactions, `GUI Client` for GUI client updates, or `RCServer` for server-initiated transactions.
- Version: The version of the client program that initiated the transaction.

If you have assigned scripts to your transactions, you can display the following information:

- Type: Whether the script was set to run before or after package transactions.
- Exit status: The exit status provided by the script. Typically a number, with 0 indicating success.
- Stderr: The standard error, or stderr, output from the script. Contains error messages or other information, depending upon the script.
- Stdout: The standard output, or stdout, from the script. Contains action messages or other information, depending upon the script.
- Time: the time the script was run.

The *Transaction Package* refers to the software installed, upgraded, or removed during the action described in any given row. You may display the following information about the package:

- Action: Whether the package was upgraded, installed, or removed. If a package has been downgraded, it will be displayed as package removal and the subsequent installation of an earlier version.
- Package Name: The name of the package in this action.
- Cur. Package Epoch: The epoch of the package now installed on the machine.
- Cur. Package Version: The version of the package now installed on the machine.
- Cur. Package Release: The release number of the package now installed on the machine.
- Old Package Epoch: The epoch of the package that was installed on the machine prior to the transaction.
- Old Package Version: The version of the package that was installed on the machine prior to the transaction.
- Old Package Release: The release number of the package that was installed on the machine prior to the transaction.
- Time: The time and date that **rcd** finished with the package action. This is in contrast with the transaction end time, which indicates the time that the machine finished conducting all portions of the entire transaction.

Adding Filters to Custom Reports

Once you have selected the data objects you will display, you will want to filter them by content. Depending on the data you have chosen to display in your report columns, different types of filters will be available to you.

Specifically, if you have chosen to display any information about one type of object, you can filter by any attribute of that object. However, if you do not display any attributes of an object, you cannot filter by any attribute of the object. For example, if you display machine names, you can filter by group membership, even if group membership is not one of your columns.

To create filters:

1. Click the *Add New Filter Rule* button to add a new filter.
2. Select any of the values from the drop-down list.
3. Select a comparison operator: *Equal To Greater Than Less Than*, or *Contains*.

4. Enter the values you wish to search for.
5. If you create multiple filters, decide how to relate them. Select *And* to show transactions that match all your filter rules. Select *Or* to show transactions that match one or more.

Chapter 8. Server Administration: Licenses, News, and Organizations

The last section of the web interface contains a number of configuration and setup items that you will use relatively infrequently, and some of which are documented elsewhere in this manual:

Activations

> For information about activation management, read the Section called *Activations* in Chapter 5.

Cache License Management

> The cache is discussed in Chapter 12.

News Items

> The *News* tool allows you to create news for users to read with the **rug news** command or in the *News* section of the GUI client. You can use it to announce new software, or provide additional information on particular updates.
>
> To enter news, click the *News* tab in the left hand navigation bar, and then enter a headline, link, and summary for the news item. Then, select the channel to which the news entry is most relevant, and click *Save*.
>
> You can do the same thing at the command line with the **rcman news-add** command. See the **rcman** man page for details.

Server Configuration

> The server configuration page displays basic information about your server. These settings should not normally be changed except at install time, and are altered by editing the **/etc/ximian/rcserver.conf** file. In addition, there are several settings not displayed here, and available only through the configuration file. For example, the **server_contact** value, which sets the return address on notifications and other server mail, is set only in that file.
>
> Database
>
> > Type of database used in this deployment.
>
> Packages Path
>
> > The location of the package repository. The default is `/ximian/red-carpet-server`.
>
> Cache Directory
>
> > Package cache location for the template engine. This will almost never need to be changed.
>
> Language
>
> > The default language is *en_US*, US English. You may set this value to other standard UNIX language codes, provided that translations have been completed.

Expire

Amount of time, in seconds, that the server will wait before expiring client authentications. In most cases, it will not need to be changed.

Secret

This is a unique string used for authentication. In most cases, this should be left alone.

Theme

Custom branding theme, if any. Check with your sales representative if you have questions about custom branded software.

For detailed configuration information, check the man page for rcserver.conf(5).

Server License Administration

The server license administration page contains a list all the seat licenses you have available. If you have not purchased any licenses since your initial deployment, you will have only one entry in the table. Seat usage information is displayed here and when you log in to the web interface.

To purchase additional seats for your organization, contact your sales representative, who will send you a two-part license key. To use the key, click the *Add License* link at the top of the list of current licenses. Enter the words as the first segment, and the code as the second, then click *Add* or press **Return**. The new license key will validate itself against your existing server license and add new seats to your organization.

Target Platforms

The *Target Platforms* page displays the target platforms recognized by the ZENworks Linux Management server. This is the same as using the command **rcman targets**.

From the target list page you can make two changes: hide or show a target, and delete it entirely. To hide or show a target, select the *Shown* or *Hidden* button, and click *Save*. To delete a target, check the box in the *Delete* column and click *Delete*.

You can also add or edit target platforms if you wish, although it is not normally suggested. To add a target, click *Create new Targets* at the top of the target list. To edit a target, click its name.

Target creation and editing allow you to specify the following information:

Name

Name your target. Typically this should be the name, version, and platform, separated by dashes, omitting decimal points and spaces: Red Hat 8.0 for the i386 platform is represented as redhat-80-i386.

Description

Enter a short description for your target.

Version

The version number of the operating system.

Arch

The architecture abbreviation, such as i386, i586, i686, ppc, or sparc64.

Package format

The type of packaging system used in this operating system. This is always RPM. Other formats are not supported.

Detect

This is a series of XML tags that allows the server to recognize the operating system. Most Linux operating systems can be identified from a particular file in the /etc/ directory, and in that case the appropriate value will be

```
< file source="[filename]" substring="[unique string in file]"/ >
```

Other Unix-based operating systems have more complex identification procedures. For example, Solaris 8 must be identified by the output of the **uname** command, making the appropriate XML string as follows:

```
< os_name substring="SunOS"/>
<os_release substring="5.8"/>
<command source="uname -s" substring="SunOS"/>
<command source="uname -r" substring="5.8"/>
```

Support Level

Select a support level for the operating system: *internal*, *presupported*, *unsupported*, *supported*, *deprecated*, or *retired*. Only *supported* and *deprecated* platforms are displayed to clients.

Organization Management

Organization management is available only for multiple organization customers. For documentation on this feature, consult the "Superadmin" manual or contact your sales or support representative.

Chapter 9. Client Actions

This section describes how to work with the client applications rug and rcd. It describes how to start the rug and rcd programs, use proxies, perform remote management with rug, and a variety of other features for the rug and rcd tools.

This chapter does not discuss client activation, which is covered in the Section called *Client Activation* in Chapter 1, nor does it discuss the graphical tool **red-carpet**, which has its own manual. You can find the Red Carpet graphical client manual in the application itself and on the Ximian support website.

Starting the Client Programs

The Red Carpet Daemon must run on every client machine that you wish to update. It controls all package transactions, whether initiated on the client side or on the server.

Daemon Port Usage

Rcd listens on port 505 for signals from the server, and for remote access by rug or red-carpet clients. The ZENworks Linux Management server listens on port 443 for signals from rcd.

To start the Red Carpet Daemon, enter the command **/etc/init.d/rcd start**. To stop or restart the daemon, use the same syntax: **/etc/init.d/rcd restart** or **/etc/init.d/rcd stop**.

Daemon Control Commands

Other daemon control commands will also work. For example, you may use **/sbin/service rcd [stop|start|restart]**, or the rug commands **rug [stop|restart]**, if you prefer.

Simple Updates with rug and rcd

To perform basic updates with **rug**, become root and make sure that **rcd** is running. In general, you should subscribe to channels before installing packages from them, although you can override the rule with the **-u** or **--allow-unsubscribed** flag.

The following is a general overview of the **rug** commands and their more important arguments and flags. For a complete reference, including the flags accepted by all commands, use the man pages. You can also type **rug commandname --help** for syntax and usage help on individual commands. Alternate names and shortcuts are shown in parentheses.

