Parallels

Parallels Server 4 Bare Metal

Templates Management Guide

ISBN: N/A
Parallels Holdings, Ltd.
c/o Parallels Software, Inc.
13755 Sunrise Valley Drive
Suite 600
Herndon, VA 20171
USA
Tel: +1 (703) 815 5670
Fax: +1 (703) 815 5675

Contents

CHAPTER 1

Preface

In This Chapter

About Parallels Server 4 Bare Metal

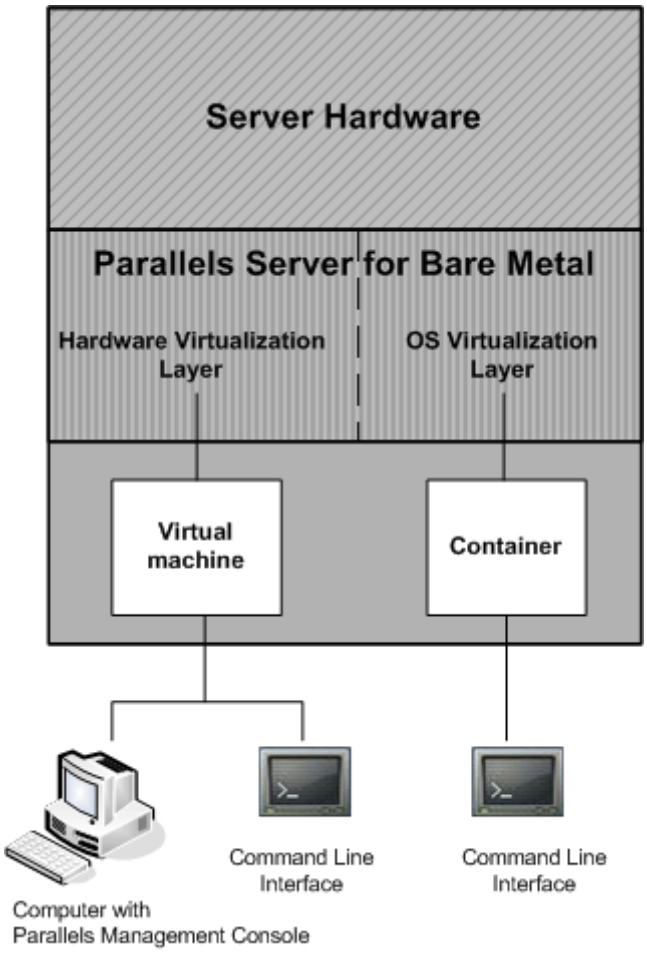
Parallels Server 4 Bare Metal provides you with the possibility to simultaneously run Parallels virtual machines and Containers on the same server. Using this software, you can efficiently use your server's hardware resources by sharing them among multiple virtual machines and Containers.

Parallels Server Bare Metal is installed directly on the server hardware and does not need any operating system for its functioning. Once it is installed, Parallels Server Bare Metal allows you to create virtual machines and Containers and manage them using the same tools you would use on systems running Parallels Server 3.0 and Parallels Virtuozzo Containers 4.0. These are the following tools:

- Command-line interface (CLI). This tool comprises a set of Parallels command-line utilities and can be used to manage virtual machines and Containers both locally and remotely.
- Parallels Management Console. Parallels Management Console is a remote management tool for Parallels Server Bare Metal with a graphical user interface. This tool can be used to manage servers and Parallels virtual machines residing on them.

  Note: In this version of Parallels Server Bare Metal, you cannot use Parallels Management Console to create and manage Parallels Containers.

Graphically, a server with the Parallels Server Bare Metal software installed can be represented as follows:



About This Guide

This guide is meant to provide complete information on Parallels templates - an exclusive Parallels technology allowing you to efficiently deploy standard Linux applications inside your Containers and to greatly save your Parallels server resources (physical memory, disk space, etc.). In particular, you will learn how to create your own application templates and manage them in a number of different ways.

The primary audience for this guide is anyone who is intended to deploy one or several applications inside their Containers and looking for ways to do it with the maximal level of efficiency. To complete all the operations described in this guide, no more than basic Linux administration habits is required.

Organization of This Guide

Chapter 2, Parallels Templates Overview, provides general information on Parallels Server Bare Metal templates: what templates are, the advantages of their usage in Parallels-based systems, etc.