Package Management Commands

The following commands are used to manipulate packages and the package database on your system:

info [packagename]

> Show detailed information about a package. Multiple packages may be listed, separated by spaces. The packages you request information on should be in a channel to which you are subscribed. If you want to look in all channels, use the **--allow-unsubscribed** flag.

install (in) [packagename]

> Install the specified package or packages. Multiple packages may be listed, separated by spaces. If packagename designates a package which you already have installed, but for which a newer version is available, the new version will be downloaded and installed. Accepts a number of flags, including **-y** to operate with no confirmation and **-d** to download but not install software.
>
> Packages with licenses embedded in them, and with the license flag set, will require the user to agree to the terms of license before installation.

packages (pa) [channelname]

> List the packages in a channel. If no channel is specified, all packages will be listed. If an ambiguous channel name is specified, specific channels will be suggested. You may use the full channel name or an alias. Multiple channel names may be used, separated by spaces. Packages are normally sorted by name; you can sort by channel with the **--sort-by-channel** flag. The first column of the package list indicates the status of the package as blank (uninstalled), i (installed), or v (a different version of the package is installed). If a package is locked (see the Section called *Locks and Locking*) the status column will also display an L.

remove (rm, erase) [packagename]

> Remove packages. Both rm and erase are equivalents. Like the install and update commands, remove accepts the **-y** flag to skip confirmation.

search [querystring]

> Search for the query string in the list of packages; you may use multiple strings one after another, separated by spaces. By default, search returns only those results which include all of the query strings; substrings are counted as matches. Use the **--match-words** and **--match-any** to adjust the query as you like. There are numerous other options for the search tool covered in the man page and **--help** text.

summary (sum)

> Display a short summary of available updates.

list-updates (lu) [channelname]

> List updates in subscribed channels. Adding channels as arguments will limit the list to those channels you specify. If you are not subscribed to a channel, it will never have updates available, even if you name it here; you must subscribe to list updates. You may use multiple channels as arguments, one after another, separated by spaces. The list is normally sorted alphabetically by the name of the packages, but you can display it by channel with the **--sort-by-channel** flag.

update (up) [channelname]

Download and install updates. With no arguments, this will get all updates for subscribed channels. To choose specific channels, list them as arguments, separated by spaces. For single packages, use rug install. Note that you must subscribe to channels before updates will appear to be available. To install or update software from a channel to which you are not subscribed, use rug install with the -u option. Update also accepts the **-y** flag to turn off confirmation, and the **-d** flag to download without installing. You can also set a minimum importance level for packages with the -i or **--importance=**flag. Possible minimum levels are ’minor’, ’feature’, ’suggested’, and ’urgent’.

verify

Verify system dependencies. If the system dependencies are not satisfied, rug will suggest the removal or installation of packages necessary to resolve the conflicts. This command accepts no arguments, but does accept the **-r** flag to permit removal of software without confirmation, and the **-y** flag to permit installs without confirmation.

Using Wildcards in Red Carpet

You may substitute standard shell wildcards in package names, although not in channel names.

Channels and Channel Commands

To work with channels, use these commands:

channels (ch)

List available channels. Accepts the flags **--subscribed** (**-s**), to list only channels to which you are subscribed, and **--unsubscribed**, to display only channels to which you are not subscribed.

mount [path]

Mount a directory as a channel, adding all packages within the directory to the channel. The platform for the channel is assumed to be the same as the platform on which you are running the command. See also the "unmount" command.

subscribe (sub) [channelname]

Subscribe to a channel. You may use the full channel name or the alias.

unmount [channel]

Unmount a directory that has been mounted as a channel. The opposite of the "mount" command.

unsubscribe (unsub) [channelname]

Unsubscribe from a channel.You may use the full channel name or the alias.

Dependency Analysis Commands

To handle and work with dependencies, use these tools:

info-conflicts (ic) [packagename]

List all known packages that conflict with the package you name.

info-provides (ip) [packagename]

Display the functionality or libraries that a package provides. If you wish to specify a particular channel from which to get the package, use the syntax **channel:package**. This tool is particularly useful in combination with the **solvedeps** tool below.

info-requirements (ir) [packagename]

List the software that must be present before the package can be installed.

what-conflicts (wc) [packagename]

List packages that conflict with the item you specify.

solvedeps (solve) [string]

Install or remove software to meet package, application, or library. This is a serious power-user tool, and handles not only package names and version numbers but also uses logical operators to indicate ranges of versions. The ! operator is used to indicate removal. For example, the command

```
rug  solvedeps "!rpm-python" "rpm404-python"
```

will remove the rpm-python package, and replace it with the rpm404-python package. Another example:

```
rug solvedeps "library.so.1" "application < 1.2"
```

installs whatever package provides library.so.1 and a version of "application" lower than 1.2. The preference is for the most recent version of a package that is in a channel to which you are subscribed. See the man page for additional flags and options.

what-provides (wp) [querystring]

List packages that provide the library, program, or package that you specify.

what-requires (wr) [querystring]

List packages that require the library, program, or package you specify.

Preferences and Preference Management Tools

Setting and getting preferences for rug does not require that you edit the configuration files.

get-prefs (get, prefs)

Displays preferences. Provide a token such as "cache-enabled" or "proxy" to display just that token and value, or without arguments to get a list of all available tokens and their current values, enter the command without arguments. Note that some token / value pairs are not displayed unless you are root. Check the rcd.conf man page for extensive detail on configuration file format and the token / value pairs that can be set and read. The display will be something like this, depending upon your configuration:

```
Name                          | Value                    | Description
------------------------------+--------------------------+-----------------------------------------------------------
proxy                         |                          | Proxy URL (if any)
proxy-username                |                          | Proxy username (if required)
proxy-password                |                          | Proxy password  (if required)
http-1.0                      | False                    | Use HTTP/1.0 for server communications
require-verified-certificates | False                    | Require verified SSL certificates
cache-enabled                 | True                     | Cache downloaded packages and metadata
cache-directory               | /var/cache/rug           | The directory to store cached packages and metadata
cache-cleanup-enabled         | True                     | Automatically clean up the cache
cache-max-age-in-days         | 30                       | The number of days a package may be in the cache
cache-max-size-in-mb          | 300                      | The maximum size of the cache (in mb)
require-signatures            | True                     | Require signed packages
heartbeat-interval            | 21600                    | The interval for refreshing server data (in seconds)
max-downloads                 | 5                        | Maximum number of concurrent package downloads
debug-level                   | 4                        | Level at which to log to standard error (0 to 6)
syslog-level                  | 4                        | Level at which to log to syslog (0 to 6)
```

set-prefs (set) [token] [value]

Set the preference token to the value you choose. These items are stored in the / etc / ximian / rcd.conf file. Check the rcd.conf man page for extensive detail on configuration file format and the token / value pairs that can be set and read. Note that the command "rug set proxy-url" changes the value "proxy" in the configuration file.

For information about how to use the configuration file and commands during install and deployment, check the Section called *Configuring Clients and the rcd.conf File* in Chapter 1.

System Commands

The following rug commands are less frequently used. They allow you to work directly with the daemon:

dump

Get an XML dump of system information. Usually used for debugging purposes; the output of this command is best directed to a file and compressed, like this:

```
rug dump > system.xml; gzip -9 system.xml
```

ping [servername]

Ping the rcd process on the server. If no server is specified, the current server will be pinged.

refresh

Refresh channel data for all channels.

restart

Restart the daemon. Because **rug** commands depend on having the daemon running, there is no ’start’ command. See the Section called *Starting the Client Programs* to learn about starting the client.

shutdown

Halt the rcd process.

User Management Commands

The Red Carpet Daemon allows you to grant administration access to remote individuals without actual system accounts, or to non-root users of the system. You can add and edit users in the web interface as described in the Section called *Machine Settings* in Chapter 5, or using the **rug** command.

When you create or edit users, you will have a variety of permission settings at your disposal:

Install

Whether the user may install new software.

Lock

Whether the user may set package locks.

Remove

Whether the user may remove software.

Subscribe

Whether the user may change channel subscriptions.

Trusted

Whether the user is considered trusted. A trusted user may install packages without package signatures.

Upgrade

Whether the user may upgrade software packages.

View

Whether the user may view packages. This allows the user to see which software is installed on the machine, and which software is in available channels. Note that local users are normally permitted to view installed and available packages, and that this permission is relevant only to remote users.

Superuser

Permits all **rug** commands except user management and settings, which must be done locally.

To work with users at the command line, use these commands:

user-add

Create an account with the Red Carpet Daemon so that someone can manage software on your system without logging into it directly. This command can also be used to grant administration access to non-root local users. You can use the **--password** (or **-P**) and **--user** (or **-U**) flags to specify the password and username.

user-delete username

Delete a specific user.

user-list

List the users that may update your system software. These are not the same as the users who have accounts on your system, but the ones created with the user-add command.

user-edit

Change permissions for the user. An interactive command, with the exception of password setting, which is handled by the **--password=[password]** (-P) flag.

Additional Commands

The following additional commands do not fit into other categories:

help

Get detailed help on a command. If no command is specified, displays a list of all of the available commands.

news

Display the Red Carpet news.

recurring

Display information about recurring events, including scheduled transactions and the heartbeat (automatic channel refresh). The command provides the label for the transaction, the total number of times it has occurred, and the date and time of the next and previous transactions. A typical "rug recurring" output might look like this:

```
Label                        | # | Next                     | Previous
-----------------------------+---+--------------------------+---------------------
supertransaction ’dev-trans’ | 0 | Sep 25, 12:22 (26m04s)   |
recurring-poll               | 1 | Sep 25, 12:49 (52m45s)   | Sep 25, 11:49 (7m15s)
heartbeat                    | 0 | Sep 25, 13:48 (1h51m46s) |
```

Remote Client Management

Both **rug** and the Red Carpet graphical update tool, because they use a client/daemon structure, allow you to administer a remote system, provided that it is running **rcd** and that it allows remote connections.

To perform remote updates, you will need to create accounts with the Red Carpet Daemon, and then use rug with the -h option, which allows you to specify a host. Root privileges are required for account creation, but not for usage.

On the target system, do the following:

1. Make sure that the remote connection option is enabled:

```
rug set remote-enabled true
```

2. Create an account with a username and password:

   ```
   rug user-add username [privileges]
   ```

3. You can specify privileges for the remote user interactively.

If you already have one account on the system, you can add more remotely, with the rather long command:

```
rug -h hostname.company.com -U your-username useradd new-username
```

The -U Flag

The -U flag always refers to your username, not the username of the account you are creating.

Once you have an account on the host system, you can add the -h flag to any rug command, and perform the action on the remote system. For example, to download and install all the updates available for the system at hostname.company.com, use this command:

```
rug -U username update -y -h hostname.company.com
```

Be Careful With The Yes Flag

Be sure that you really want to perform the operation before running any command with the -y ("yes to all questions") flag. You may wish to perform a dry-run (-N) transaction first, to make sure that the actions are those that you expect.

Package Sets on the Client Side

To the client, a package set appears to be a normal package. For information about using package sets on the server side, see the Section called *Virtual Packaging with Package Sets* in Chapter 4.

A package set has three possible states on the client: installed, not installed, or partially installed. "Partially installed" indicates that the package set has been installed, but that one or more of its component packages have been removed.

To avoid a "partially installed" state, you can lock the package set on the client with the **rug lock-add** command. This is equivalent to locking every package in the package set.

The client keeps track of the package set state.

You can use rollback with package sets just as you would with packages.

Using Multiple ZENworks Linux Management Servers from One Client

As of Red Carpet 2.0 you can use multiple servers for a single client. Each server is called a service, and is manipulated with the following commands:

rug service-list

Displays a list of services currently known to the client. Other commands that refer to services or take the **--service=** (**-s**) option flag allow you to refer to these services by name, URL, or number.

rug service-add

Add a service. As an argument, provide the URL of the service. For example, **rug add service https://my-server.com/data**.

rug service-delete

Delete a service. As an argument, use the name, number, or URL of the service you wish to delete.

If you use multiple servers, you must name them individually when performing server-specific actions, such as activation. Do so with the **--service=** flag:

```
rug activate --service=1 [key] [email@address]
```

Additional rug Features

Additional features available only in the command-line tool include: extra preferences settings, locking (update exclusions) and a history / rollback feature.

Getting and Setting Preferences

The Red Carpet client applications share a set of preference properties manipulated with the **rug get** and **rug set** commands.

To set the value for a key, use **rug set** with the key you wish to set and the value as arguments. For example, if you are working entirely inside a firewall and feel that it is not necessary to verify system certificates, you would enter the command **rug set require-verified-certificates False**.

Locks and Locking

The rug client allows you to place a lock on one or more packages. Packages which are locked cannot have their installation status changed without a specific, explicit command. That is, if you have installed and locked a package, and try to perform an upgrade, installation, or removal which would otherwise remove or upgrade your locked package, your action will fail. To override the lock, you will have to specify which actions to take. For example, if libfoo is locked at version 1.0, and your planned upgrade requires 1.1, you would have to use the command **rug install libfoo-1.1** before proceeding with the planned update.

Short Names for Locking Commands

There are three locking commands, each of which has a two letter shortcut. To create a lock, use **rug lock-add** or **rug la**. To list all the locks by number, use **rug lock-list** or **rug ll**. And to delete a lock, use **rug lock-delete** or **rug ld**, specifying the number of the lock you want to delete.

Adding Locks

Locks can be added with the command **rug lock-add**. You can lock packages based on their name, version number, channel, importance, or a combination. For example, to lock the status of the Evolution package on your system, you would use the command **rug lock-add evolution**.

A more complex example uses wildcards to match patterns in package names: **rug lock-add *kernel*** will prevent any kernel updates.

You can also use mathematical relation operators to choose ranges of version numbers. For example, if you want to install any minor updates to Evolution 1.2, but do not want to upgrade to the development releases (1.3 series), you could use the command: **rug lock evolution <1.3** to lock the version number below 1.3.

Lock Overrides

If you override a lock, it will still exist for future operations. For example, if you have locked your Apache package, but then upgrade it specifically, it will still be locked and to upgrade it again you will need to repeat the override. Note also that the rollback feature ignores locks; see the Section called *Rollback to Previous Configurations* for more information.

To use channel or importance settings, use the **--channel=** or **--importance=** flags (abbreviated **-c** and **-i** respectively). Importance sets the minimum importance level at which a package will be installed. For example, to install only updates deemed ’urgent,’ but not those described as ’minor’, ’feature’, or ’suggested,’ you would enter the following:

```
rug la --importance="urgent"
```

Listing and Deleting Locks

Locks are displayed as a numbered list, which you can see with the **rug lock-list** command. To delete a lock, use **rug lock-delete** and specify the number of the lock you wish to delete.

Using the Update History

rug history packagename Search log entries for the package you specify. Accepts a variety of flags, detailed in the man page.

Rollback to Previous Configurations

The command **rug rollback** reverts software to the state it was in on the date you specify. It accepts the **-y**, **--dry-run,**, **--allow-removals**, and **--download-only** option flags. It works in the same fashion as rollback transactions initiated by the ZENworks Linux Management server.

For example, if you have version A of an application installed, and upgrade to version B, and decide you don't like it, you can use rollback to switch to version A. If, having done so, you decide you really did prefer version B, you can go back to B with the same command.

There are a few caveats, however, with this feature.

- You must turn on the "rollback" option on the client. enter the command **rug set rollback true**. See the manual pages rcd.conf(5) or rug(1) for more information.
- Rollback only works on systems using the RPM package system.
- Rollback is by date, but it only affects packages that have been changed using **rcd** clients.
- Any packages to which you downgrade must still be available on the ZENworks Linux Management server.
- The rollback option does not store a complete archive of each package you remove or upgrade, but it increases the amount of disk space used, and increases the time it takes to upgrade or remove software during normal transactions.

Rollback and Locking

Rollback automatically overrides any package locks.

Chapter 10. The rcmirror Command

To simplify the process of distributing large numbers of packages, especially entire channels, ZENworks Linux Management includes the **rcmirror** command. You can use it to move software from a server hosted outside your network ("the remote server") to your local server, so that you can redistribute the software to all your client machines.

The command creates a duplicate of full or partial channel contents from a remote server. The most obvious use is to create a duplicate of a channel on one of the servers hosted by Ximian, moving it from the Internet to your ZENworks Linux Management server. The server that is mirrored may be a ZENworks Linux Management, Red Hat Network, or YaST server.

The rcmirror usage process typically works in the following fashion:

1. System administrator creates configuration file `rcmirror.conf`, which details specifics of the remote and local servers, as well as which channels and packages will be copied.
2. System administrator sets a cron entry or other scheduled time for rcmirror to run.
3. Using the rcmirror.conf file, rcmirror determines what channel data to request from the servers.
4. **rcmirror** logs in to the remote server and downloads information about selected channels.
5. **rcmirror** logs in to the local server as a standard client, and downloads information about those same channels.
6. Channel data from the remote and local servers is compared with the package and target filtering information from `rcmirror.conf`, to determine which packages need to be downloaded.
7. Packages are downloaded from the remote server.
8. **rcmirror** logs in to the local ZENworks Linux Management server as an rcman administrator and adds the packages to the server in the appropriate channels, creating channels as necessary.

For a complete example of how to use rcmirror, see the Section called *An Example rcmirror Deployment*.

You will note that during this process, **rcmirror** must authenticate itself twice. When logging in to a ZENworks Linux Management machine, it uses the same sort of authentication as an rcd client. For rcman, it will use the email address and password of a ZENworks Linux Management administrator.

Differencess between rcmirror and rcd

The **rcmirror** command uses the same identification strings as rcd when downloading packages and package information. However, it is not actually an rcd client, and does not use any of the settings in rcd.conf.

ZENworks Linux Management comes with a Red Carpet Express activation key. We suggest that you use rcmirror as a Red Carpet Express client for access to dedicated high-bandwidth servers.

Staging Server?

You might wish to use a staging server to download software from Ximian or other distributors on the Internet; this allows for thorough testing and a more complete separation of the production environment from the Internet at large. You can do this easily using rcmirror: first mirror from the Internet to your staging server, then, from the staging server to your production server.

Flags and Arguments for rcmirror

rcmirror accepts no arguments, and draws the vast majority of its information from its configuration file, covered in the Section called *Configuration File Format for rcmirror*. The command does accept the following option flags:

-c, --conf [filename]

Use the configuration file [filename] instead of the default (`/etc/ximian/rcmirror.conf`).

-h, -?, --help

Print help information and quit.

-n, --dry-run

Display actions that would be performed with this command, but do not actually perform any actions.

-l, --list-channels

List the channels available on the remote server.

-p, --list-packages [channel]

List the packages available in the specified channel.

-t, --target [target]

Narrow the channel or package listing (-l or -p) to the specified distribution target.

--syslog

Log the rcmirror output to syslogd.

--email [address]

Send the rcmirror output to the specified email address.

Configuration File Format for rcmirror

The default configuration file for rcmirror is `/etc/ximian/rcmirror.conf`, although you can use the -c or --conf flags to specify another file if you wish. The file format is that of GNOME applications, which should also be familiar to Windows users who know INI files: sections are named in square brackets, and each section contains one or more token / value pairs, one definition to a line.