Chapter 3, Managing EZ Templates, provides instructions on managing OS and application EZ templates. You will know how to create and install EZ templates on the Parallels server, add them to and remove them from Containers, make OS template caches and update them, etc.

Documentation Conventions

Before you start using this guide, it is important to understand the documentation conventions used in it.

The table below presents the existing formatting conventions.

Formatting convention	Type of Information	Example
Special Bold	Items you must select, such as menu options, command buttons, or items in a list.	Go to the Resources tab.
	Titles of chapters, sections, and subsections.	Read the Basic Administration chapter.
Italics	Used to emphasize the importance of a point, to introduce a term or to designate a command-line placeholder, which is to be replaced with a real name or value.	These are the so-called *EZ templates*. To destroy a Container, type `vzctl destroy` *`ctid`*.
`Monospace`	The names of commands, files, and directories.	Use `vzctl start` to start a Container.
`Preformatted`	On-screen computer output in your command-line sessions; source code in XML, C++, or other programming languages.	`Saved parameters for Container 101`
`Monospace Bold`	What you type, as contrasted with on-screen computer output.	**`# rpm –V virtuozzo-release`**
Key+Key	Key combinations for which the user must press and hold down one key and then press another.	Ctrl+P, Alt+F4

Besides the formatting conventions, you should also know about the document organization convention applied to Parallels documents: chapters in all guides are divided into sections, which, in their turn, are subdivided into subsections. For example, About This Guide is a section, and Documentation Conventions is a subsection.

Getting Help

In addition to this guide, there are a number of other resources available for Parallels Server Bare Metal which can help you use the product more effectively. These resources include:

Manuals:

- *Parallels Server 4 Bare Metal Installation Guide*. This guide provides detailed information on installing Parallels Server Bare Metal on your server, including the pre-requisites and the stages you shall pass.
- *Getting Started With Parallels Server 4 Bare Metal*. This guide provides basic information on how to install Parallels Server Bare Metal on your server, create new Containers and virtual machines, and perform main operations on them. As distinct from the *Parallels Server 4 Bare Metal Installation Guide*, it does not contain detailed description of all the operations needed to install and set Parallels Server Bare Metal to work (e.g. installing Parallels Server Bare Metal in the text mode).
- *Parallels Server 4 Bare Metal User's Guide*. This guide provides comprehensive information on Parallels Server Bare Metal covering the necessary theoretical conceptions as well as all practical aspects of working with the product. However, it does not deal with the process of installing and configuring your system.
- *Parallels Command Line Reference Guide*. This guide is a complete reference on all Parallels Server Bare Metal configuration files and command line utilities.
- *Deploying Clusters in Parallels-Based Systems*. This guide describes the process of creating Parallels failover and GFS clusters using the Red Hat Cluster Suite (RHCS) software.

Help systems:

- *Getting Started with Parallels Management Console*. This help system provides information on how to start working in Parallels Management Console. You will learn how to install this application on your computer, connect to a physical server running Parallels Server Bare Metal, and perform the basic operations on your virtual machines.
- *Parallels Management Console User's Guide*. This help system provides detailed information on Parallels Management Console - a graphical user interface tool for managing physical servers and their virtual machines.

Feedback

If you spot a typo in this guide, or if you have thought of a way to make this guide better, you can share your comments and suggestions with us by completing the feedback form at the Parallels documentation feedback page (http://www.parallels.com/en/support/usersdoc/).

CHAPTER 2

Templates Overview

A template in Parallels Server 4 Bare Metal is a set of application files and registry settings installed on the Parallels server in such a way as to be usable by any Container by mounting over Parallels Virtuozzo File System (Parallels VZFS). PSBM provides tools for creating templates, installing and removing them on/from the Parallels server, adding them to a Container, etc. Using templates lets you:

- Securely share the RAM among similar applications running in different Containers to save hundreds of megabytes of memory
- Securely share the files comprising a template among different Containers to save gigabytes of disk space
- Install applications and patches simultaneously in many Containers
- Use different versions of an application on different Containers (for example, perform an upgrade only in certain Containers)

There are two types of templates in Parallels Server Bare Metal. These are OS templates and application templates. An OS template is an operating system and the standard set of applications to be found right after the installation. Parallels Server Bare Metal uses OS templates to create new Containers with a pre-installed operating system. An application template is a set of repackaged software packages optionally accompanied with configuration scripts. Parallels Server Bare Metal uses application templates to add extra software to the existing Containers. For example, you can create a Container on the basis of the `redhat` OS template and add the MySQL application to it with the help of the `mysql` template.