You can find example rcmirror.conf files in `/usr/share/doc/packages/rcmirror/` (for SUSE systems) or `/usr/share/doc/rcmirror/` (for Red Hat systems). The files are named rcmirror-full-example.conf, rcmirror-rhn-example.conf, rcmirror-yast-example.conf, and rcmirror-ximian-example.conf. Respectively, they demonstrate a variety of mirroring options, and focused mirroring of RHN, YaST, and Ximian servers.

Before running rcmirror, you will need to edit the default `rcmirror.conf file`, which provides several suggested templates as examples.

There are four types of section you may create:

[rcmirror]

> There can be only one `[rcmirror]` section per configuration file. It supports a single token: `cachedir`.
>
> The `cachedir` token is the path to a directory you wish to use for downloaded files. This directory must exist, and it must be writable by the user that is running rcmirror.
>
> If the `[rcmirror]` section is absent or `cachedir` is not set, **rcmirror** uses a temporary directory in `/var/tmp` for all downloaded files. That directory is completely removed on exit, regardless of the success of the rcmirror run.
>
> If `cachedir` is set, its contents will only be removed on the success of the mirroring run. Anything that is already in that directory will not be downloaded on the next mirroring run.
>
> Note that this behavior is only relevant when mirroring fails - in general, rcmirror only downloads files that are missing from your local server.

[remote]

> There can be only one `[remote]` section per configuration file. The section can have the following tokens:
>
> base
>
> > The `base` token is the path to the server you wish to mirror from.
> >
> > Mirroring from servers hosted by Ximian: Set the value to the URL of the RCX server, beginning with https://. If you are running as root, you do not need to provide the username or password tokens, as these are read directly from files in /etc/ximian.
> >
> > From the free YaST Service: Set this value to the mirror of your choice, beginning with http:// or ftp://. SUSE maintains a list of update mirrors on its website.
> >
> > From SUSE Maintenance Web (authenticated YaST updates for SUSE enterprise products): Set the value to your SUSE update server.
> >
> > From RHN: You do not need to set this value. The information is derived from the systemid file (see below).
> >
> > In some cases, usually in development or testing, base may be a path to a file, or an http URL.

proxy

If you use an HTTP proxy, set the proxy token to that server's name and port. For example: **`proxy=http://proxy.location.corp.com:3182`**. Unlike **rcd**, there is no support for proxies that require authentication. If your remote server is RHN, you will need to set your proxy using up2date or by editing the systemid file (see **man up2date** or the systemid entry below).

type

If you are mirroring from authenticated Ximian servers (Red Carpet Express or ZENworks Linux Management): You do not need to set this token.

From Red Hat Network: Set the value to rhn. If you wish to mirror anything other than the operating system running on your server, you must also set the systemid token. The proxy and base tokens are not set when using RHN.

From Red Carpet basic servers: Set the value to basic. You will also need to set the `base` token, but not `user` or `password`.

From YaST: Set the value to yast. For SUSE Maintenance Web you will need to set the `base` token as well.

systemid

This token is used only for mirroring from Red Hat Network. Its default value is `/etc/sysconfig/rhn/systemid`. If you are running ZENworks Linux Management on a Red Hat Linux operating system, and wish to mirror that same operating system from RHN, you do not need to set this token.

To mirror anything other than the operating system actually running on your local server, you must use a different systemid file, and set this value as the path to that file.

RHN will only grant access to one operating system per systemid file, so if you want to mirror another OS, you need another systemid file. In addition, there can be only one `[remote]` section per **rcmirror** configuration file, so you will need a second configuration configuration file as well.

Using two configuration files and two systemid files is only moderately complicated: the files will be nearly identical, and you can use the **--conf** flag to specify which configuration file you wish to use when executing **rcmirror**. It may help to run the mirror actions at different times, to avoid saturating your bandwidth.

To get a **systemid** file, copy it from a registered RHN client that is running the operating system you wish to mirror. The file is found at /etc/sysconfig/rhn/systemid.

The proxy setting in rcmirror does not work with RHN: you must set the proxy with up2date, or edit the systemid file, changing the value "httpProxy" to your proxy and port, and the value "enableProxy" to 1.

user and password

Mirroring from RHN: Neither user nor password are required when mirroring from RHN.

Mirroring from SUSE Maintenance Web: rcmirror can read the user and password values from the /etc/sysconfig/onlineupdate file generated by

YaST. If you have not run and registered YaST Online Update on your ZENworks Linux Management server, set the user token to your SUSE product registration code, and password to the password you set when registering the product. The public YaST service does not require authentication.

Mirroring from Ximian servers: If you are running as root, **rcmirror** can discover these values automatically. If you are not, you can find the user value in /etc/ximian/mcookie, and the password value in /etc/ximian/partnernet. These two tokens identify the machine to the remote server. If you have ever used rug or rcd on this machine, the files will already exist. However, you must make sure that the machine has permission to access the channels you wish to mirror. You must do this regardless of whether you put the user and password values into the file. To do so, do the following:

1. Your sales or support representative will give you a valid activation key.
2. Enter the command:

   ```
   rug service-add https://web-bos-ma-us.rc.ximian.com
   ```

   This will make the Red Carpet client access the Ximian Red Carpet Express server (if it is not already pointed there). If you wish to use a different remote server, substitute it for the one in the example.
3. To activate the machine, use the command:

   ```
   rug activate -s [remote-server] [key] [email@address]
   ```

   You may specify the remote server by URL, name, or number. Use **rug service-list** to determine which values to enter. If you are using only one server, you can omit the **-s [servicename]** portion entirely.
4. If you wish, use **rug service-delete** to remove the Red Carpet Express server from your **rcd** service list.

[local]

The [local] section describes the machine which is running rcmirror. It may have the following tokens:

type

Sets the type of server, which may be rce (ZENworks Linux Management) or static (plain directories of files). If you do not set this token, it is assumed to be rce.

If the value is static, **rcmirror** will produce a simple directory tree of package files. If the value is rce or the token is unset, **rcmirror** will insert the packages into the server noted in the base value.

rcmanuser and rcmanpassword

The username and password for an **rcman** administrator. The account must have control over the channel or channels being mirrored. It's a good idea to use an md5 hash instead of putting your password directly into a configuration file:

```
echo -n "password" | md5sum
```

[channel]

The `[channel]` section is the description of a single remote channel that is to be mirrored. There may be multiple channel sections per file. There is one required token, `name`. You may also use the tokens `localname`, `target`, `excludetarget`, `package`, and `excludepackage`.

`name` specifies the channel you will be mirroring from the remote server. For example, "ximian-gnome", "redhat-73-i386", and "openoffice" are all channel names.

`localname` gives the channel a different name on your server than it does on the remote server. For example, if you were mirroring a channel called "mono-software-devel-kit-redhat-80-i386" and wanted a more convenient name you could enter `localname=mono`.

`target` token limits the mirroring to the specified distribution target. This is matched with a shell-like globbing function, so you can specify **`redhat-73-*`** to get Red Hat Linux 7.3 for all supported architectures, or **`redhat-*`** for all versions of Red Hat Linux. You can specify multiple `target` tokens in one `[channel]` section. Any target matched by any of the `target` tokens will be marked as eligible for mirroring. If no `target` token is specified, all supported targets will be mirrored.

`excludetarget` excludes a target from the list to be mirrored. This is matched with a shell-like globbing function, so you can specify **`redhat-*`** to exclude all variants of Red Hat Linux. You can specify multiple "excludetarget" tags in one `[channel]` section. All target exclusions are performed after `target` tags have been processed.

`package` token limits the mirroring to a specified set of package names. Semantics are the same as with the `target` token. Globbing is allowed, and multiple `package` tags will result in all matching packages being mirrored.

`excludepackage` excludes a package from the list to be mirrored. Globbing is allowed, as are multiple `excludepackage` tags. All package exclusions are performed after `package` tags have been processed.

[product]

The [product] section allows you to mirror specific pre-arranged sets of software from Ximian. It was designed to provide an easy method of installing Ximian Desktop or Ximian Evolution on a large number of systems. Each "product" consists of a set of "components," each of which represents a set of packages. A component is roughly analogous to a package set, or one of the "task" packages from the Debian distribution.

Mirroring More than you Bargained For

rcmirror will also pull in the operating system for any given product mirrored. If you have mirrored the operating system under a different name, rcmirror will download it again. If you do not specify a target when mirroring a product, rcmirror will fetch the product, plus the entire operating system, for all available targets. To avoid downloading more than you wanted, be sure to set **`target`** or **`excludetarget`** token.

As in other rcmirror operations, the [product] section causes rcmirror to fetch software from Ximian servers and add it to the local server. If you use the [product] section, **rcmirror** will also fetch the XML files used by the Ximian installer, making it possible to use the Ximian installer with your local server.

rcmirror will also create a package set for each component of the product mirrored, containing the individual packages for the component. These package sets make it more convenient to install the large bundles of software contained in products. The naming scheme for the package sets created in this manner is "product-component-pkgset." For example, when mirroring the "ximian-evolution" product, the package set "ximian-evolution-pilot-pkgset" is created on the local server, containing a set of PalmOS synchronization packages.