In Parallels Server Bare Metal, you can perform the following operations on templates:

- create new application templates
- list the templates currently installed on the Parallels server
- install templates on and remove them from the Parallels server
- add templates to any number of Containers
- remove templates from the Parallels server
- remove templates from Containers
- migrate templates from one Parallels server to another

All these operations are described in the following chapters in detail.

Note: The current version of Parallels Server Bare Metal does not support using templates in virtual machines.

CHAPTER 3

Managing EZ Templates

The given chapter describes the main operations you are likely to perform on Parallels EZ templates.

In This Chapter

Understanding EZ Templates

EZ Templates Overview

EZ templates are part and parcel of the Parallels philosophy because they provide a way of sharing resources among lots of Containers, thus enabling huge savings in terms of disk space and memory. For example, when you install and cache an OS template on the Parallels server, Parallels Server Bare Metal creates the `/vz/template/<name_of_the_OS>` directory containing all the OS files that can be shared among Containers. When a Container based on this template is created, it contains only symlinks to the OS template files. These symlinks occupy very little space on the hard disk. They are situated in the so-called *private area* of the Container. The corresponding directory is `/vz/private/<CT_ID>`. The private area of a Container contains not only symlinks to the necessary template files, but also the copy-on-write area of the Container (the area for storing the information about those changes that the Container makes to the template files; this information pertains only to the given directory) and all the private Container files. When the Container is started, this private area is mounted as Virtuozzo File System (VZFS) to the `/vz/root/<CT_ID>` directory. This directory is seen as the root directory from within the Container. And, which is the pivot of it all, thanks to the VZFS, the symlinks of the Container private area are seen as real files there!

Thus, the most important directories in the `/vz` partition are the following:

`/vz/template` - contains OS and application files shared among Containers

`/vz/private` - contains VZFS symlinks to template files

`/vz/root` - contains Container mounted symlinks and all other Container files

The relation of these directories may be represented as below:

`/vz/template` (real files) → `/vz/private` (symlinks) → `/vz/root` (symlinks seen as real files in `/` for the Container)

While you are able to perform all kinds of tasks within a Container including building RPM packages and installing them, Parallels Server Bare Metal provides an easy and far more efficient way of installing the applications you need inside Containers. In the same way as you install an OS template on the Parallels Server Bare Metal system to create any number of Containers on its basis and share its resources, you can install application templates in Parallels Server Bare Metal to share application files among any number of Containers. You can then add these applications to any number of Containers with a single command.

EZ Templates Basics

All OS and application EZ templates are defined by the following features:

- EZ templates do not carry the necessary package files inside themselves. They contain only the information about what packages should be installed on the Parallels server to make the templates fully operational and from what network repository these packages should be downloaded.

  Note: For the sake of brevity, we will be saying throughout this guide that packages are included in EZ templates, which actually means that EZ templates contain the information on the corresponding packages without carrying the packages themselves.

- The dependencies of software packages included in an EZ template are automatically resolved during the packages installation on the Parallels server. So, if the specified packages require other packages to be installed, these packages are also downloaded from the repository and installed on the server. In case a package has requirements that conflict with existing software on the server or any dependencies for the package being installed cannot be satisfied, the package installation process fails without making any changes to the system.
- The EZ templates technology allows you to use the original OS and application vendor's packages and to receive the updated packages from a central repository right after their release.

One of the basic concepts in the EZ template technology is the concept of 'repository' where software packages for the given EZ template are stored. A repository is a prepared directory or web site containing the packages and index files for Linux operating systems and/or any of their applications. An example of such a repository is the repository located at the http://mirrors.usc.edu/pub/linux/distributions/fedora/ web site and storing software packages for the Fedora Core releases. Using repositories gives you the following advantages:

- Software packages included in the given EZ template do not contain versions, but only names (e.g. `bash`, `crontabs`). So, you always update any package included in the EZ template to its latest version available in the repository.
- As a result of the fact that a list of packages does not provide their versions, EZ templates do not have versions either (e.g. `redhat-el5-x86`). Thus, you install any EZ template on the Parallels server only once and, after that, use the installed template to update the packages inside any Container where it is applied.
- You can create several OS EZ template sets for one and the same Linux operating system. Any OS EZ template you are provided with has the default packages set included in it and is called the base OS EZ template. However, you can make your own OS EZ template sets (the so-called non-base OS EZ template sets) which may differ from the corresponding base template:
  - in the number of packages included in these EZ template sets
  - in the number and location of repositories to be used for these EZ template sets
  - in the number and kind of scripts to be executed on different EZ template sets lifecycle stages