The [product] section uses the **`name`** token, followed by **`target`** and **`excludetarget`** as in the [channel] section.

Mcookie, Partnernet, and Identity

The `mcookie` and `partnernet` files define the identity of a Red Carpet client machine, and do not normally change. Copying those files copies the identity of the client. You may use the values created on a different machine if you wish, although there is no good reason to do so.

An Example rcmirror Deployment

This section will demonstrate using **rcmirror** to copy software from Ximian servers to your ZENworks Linux Management server.

As with a ZENworks Linux Management deployment, the first step in using rcmirror is planning: you need to decide what software to mirror, and where to get it. After that, it’s a matter of writing a configuration file, testing, and scheduling a repeated rcmirror job with **cron**.

Planning a Mirror

As a rule, you should mirror the complete distribution for all operating systems you support, the Ximian Red Carpet tools, and additional Ximian products you plan to use, such as Ximian Desktop or Ximian Evolution. Most operating systems and all Ximian products can be mirrored directly from Ximian. However, the "Enterprise" versions of many operating systems have redistribution restrictions that mean you must download from the vendor directly.

To mirror from an OS vendor as well as from Ximian, you will need an additional configuration file and mirror action. You will need one file per source; for Red Hat Enterprise offerings you will need one file per version of the operating system.

Writing a new rcmirror.conf file

Writing a configuration file is not as difficult as it sounds, but it's a little complicated.

Before anything else, make a working copy of the rcmirror.conf file, and keep a backup of the default, just to be safe. The default file consists almost entirely of examples, so you can just go through the file and remove the comment symbols (#) from the lines you want to use. Also, check the additional example files (**rcmirror-yast-example.conf**, **rcmirror-full-example.conf**, and so forth) for more information.

Permissions and Activation

Because **rcmirror** logs in to two services, it must have two sets of access permissions. At the remote server it logs in to download packages and package information; at the local server it logs in again as an rcman administrator to add packages.

If you installed from CD, it is likely that you have not yet activated your local machine against the servers hosted at Ximian.Your sales or support representative will provide you with an activation key, and you can run the following activation commands as described in the network install instructions:

```
rug service-add https://web-bos-ma-us.rc.ximian.com
rug activate -s https://web-bos-ma-us.rc.ximian.com [key] [email@address]
```

Section one is `[rcmirror]` and the suggested contents for the section are `cachedir=/var/cache/rcmirror`. That's simple: we want a cache directory in case something fails. Make sure that the directory `/var/cache/rcmirror` exists and is writeable.

The next section is `[remote]`, controlling the connection to the remote server. For this example, we'll be mirroring from the Ximian Red Carpet Express server:

```
[remote]
base=https://web-bos-ma-us.rc.ximian.com/
```

Other Types of Remote Servers

See the Section called *Configuration File Format for rcmirror* , the rcmirror.conf man page, or the Section called *Red Hat Network and rcmirror* for information about using Red Carpet basic, YaST, or Red Hat Network servers and the `type` token.

Note that ZENworks Linux Management version 2.0 clients, including rcmirror and rcd, will not work with 1.4 servers.

The [local] section defines access to the local ZENworks Linux Management server. In this case, we only need to define the rcmanuser and rcmanpassword values. Everything else can be left at its default.

For the rcmanuser value, enter a login name for the ZENworks Linux Management server. For rcmanpassword, use your password or the output of the command **echo -n "password" | md5sum**.

Our [local] section looks like this:

```
[local]
rcmanuser=email@address.company.com
rcmanpassword=5f4dcc3b5aa765d61d8327deb882cf99
```

The last section is easy enough: defining the software we will mirror:

```
[channel]
name=redhat-80-i386
```

Mirroring In Action

The first test is checking to see if the channels are listed properly:

```
rcmirror -l
```

You should see only one channel listed: redhat-80-i386

To perform a full run of the program, type:

```
rcmirror -c /path/to/my-rcmirror.conf
```

When you are sure that it works, copy the new rcmirror.conf file into the /etc/ximian/ directory and add **rcmirror** to the server's **crontab** for nightly execution (see **man cron** or **man anacron** for more information about scheduling events).

Now, any new packages added to the Red Hat 8.0 channel will appear on your ZENworks Linux Management server overnight. If you have set up a recurring update transaction for that channel, the packages will be automatically installed on your client machines as well.

Red Hat Network and rcmirror

Due to licensing restrictions, it is difficult to redistribute Red Hat Network software. However, **rcmirror** can download packages from RHN and add them to a ZENworks Linux Management server for distribution within your organization.

Mirroring Red Hat Enterprise Linux from a Red Hat Network server differs only slightly from the previous example. This section will show you how to check your

RHN registration, add the appropriate channels to your ZENworks Linux Management server, and create an `rcmirror.conf` file for RHN.

Check Registration and Channel Availability in RHN

If you are mirroring from Red Hat Network, check your registration and channel availability with the command:

```
up2date --nox --show-channels
```

If you are registered, you will see the list of channels available, such as:

```
[root@10-0-0-233 ximian]# up2date --nox --show-channels
redhat-advanced-server-i386
redhat-stronghold-i386
bea-weblogic-jrockit-i386-7.0
```

If you are not registered, do so with the **rhn_register** command.

After registration, add channels to your RHN subscription by doing the following:

1. Go to http://rhn.redhat.com and log in.
2. Click the *Systems* tab of the web page.
3. Click *View System* and select the system (machine) you have registered.
4. Click the *Channels* link to view the channel subscriptions.
5. Select the channels you wish to mirror by clicking the check boxes next to them.
6. Then click the *Change Subscriptions* button.
7. Now, the command **up2date --nox --show-channels** will provide you with the list of channels available to mirror.

Create Channels and Groups on the ZENworks Linux Management Server

Once you have ensured that you can access the software on your remote server, you will need to prepare the local server to receive it. First, select a ZENworks Linux Management administration account that will manage the mirroring procedures. You may use an existing administrator or create your own. If you create a new one, make sure that the account has full organization privileges, rather than just channel or group privileges.

The rcmirror process will create channels for the mirrored software, so you do not need to create them beforehand. However, you may wish to create groups of clients at this time. Here's how:

1. Create a new group, controlled by your chosen administrator account. This is the group to which you will distribute the mirrored software. Using the command line, you might enter:

   ```
   rcman group-add "Red Hat Enterprise Clients" rhel-clients --desc="Machines updating RHEL"
   rcman ga "Stronghold Clients" stronghold-clients --desc="Machines with Stronghold"
   rcman ga "JRockit Group" jrockit-group --desc="Machines with JRockit"
   ```

2. Generate new activation keys and add the group or groups to them. For example, for the rhel-clients group, you might do the following:

   ```
   rcman act-add --key=rhel-activation-key
   rcman act-addgroup rhel-activation-key rhel-clients
   ```

The client machines can now activate and will be easily grouped when you wish to ship software to them.

Write the rcmirror.conf File

Finally, you are ready to create your `/etc/ximian/rcmirror.conf` file. Open a text editor and begin:

1. The [remote] section will be very simple: type="rhn" is the only token you need.
2. The [local] section will contain only the rcmanuser, and rcmanpassword tokens. The rcmanuser and rcmanpassword tokens are, of course, the username and password for a ZENworks Linux Management administrator account. We strongly suggest that you use an md5 hash of the password as described in the last example.
3. The [channel] section should include the names of the channels you wish to mirror, such as redhat-advanced-server-i386.

The resulting file will look like this:

```
[remote]
type=rhn

[local]
rcmanuser=my-rce-admin@ximian.com
rcmanpassword=md5sum-of-my-password

[channel]
name=redhat-advanced-server-i386

[channel]
name=redhat-stronghold-i386

[channel]
name=bea-weblogic-jrockit-i386-7.0
```

Test and Run

Once you have completed setup, you should run a few tests to make sure everything will go according to plan. First, list the channels your file will mirror, using the command:

```
rcmirror --list-channels --conf=[my-rcmirror-file]
```

You should see all the channels for which you have created a [channel] section in the configuration file.

If that list of channels is indeed the one you wish to mirror, use the **-n** or **--dry-run** flag, to perform a test, checking the servers and get package information, but not downloading packages or adding them to the server:

```
rcmirror --dry-run --conf=[my-rcmirror-file]
```

If you are satisfied, run without the **--dry-run** flag:

```
rcmirror --conf=[my-rcmirror-file]
```

The download may take quite awhile, especially the first time you run it, depending upon the number of packages you are downloading.

When you are ready to ship the software, create a transaction as described in Chapter 6.

Chapter 11. Variations on the Database Backend

ZENworks Linux Management 6.5 allows you to choose your database backend. The default database, PostgreSQL, is recommended for most installations. If you have an existing Oracle 9i database deployment, ZENworks Linux Management can work with that instead of using PostgreSQL.

Use the Correct Software Versions

Oracle database support is limited to the following configuration: an Oracle 9i database and either Red Hat 7.3 or Red Hat Enterprise Linux 2.1 Advanced Server (RHEL 2.1 AS). The recommended Oracle client is Oracle 9i client, Release 2.

Using an Oracle Database with ZENworks Linux Management

Preparing the Oracle Database

At a very general level, using Oracle involves loading the ZENworks Linux Management schema into your Oracle database, and then altering the configuration file **rcserver.conf** to point to your chosen database. Both steps are performed after the initial software package installation, and before running the **rce-init** script.

Before anything else, review the server schema and the installation process with your Oracle database administrator.

On the Oracle server, you will perform the following steps. Note that we have provided command-line SQL instructions, but depending on your tools, your actual database manipulation steps may be different.

1. Create a new database, or select an existing database for use with ZENworks Linux Management.
2. Create two new Oracle users. You will administer the server with the first user, and the ZENworks Linux Management server will use the second to log in to the database. The first account should be called "rcadmin" and the second should be called "redcarpet."

   ```
   CREATE USER rcadmin IDENTIFIED BY password
   CREATE USER redcarpet IDENTIFIED BY password
   ```

   If you wish to use different names, you must also alter the database schema on the ZENworks Linux Management server, which you can find in **/usr/share/rcserver/rcserver.sql.ora**.
3. Grant the users access to your chosen databases.

   ```
                 GRANT CONNECT to redcarpet;
                 GRANT CONNECT to rcadmin;
   ```
4. Grant your users the rights to modify the database.

   ```
   GRANT CREATE TABLE to redcarpet;
   GRANT CREATE TABLE to rcadmin;
   ```

```
GRANT CREATE SEQUENCE to redcarpet;
GRANT CREATE SEQUENCE to rcadmin;
GRANT CREATE PROCEDURE to redcarpet;
GRANT CREATE PROCEDURE to rcadmin;
```

5. Choose a location for the server database and size it appropriately. Then, create a TABLESPACE. The default TABLESPACE is named "rcserver." If you use a different name, be sure to change the name in the schema as well.
6. Load the schema file **/usr/share/rcserver/rcserver.sql.ora**.

Back on the ZENworks Linux Management server, perform the following steps:

1. Install the Oracle 9i client on the designated ZENworks Linux Management server system. Use the Oracle 9i client installation instructions.
2. Using the **sqlplus** utility, test for connectivity to your Oracle database. Make sure that you can log in with the user accounts you created earlier.
3. Install the additional packages required for Oracle support. To install them from the Ximian server, run the command:

   ```
   rug in zlm-oracle
   ```

   Any other required packages will be pulled in when you do so.

   If you wish to install from the CD, install the following packages using **rpm -Uvh**: php-oci8, cx_Oracle, rcserver-oracle.

   The "zlm-oracle" package provides Oracle library dependencies which are not otherwise packaged as RPMs, such as libclntsh.so.9.0 and libwtc9.so.

4. Configure the Apache server to run with the Oracle environment by creating an **/etc/sysconfig/httpd** file. Use the following example, replacing the Oracle SID "test" with your own.

   ```
   export ORACLE_BASE=/opt/oracle
   export ORACLE_HOME=$ORACLE_BASE/product/9.2.0
   export NLS_LANG=AMERICAN;
   export ORA_NLS33=$ORACLE_HOME/ocommon/nls/admin/data
   export ORACLE_SID=test
   export TNS_ADMIN=/opt/oracle/product/9.2.0/network/admin/tnsnames.ora
   export LD_LIBRARY_PATH=$ORACLE_HOME/lib:/lib:/usr/lib:/usr/local/lib
   ```

5. Create the file **/etc/sysconfig/rcq-runner**. Again, use the following example, but replace the Oracle SID variable with your own SID:

   ```
   export ORACLE_BASE=/opt/oracle
   export ORACLE_HOME=$ORACLE_BASE/product/9.2.0
   export NLS_LANG=AMERICAN;
   export ORA_NLS33=$ORACLE_HOME/ocommon/nls/admin/data
   export ORACLE_SID=test
   export TNS_ADMIN=/opt/oracle/product/9.2.0/network/admin/tnsnames.ora
   export LD_LIBRARY_PATH=$ORACLE_HOME/lib:/lib:/usr/lib:/usr/local/lib
   ```

6. Edit the file **/etc/ximian/rcserver/rcserver.conf**. Like the majority of other configuration files described in this manual, it consists of token / value pairs which you will edit:
   a. Set the token **dbname** to the SID of your Oracle database.

b. Set **dbhost** to the hostname of your Oracle database server. If you need a port number, append it with a colon, like this: **oracle.mycompany.com:8080**.

c. Set **ora_tnsname** to the TNS of your Oracle database.

d. Set **dbuser** to **redcarpet** and **dbpass** to the password you chose when you created the account on your Oracle server.

e. Set **dbproto** to **tcp** if your Oracle server is on a different machine from your ZENworks Linux Management server. Otherwise, set it to **unix**.

f. Set **dbback** to **oci8**.

Your file will look something like this:

```
[System]
smarty_compile_dir = /var/tmp/smarty-compile
dbname = test
ora_tnsname = oracle.tnsname.mycompany.com
dbuser = redcarpet
dbpass = [Your Password]
dbhost = oracle.database.mycompany.com
dbproto = tcp
dbport = 1521
dbback = oci8
ora_sid = test
packages_path = /ximian/red-carpet-server
cachedir = /var/tmp/rce-cache/
rdf_path = /var/www/html/data/red-carpet.rdf
lang = en_US
expire = 7200
secret = ssshhh
```

7. Add your new database services to your Oracle server. You may use an Oracle name server or the **tnsnames.ora** file. Your completed **tsnames.ora** file will look something like this:

```
# TNSNAMES.ORA Network Configuration File:
# /opt/oracle/product/9.2.0/network/admin/tnsnames.ora
# Generated by Oracle configuration tools.

ORACLEDB.MYCOMPANY.COM =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = updateserver.mycompany.com)(PORT= 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = oracledb.mycompany.com)
    )
  )

  UPDATESERVER.MYCOMPANY.COM =
    (DESCRIPTION =
       (ADDRESS_LIST
          = (ADDRESS = (PROTOCOL = TCP)(HOST = .ximian.com)(PORT= 1521))
       )
       (CONNECT_DATA =
           (SERVICE_NAME = oracledb.mycompany.com)
       )
    )
```

8. Run the command **/usr/sbin/rce-init** with the -U, -P, and -R options as described in the Section called *Start and Configure the Server* in Chapter 1.

Chapter 12. The Caching Server

In larger networks and those spread across multiple physical locations, network traffic and performance take on additional importance in the software management process. In those cases, it can become desirable to add a caching server to the network. The **rc-proxy** is a Squid-based caching server that stands between the client machines and your main ZENworks Linux Management server. Using a cache reduces bandwidth consumption, permits centralized management at a greater distance, and permits tighter control of a each location's firewall for added security.

The ZENworks Linux Management cache functions like the **rcmirror** command in that it fetches software from a remote repository. However, it does not store packages in a local ZENworks Linux Management server, nor does it fetch software from multiple servers. In the figure below, the ZENworks Linux Management server in the upper left corner pulls software from the Ximian software repository and distributes it to two caches inside separate LANs. The LANs save time and network resources by downloading the software only once, and the administrator saves time by managing both networks simultaneously from a single location.

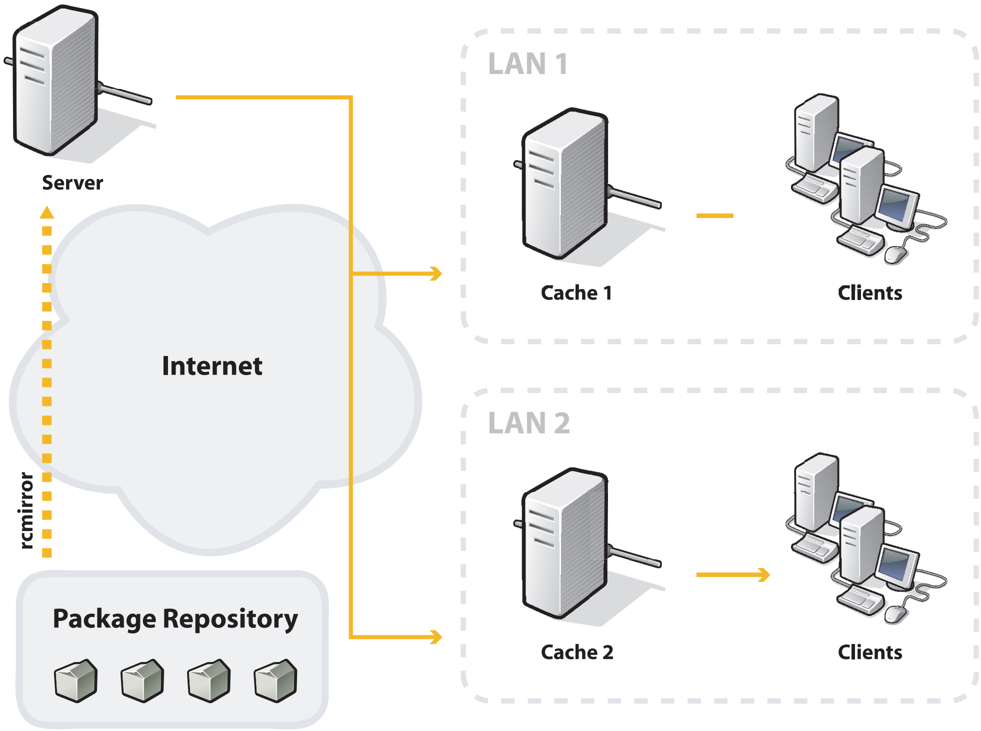


Figure 12-1. Caching

The first time a client machine makes a request from the cache server, the cache server requests and fetches it from the main ZENworks Linux Management server. Future requests are honored by the cache without having to download anything from the central server. The cache stores packages, package metadata, and in addition stores authentication for a certain amount of time, so that clients do not need to authenticate against the central server with every request.