Non-base OS EZ template sets must have their own names and are created by appending a random identifier to the base OS EZ template name. For example, if you wish your Container to run Red Hat Enterprise Linux 5 and to function as a Linux-based server only, you can create the `redhat-el5-x86-server` OS EZ template set and include only those packages in it that are needed for performing main server tasks. So, you can specify packages to be used for setting up file and print sharing and exclude all the packages for graphical interfaces (GNOME and KDE).

Parallels Server Bare Metal provides you with a `vzpkg` tool allowing you to automatically locate and obtain the correct packages for your EZ templates from one or several package repositories. The packages are downloaded from the repository and installed on the Parallels server in one of the following cases:

- when creating a cache for an OS EZ template
- when updating an existing OS EZ template cache (if there are new packages available in the repository)
- when adding an application EZ template or package to the first Container
- when updating EZ templates or software packages inside a destination Container

Note: Detailed information on how to manage software package repositories is provided in the Setting Up Repository and Proxy Servers for EZ Templates section (p. 22).

EZ Template Directory Structure

All EZ templates and the software packages included in them and installed on the Parallels server are located in the so-called template area the path to which is set as the value of the `TEMPLATE` variable in the `/etc/vz/vz.conf` file. By default, the `/vz/template` directory is used. The template area includes two main subdirectories:

- The `cache` subdirectory where the tar archive of the potential private area of a Container based on the corresponding OS EZ template is stored. The tar archive is created during the OS EZ template caching. Keep in mind that the OS EZ template should be obligatorily cached before you can start creating Containers on its basis.
- The template directory having the name of *<os_name>/<os_version>/<arch>* where:
  - *<os_name>* denotes the name of the Linux distribution for which the OS EZ template is created (e.g. `redhat`, `centos`, `fedora-core`).
  - *<os_version>* is the version of the Linux distribution specified as *<os_name>* (e.g. `7` or `8`).
  - *<arch>* denotes the microprocessor architecture where the OS EZ template is to be run (`x86`, `x86-64`, `ia64`).

  For example, after installing the 32-bit version of the Fedora 8 EZ template, the `/vz/template/fedora-core/8/x86` directory on the Parallels server is created.

In its turn, the *<arch>* directory contains the following subdirectories and areas:

- The template configuration subdirectory including:
  - The `config/os/default` directory where the appropriate configuration files for the base OS EZ template are stored.
  - The `config/os/`*<setname>* directory where the appropriate configuration files for non-base OS EZ templates, if any, are stored.
  - The `config/app/`*<app_name>*`/default` directory where the appropriate configuration files for the base application EZ template are stored. This directory is created if at least one application EZ template for the given OS EZ template is installed on the Parallels server.
  - The `config/app/`*<app_name>*`/`*<setname>* directory where the appropriate configuration files for non-base application EZ templates, if any, are stored.
- The packages area containing a number of software packages downloaded from the repository and installed on the Parallels server. The installed files can be shared among Containers, i.e. when a Container based on the given OS EZ template is created or application EZ templates are added to any Container, it contains only symlinks to the template files in the packages area. The installed package has the following structure:

```
<name>-<epoch>:<version>-<release>.<arch>
```

  where:

  - *<name>* is the package name.
  - *<epoch>* denotes the package epoch.
  - *<version>* indicates the package version.
  - *<release>* is the package release.

- <*arch*> denotes the microprocessor architecture where the package is to be used.

Examples of the installed software packages are the `zlib-1.2.3-14.fc8.i386.rpm` or `glib-1.2.10-28.fc8.i386.rpm` packages that can be found in the packages area on the Parallels server after installing and caching the Fedora 8 OS EZ template.

- One or several subdirectories containing the packages comprizing the corresponding OS EZ template. The directories have the following names:
  - `base`*N* for the base OS template
  - <*setname*>*N* for the non-base OS template with the name of <*setname*>, if any
  - <*appname*>*N* for the base application template
  - <*appname*>-<*setname*>*N* for the application template with the name of <*setname*>, if any

  N denotes the index number of the URL specified in the `repositories/mirrorlist` file (see the information on the `repositories/mirrorlist` file below).

As has been mentioned above, the configuration directory (i.e. /<*template_area*>/<*template_directory*>/`config`) contains a number of subdirectories storing 'EZ templates'-related configuration files. The contents of these subdirectories can vary depending on whether it is a base OS EZ template or a non-base one and on the EZ template type (OS or application template). The most important configuration files are listed below:

- Data files:
  - `packages`: contains a list of software packages names included in the corresponding EZ template.
  - `package_manager`: specifies the packaging system used to handle the EZ template.
  - `repositories`: a list of repositories where the packages comprising the EZ template are stored.
  - `mirrorlist`: one or several URLs to the file containing a list of repositories from where the packages comprising the EZ template are to be downloaded.
  - `distribution`: the name of the Linux distribution for which the EZ template is created. This file should be absent for application EZ templates.
  - `summary`: brief information on the EZ template.
  - `description`: detailed information on the EZ template. As distinct from the `summary` file, it can contain additional data relevant for the EZ template.
  - `environment`: a list of environment variables set in the form of *key*=*value*.
- Scripts:
  - `pre-cache`: this script is executed before installing the packages included in the EZ template on the Parallels server.
  - `post-cache`: this script is executed after installing the packages included in the EZ template on the Parallels server.
  - `pre-install`: this script is executed before adding the EZ template to or installing the package inside the Container.

  - `post-install`: this script is executed after adding the EZ template to or installing the package inside the Container.
  - `pre-upgrade`: this script is executed before updating the packages inside the Container.
  - `post-upgrade`: this script is executed updating the packages inside the Container.
  - `pre-remove`: this script is executed before removing the application EZ template/package from the Container.
  - `post-remove`: this script is executed after removing the application EZ template/package from the Container.
- Document files: one or several files with arbitrary names containing the information on the EZ template (e.g. `README`).

Note: Detailed information on the files contained in the OS template configuration directory is provided in the *Parallels Command LIne Reference Guide*.

While working with EZ template configuration files, keep in mind the following:

- The `packages` file should be specified for all EZ templates.
- The `packages`, `package_manager`, and `repositories/mirrorlist` files should be specified for all base OS EZ templates.
- The `package_manager` and `distribution` files should be specified for all base OS EZ templates and absent for non-base OS EZ templates and all application EZ templates.

The information from the `repositories/mirrorlist` files created for non-base OS and all application EZ templates is added to that in the `repositories/mirrorlist` files for the base OS EZ template.

Differences Between OS and Application EZ Templates

Actually, there are four major differences between OS EZ templates and application templates:

- OS templates are used to create new Containers, whereas application templates provide additional software for already created Containers.
- OS templates may and usually do use action scripts, whereas application templates cannot use action scripts in the current version of Parallels Server Bare Metal.
- You may define a list of compatible templates and a list of required templates for application templates.
- OS templates and their updates are cacheable, whereas application templates and their updates are not.

The last point needs further explanation. The fact is that creating a huge number of symlinks to the OS template when creating a Container (i.e. its private area) may take a very considerable amount of time. To reduce the time needed for creating a new Container, you should use the `vzpkg create cache` command allowing you to make a tarball of the potential private area of a Container based on the corresponding template. This tarball is also located in the `/vz/template/cache` directory. When a Container is being created, the tarball is simply deployed into the Container private area.

EZ Template Lifecycle

An EZ template has the following development stages:

1. Any EZ template should be first installed on the Parallels server. The `vzpkg install template` command enables you to install OS and application EZ templates on the server.
2. The EZ template should be cached by using the `vzpkg create cache` command. This step is required for OS EZ templates only. As a result of the `vzpkg create cache` execution, the necessary packages included in the OS EZ template are downloaded from the network repository, installed in the `/vz/template/<os_name>/<os_version>` directory on the Parallels server, and a gzipped tarball for the OS EZ template is created and put to the `/vz/template/cache` directory.
3. Either a new Container is created on the basis of the cached OS EZ template with the `pctl create` command or an application EZ template is added to any number of Containers by using the `vzpkg install` command. In the latter case, the necessary application files are first downloaded from the network repository and installed on the Parallels server and then the VZFS links to the installed files are created and added to the Container private area.
4. An obsolete EZ template applied to the corresponding Container can be updated by using the `vzpkg update` command.
5. Any EZ template excluding OS EZ templates can be removed from the Container with the `vzpkg remove` command.
6. An EZ template that is not used by any Container may be completely removed from the Parallels server with the `vzpkg remove template` command.