Caching with Unreliable Networks

Using a cache with ZENworks Linux Management can allow clients to update even if their connection to the central server has failed.

Installing the Caching System

Installation of the ZENworks Linux Management cache is relatively simple:

1. Configure the computer which will act as the cache as a normal ZENworks Linux Management client.
2. Install the additional **rc-proxy** package on the cache machine.
3. On the *Server* page in the ZENworks Linux Management server web interface, click the *Cache License Administration* link. If your server key includes cache seats, you will see them listed in the Licences table.
4. If you already have cache licenses listed, skip this step (most server licenses include at least one cache license). Otherwise, click the *Add License* link and enter the two-part cache license key provided by your sales representative. Then, click the Add button.
5. Click the *Add Server* link, and enter the IP address for the cache server. Then, click the *Add* button. The ZENworks Linux Management server generates an SSL certificate, and displays a link to it next to the IP of the cache server. The file should be saved as **/etc/ximian/rcproxy/client-cert.pem** on the cache server.
6. On the cache server, run the command **rc-proxy-init**. Enter the requested information (company name, email address, and the host name and port for your primary ZENworks Linux Management server) when prompted. The cache service is now running.
7. Add the caching server as a service for the client machines instead of the central ZENworks Linux Management server. For example: **rug sa https://cachename.example.com/data**

Chapter 13. Avoiding and Resolving Problems

This chapter will help you to identify and solve problems that you encounter with ZENworks Linux Management. The main tools at your disposal are configuration and log files, data and package caches, PHP and Apache configuration files, and your support representatives.

Preventing and Preparing for Trouble

The best way to deal with problems is to prevent them, and the second best is to be prepared for them. This section will help you work with the backup, failover, and database replication features of ZENworks Linux Management, so that you can be ready for disk failure, power outage, natural disaster or other inconveniences.

Backups and Archival

To create an XML backup of the server database, use the command **rce-dump**. It accepts no arguments or flags and outputs two files ending in `.dump`. The first file contains the database information and the second file contains log information.

To restore your server from those files, use the command **rce-restore** with the main and log files as arguments.

You can use **rce-dump** on a regular basis for scheduled backups, but you should definitely use it before upgrading to a new version of ZENworks Linux Management.

The package repository, typically stored in /ximian, can be backed up and restored with standard filesystem backup tools. Be sure to run all the backup processes at the same time, so that your database matches the package repository contents.

Dependency Problems

Software dependencies can get incredibly complicated. Each individual package can have an arbitrary number of "provides," "conflicts," and "requires," and after a few packages, you end up with an intimidating list of packages, libraries, functional equivalents, and filenames.

There are several tools you can use to work with dependencies and simplify the process of shipping a coherent channel of software. The first and most obvious are those from **rug**:

dangling-requires

This command takes no arguments and no options. It simply goes to the server and checks to see if anything you ship requires something that you don't. As a general rule, it is best to minimize the number of dangling requirements.

info-conflicts (ic)

Given a package, display a list of software that conflicts with it. For example, "httpd" conflicts with "thttpd."

info-provides (ip)

List the libraries and functionality that a package provides. For example, the package rug provides:

```
/usr/bin/rug
```

```
rug = 2.0.1-0.ximian.8.0
ximian_unmarshaller.so
ximian-rug = 2.0.1-0
```

info-requirements (ir)

List a package's requirements, the package that provides them, and the channel from which the required packages come. For example, rug lists its own requirements as:

```
! | Requirement      | Provided By | Channel
--+------------------+-------------+------------------
  | /usr/bin/python  | python      | Red Hat Linux 8.0
  | python >= 1.5.2  | python      | Red Hat Linux 8.0
```

solvedeps (sd, solve)

Resolve dependencies for libraries, installing packages as necessary. This command can be used with version numbers and comparison operators. ZENworks Linux Management Server, for example, requires libgobject-2.0.so.0 and postgresql-server >= 7.2.1. To install whatever packages provide those two things, you could run the command

```
rug solvedeps libgobject-2.0.so.0 postgresql-server >= 7.2.1
```

to install glib2 and some version of postgresql-server later than 7.2.1.

what-conflicts (wc)

Display the conflicts for a library or functionality (rather than a package, as in info-conflicts).

what-provides (wp)

List packages that provide the library you specify. If, in the solvedeps example above, you wanted to know what package provided libgobject-2.0.so.0, you might use the command **rug wp libgobject-2.0.so.0** and get the response:

```
S | Channel                  | Package | Version
--+--------------------------+---------+--------
i | Red Hat Linux 8.0        | glib2   | 2.0.6-2
```

what-requires (wr)

List packages that require the library you specify (rather than a package you specify, as in info-requirements).

rug dump

The dump command outputs the current state of the **rcd** daemon. To troubleshoot someone else's daemon problems, stop your daemon and run the command **/usr/sbin/rcd --undump=[filename]**.

You may also use the various tools provided directly by **rpm**:

rpm --rebuilddb

The rpm --rebuilddb command rebuilds the database from the headers of the installed packages.

rpm --verify [packagename]

The rpm --verify command checks to see if a particular package has been installed properly. Some of the checks performed during the verification process confirm the size, MD5sum, permissions, and type of each file installed with the package. This can be useful if you suspect that files may have been damaged, or that an unauthorized update has occurred.

Specify the Channel

If you’re doing an installation and you can’t figure out why a package, usually of a lower version, is being pulled in, try explicitly choosing the package using the channel:package notation (ie, "rug in xd2-pro:acroread"). Chances are there’s a dependency problem that prevents it from being a valid solution.

Finally, it may be helpful to set a few environment variables for **rcd**:

RCD_DEBUG_DEPS

When set to true, displays additional information in the rcd logs.

RC_SPEW

Set to true to display the entire resolution process. This should not be necessary in most cases, and will produce extremely large log files.

RC_DEPS_TIME

Set to a number of seconds to provide an approximate maximum for dependency resolution. This should not normally be necessary unless you have a very large number of channels providing many possible solutions to a given request.

Using Log Files

When something goes wrong, the first place to look for a cause is in the log files:

/var/log/rcd/rcd-messages

This is a list of messages from rcd, describing actions taken with the date, time, pid of the rcd process, and the status of the action. For example, a segment might look like this:

```
Dec 23 02:22:39 [19021] Running heartbeat at Mon Dec 23 02:22:39 2003
Dec 23 02:22:39 [19021] id=468 BEGIN ’Downloading
    https://hostname.server.com/channels.php?distro_target=redhat-80-i386’
    (running)
Dec 23 02:22:40 [19021] id=468 COMPLETE ’Downloading
    https://hostname.server.com/channels.php?distro_target=redhat-80-i386’
    time=1s (finished)
Dec 23 02:22:41 [19021] Can’t subscribe to non-existent channel
    ’channelname’
```

Be sure to check this file after **rug** or **red-carpet** failures, especially if you see "ximian_xmlrpclib.py" in the traceback.

/var/log/rcd/rcd-package-history

This file describes the actions taken by rcd, one action per line, with data fields demarcated by pipe (|) characters, with an empty field represented by the underscore (_). Each line describes a single transaction. The first half of the line lists the date, a hex number identifying the transaction, whether the transaction was remote or local, the name of the user taking the action, and the action taken (upgrade, remove, or install). The last eight fields display the name, epoch, version, and build number of the package before and after the action. For example, an upgrade would look like this:

```
Fri Aug 16 19:10:10 2002
    |3d5b2|local|root|upgrade|gtk+|1|1.2|ximian.0|gtk+|1|1.2.1|ximian.3
```

/var/log/httpd/* or /var/log/apache2/*

Be sure to check both locations for web server logs, which will be placed differently depending on your operating system and the version or versions of Apache you have installed. If the ZENworks Linux Management server runs into problems, you are likely to find them described here. Check the Apache 2 documentation for more information.

The **rcman** error message `ERROR: Invalid server response` appears only when the server does not respond. That is most likely due to a PHP error, and the exact reason is likely to be found in the file **/var/log/httpd/ssl_error_log** or **/var/log/apache2/ssl_error_log**.