Creating an EZ Template

Parallels Server Bare Metal is shipped with a certain number of OS (`fedora-core-10-x86`, `redhat-el5-x86`, etc.) and application (`proftpd-fedora-core-10-x86`, `php-redhat-el5-x86`, etc.) EZ templates, which is usually sufficient to deploy the main Linux distributions and their applications inside your Containers. However, you may wish to create your own OS and application EZ templates and use them to base your Containers on or run different applications inside your Containers, respectively. To create an EZ template:

1. Make a special metafile that will be used as the basis for the EZ template creation.
2. Run the `vzmktmpl` utility and pass the corresponding options to it, if needed.

The following subsections describe both operations in detail.

Creating a Metafile for the EZ Template

In the first step, you should build a metafile - a special file serving as the basis for your new OS or application EZ template and used by the `vzmktmpl` utility during the template creation. A metafile is a text file having the `.metafile` extension and containing a list of parameters for your EZ template.

Let us assume that you wish to create an OS EZ template for the Ubuntu 8.10 distribution which is to be run under x86-64-bit processors. To create a metafile for the `Ubuntu` EZ template, perform the following operations:

1. Create a new metafile with an arbitrary name (e.g. `ubuntu-64.metafile`) and open it for editing. The easiest way to do it is to make a copy of the appropriate metafile sample located in the `/usr/share/vztt/samples` directory on the server and configure it to meet your demands. The `/usr/share/vztt/samples` directory contains the metafile samples of virtually all major Linux distributions. For example, you can use the provided `/usr/share/vztt/samples/ubuntu-6.06-x86_64/metafile` file as the basis for creating the `ubuntu-64.metafile` metafile:

```
# cp /usr/share/vztt/samples/ubuntu-6.06-x86_64/metafile /root/ubuntu/ubuntu-64.metafile
# vi /root/ubuntu/ubuntu-64.metafile
```

   Note: When creating an EZ template metafile, pay close attention to its correct configuration. EZ templates made out of incorrect metafiles may cause the Containers you will create on the basis of these templates to malfunction. For example, you may have one or more unwanted services running inside your Containers (such as `mingetty` and `klogd`), or the 'passwordless' root user access to your Container can be enabled. Therefore, we highly recommend that you use the appropriate EZ OS template metafile samples shipped with Parallels Server Bare Metal and configure them in accordance with your demands.

2. Provide the following information in the file:

   - Specify the name of the Linux distribution for which you are creating the OS EZ template as the value of the `%osname` parameter. For example:

```
%osname
ubuntu
```

     You can specify any name you like as the value of the `%osname` parameter. This name will then be assigned to the template directory on the server where the base OS EZ template will be installed (e.g. `/vz/template/ubuntu`).

   Note: If you are creating an application EZ template, you should make sure that the value of the `%osname` parameter corresponds to the name of the main template directory on the server where the OS EZ template of the Linux distribution, under which your application EZ template is to be run, is installed. More detailed information on how EZ template directories are organized is provided in the EZ Template Directory Structure subsection (p. 14).

   - Specify the version of the Linux distribution for which you are creating the OS EZ template as the value of the `%osver` parameter:

```
%osver
8.10
```

     You can specify any name you like as the value of the `%osver` parameter. This name will be then assigned to the subdirectory on the server which will denote the version of your Linux distribution (e.g. `/vz/template/ubuntu/8.10`).

Note: If you are creating an application EZ template, you should make sure that the value of the `%osver` parameter corresponds to the name of the subdirectory located in the main template directory on the server and denoting the version of the Linux distribution specified as the value of the `%osname` parameter.

- Provide the information on the microprocessor architecture where the OS EZ template is to be run as the value of the `%osarch` parameter:

```
%osarch
x86_64
```

You can set the value of the `%osarch` parameter to one of the following:

* `x86`: this value should be specified if your EZ template is to be used on 32-bit platforms;

* `x86_64`: this value should be specified if your EZ template is to be used on x86-64-bit platforms (e.g. on servers with the AMD Opteron and Intel Pentium D processors installed);

* `ia64`: this value should be specified if your EZ template is to be used on IA-64-bit platforms (i.e. on servers with the Itanium 2 processor installed).