Configuration Files

ZENworks Linux Management uses flat files or XML files for configuration, mostly stored in /etc/ximian/ and subdirectories.

ZENworks Linux Management Server Configuration Files

The ZENworks Linux Management Server configuration files are kept in /etc/ximian/rcserver/. The most important one is rcserver.conf, which has a similar format to rcd.conf, and sets a number of variables for the system. For the most part, you should not need to change this file, or the "configured" file in the same directory.

ZENworks Linux Management Client Configuration Files

Client configuration files are kept in /etc/ximian as well. The most important is rcd.conf. The rcd.conf file format is described in its own manual page; to read it, use the **man rcd.conf** command.

The /etc/ximian/mcookie file is a system identifier, and /etc/ximian/partnernet is the key used to log in to ZENworks Linux Management servers. Except at setup time, you should not need to touch these files, and even then, they are more likely to be manipulated by the software than edited or created by hand.

If packages refuse to upgrade properly, or if you have trouble with software package locks, check the file /etc/ximian/rcd-package-locks

ZENworks Linux Management Backend Configuration Files

In the vast majority of cases, you will not need to alter the configuration files for Apache, PHP, or the SQL database. However, if you do wish to customize them, check the documentation associated with the individual products; you may also wish to speak with your support representative.

The files most directly customized by the ZENworks Linux Management server are `php.ini` and `httpd.conf`.

Machine and Server Connection Trouble

If transactions are failing inexplicably, your clients may not be connecting properly with the server. First, make sure that your ZENworks Linux Management server recognizes the machine, by seeing that it appears in the output of the **rcman machine-list** command. The output will display the name, IP address, and last contact time for each machine (use **group-listmachines** if you wish to list only the machines in a given group).

If machines do not contact the server, add services, or activate properly, check to be sure:

- That the **rcd** process on the client machine is running.
- That the client can ping the server.
- That the client is pointed to the correct server (use **rug service-list** to check).
- That the channel access and activation keys are correct.
- That the server SSL certificate is valid, or that the client is not checking SSL certificates (use **rug set require-verified-certificates false** to turn off SSL checking.)

  If, during activation or the addition of a service, the client machine displays the error message `Unable to download service info: IO error - Soup error: SSL handshake failed (8)`, it is likely that you need to turn off SSL certificate checking.

If you get the error message "Server could not contact client" during a transaction, check the following:

- In the web interface for the client, go to the settings tab. Can you set any settings on the client? This is a good quick test for a full connection that doesn't require you to run a transaction.
- Make sure that rcd on the client is not running with -r or --no-remote. If those flags are on, transactions initiated by the server will fail.
- Be sure that the server is not firewalled off from the client: can you ping the client when logged in on the server? Can you ping the server when logged in on the client?
- Check local firewall rules on the client to see if the ports you need are open. Use **rug get port** to determine the port, and compare that against your iptables or ipchains rules. Try using telnet to access the rcd port.

- Make sure DNS works from the server.
- Less likely, you may need to edit the server's **rcserver.conf** file to set **client_comm_type = ip**.

Contacting Support

Your software purchase includes support. Your sales representative will provide you with instructions for contacting the support team.

Support representatives can help you best when they have a clear understanding of your problem, so for best results please include as much as possible of the following information:

- What you were trying to do, and what actually happened.
- Whether the problem happens consistently and repeatably.
- The operating system and version you are using.

If you have trouble during a transaction, let us know whether it was initiated by the server or by the client.

For server-initiated transactions and issues involving client information conveyed to the server, be sure that rcd-modules is installed on the client.

We may ask you to attach files or portions of files to your incident:

- For client problems, attach the output file from the command **rug dump > system.xml; gzip -9 system.xml** and any relevant information from /var/log/rcd/rcd-messages.
- For database trouble when using PostgreSQL databases, attach the output of **rce-dump > db.xml; gzip -9 db.xml**. You may also want information from **/var/log/httpd/error_log** and **/var/log/httpd/access_log**.
- Attach any configuration files you have customized, particularly for **rcmirror** issues.

Glossary

Activation:

An activation, or activation profile, consists of an activation key and a set of attributes. When a client machine uses the activation key to register with the ZENworks Linux Management server, the attributes are applied to the client. The attributes include group membership, channel access, administrator assignments, and notification recipients.

Channel:

A channel is a group of software packages grouped in some logical manner. Some examples are "Ximian Desktop," "Red Hat 8.0 System Packages," and "Internal Development Tools."

FQDN:

Fully Qualified Domain Name: the complete hostname for a given machine; the name associated with a given IP address. Client machines are not identified only by FQDN.

Grant Permission:

Permissions are granted in three stages: View, Modify, and Grant. Administrators with the Grant privilege for an object have the ability to read it and alter it, and may also grant access to other administrators. An organization administrator has Grant permission for all objects in the organization.

Group:

For ZENworks Linux Management services, a group is a set of individual client systems. Actions that can be performed on a single system can be performed on an entire group as well.

Machine:

A single client computer.

Modify Permission:

Permissions are granted in three stages: View, Modify, and Grant. Administrators with the Modify privilege for an object have the ability to read it and alter it, but not to grant access to other administrators.

Package:

A package is a bundle of software in a single file. ZENworks Linux Management services ship all software in this format. Packages contain one or more files that are installed to particular locations on a system, plus information about the package, such as what applications must also be present for it to be installed. ZENworks Linux Management supports only the RPM package format.

Package Set:

A package set is a bundle of packages created on the ZENworks Linux Management server. It allows an administrator to handle a group of packages as though they were a single package.

Target:

A target, or target platform, is a combination of operating system and hardware for which software is shipped through ZENworks Linux Management services. For example, "Solaris 9 for UltraSPARC" and "Red Hat 9 for i386" are both target platforms.

Transaction:

A transaction, for the purposes of ZENworks Linux Management, refers to the installation or removal of software. Other actions, such as refreshing channel information, do not count. Transactions are normally initiated by the client.

View Permission:

Permissions are granted in three stages: View, Modify, and Grant. Administrators with the View privilege for an object have the ability to read its contents, but not to change them, nor to grant access to other administrators.

Appendix A. System Requirements for ZENworks Linux Management

This section contains information about the software and hardware you will need to deploy ZENworks Linux Management.

Client Operating Systems Supported

Both GUI and command line clients are available for the following operating systems:

- SUSE Linux Enterprise Server (SLES) version 8 and 9, SUSE Linux Desktop (SLD) version 1.0, and SUSE Linux 8.2 (x86 hardware)
- Mandrake Linux version 9.2 (x86 hardware)
- Red Hat Linux versions 7.3, 8.0, and 9 (x86 hardware)
- Red Hat Enterprise Linux (RHEL) AS, ES, and WS, version 3 (x86 hardware)
- Solaris 8 (SPARC hardware)

Command line clients, but not GUI clients, are also available for SUSE Linux Enterprise 8 IBM z64, z31, iSeries, and pSeries hardware. and on Red Hat 7.2, Red Hat Enterprise Linux 2.1 (AS, ES, and WS) and Mandrake 9.1 on x86 hardware.

The client software is largely self-contained. It requires bash, glibc, and Python. If you wish to use only the daemon, your requirements are reduced still further, to glibc and bash.

Server Operating System Requirements

ZENworks Linux Management Server 6.5 will run on SUSE Linux Enterprise Server (SLES) 8, Red Hat Linux 7.3 and 9, Red Hat Enterprise Linux Advanced Server 2.1, and Red Hat Enterprise Linux 3 Advanced Server or Enterprise Server. It requires Python, PHP, PostgreSQL, and Apache.

Appendix B. Server Installation Worksheet

You may want to fill out this worksheet and keep it handy as you install the ZENworks Linux Management server, so that you have all the necessary information in one place.

Table B-1. Server Information

Hostname	
IP Address	
Distribution and version	
Target platform	
Available disk space (df -h output)	
CD mount point	
Red Carpet Express activation key, if any	
Location of server.key file	
Destination of server.key file	
Administrator login (email)	
Administrator password	
Proxy URL, if any	
Proxy username, if any	
Proxy password, if any	

Appendix C. Server Date Handling

Because is not feasible to reliably determine the local time for each client, the physical locations of clients are not taken into account when performing transactions. All times entered are taken as local for the server, translated into UTC, and stored in the database as UTC times.

This has several results which you might not expect:

- To run a transaction at seven P.M. local time for all your client machines, you will need one transaction per time zone.
- If you move your server to a new time zone, previously created transactions will not follow it: those times are already stored as UTC times.

  Similarly, when your area switches to Daylight Saving Time (Summer Time), previously created transactions are not affected, and will appear an hour late compared to local clocks. Transactions created during Daylight Saving Time will appear to be an hour early when your time zone moves its clocks back.
- Daylight Saving Time (Summer Time) is taken into account only when you create a transaction. For example, in the United States, the time between 2:00 and 2:59 A.M., on April 3, 2005 does not exist. If you try to create a transaction with a start time of 2:30 on that day, it will be automatically set to occur at 3:30.

Because of the potential confusion surrounding time zones, you may find it most convenient to set your server's clocks to use UTC as their local time zone, and then schedule all transactions on that basis.