As our template is intended for use on x86-64-bit platforms, the value of the `%osarch` parameter should be set to `x86_64`.

- Indicate what packages are to be included in your OS EZ template. The names of the packages should correspond to the names of real packages (with or without indicating the package version: e.g. `wget` or `wget=1.9.1`) that are stored in the repository to be used for managing the Ubuntu 8.10 OS EZ template. These packages will be downloaded from the package repository and installed on the server when caching the OS EZ template. The packages for Ubuntu 8.10 should be indicated as the value of the following parameters:

a `%packages_0`: provide a list of packages to be used for creating a minimal Ubuntu `chroot` environment. These packages should correspond to those installed on a standalone server on the first stage of the Ubuntu distribution installation. The packages will be installed on the server one by one in the specified order during the OS EZ template caching. If you wish several packages to be simultaneously installed on the server, you should specify the package names on a single line and separate them by spaces.

b `%packages_1`: specify a list of 'base' packages for the Ubuntu 8.10 distribution. These packages are needed to install the packages listed as the value of the `%packages` parameter.

c `%packages`: list the packages that are not specified as the values of `%packages_0` or `%packages_1` and that you to include in the Ubuntu OS EZ template.

Notes:

1. The `%packages_0` and `%packages_1` parameters should be specified only if you are creating Debian/Ubuntu-based EZ templates, which is caused by the fact that the Debian/Ubuntu distribution installation is carried out in three stages.

2. Please make sure that you have a clear understanding of what packages are to be included in `%packages_0` and `%packages_1`; otherwise, consult the corresponding Ubuntu documentation.

- Specify the package manager to be used for handling the OS EZ template.

```
%package_manager
dpkgx64
```

Depending on the Linux distribution for which you are creating the template or under which the template will be used, you should set the following values for the `package_manager` parameter:

32-bit Linux distributions:

* `rpm44x86`: Red Hat Enterprise Linux 5, Fedora Core 4, 5, and 6 and Fedora 7 and 8

* `rpm43x86`: Red Hat Enterprise Linux 3 and 4 (with the 2.6 kernel and NPTL support), CentOS 4 and 5

* `rpm41x86`: SUSE Linux Enterprise Server 10 and SUSE Linux 10.*x* where *x* denotes the minor number of the SUSE Linux 10 release (e.g. 10.1 or 10.2)

* `rpm41s9x86`: SUSE Linux Enterprise Server 9

* `dpkg`: Debian and Ubuntu

64-bit Linux distributions for x86-64 processors:

* `rpm44x64`: Red Hat Enterprise Linux 5, Fedora Core 4, 5, and 6 and Fedora 7, 8

* `rpm43x64`: Red Hat Enterprise Linux 3 and 4 (with the 2.6 kernel and NPTL support) and CentOS 4 and 5

* `rpm41x64`: SUSE Linux Enterprise Server 10 and SUSE Linux 10.*x* where *x* denotes the minor number of the SUSE Linux 10 release (e.g. 10.1 or 10.2)

* `rpm41s9x64`: SUSE Linux Enterprise Server 9

* `dpkgx64`: Debian and Ubuntu

64-bit Linux distributions for ia64 processors:

* `rpm44i64`: Red Hat Enterprise Linux 5

* `rpm43i64`: Red Hat Enterprise Linux 3 and 4 (with the 2.6 kernel and NPTL support) and CentOS 4 and 5

* `rpm41s9i64`: SUSE Linux Enterprise Server 9

* `rpm41i64`: SUSE Linux Enterprise Server 10

* `dpkgi64`: Debian and Ubuntu

- Define the list of repositories where the packages comprising the EZ template are stored:

```
%repositories
http://archive.ubuntu.com/ubuntu breezy main restricted universe
multiverse
http://archive.ubuntu.com/ubuntu breezy-updates main restricted
universe multiverse
http://archive.ubuntu.com/ubuntu breezy-security main restricted
universe multiverse
```

All the aforementioned parameters should be necessarily set in any metafile to be used for the OS EZ template creation. You can also specify a number of supplementary parameters in your metafile (e.g. you can define the `version` and `release` parameters). Detailed information on the parameters that should be present in a metafile intended for creating an OS EZ template and all the additional parameters is provided in the `vzpkg.metafile` manual pages; examples of metafiles for EZ templates can also be found in the `/usr/share/vztt/samples` directory on the server.

3 Save the file.

Using vzmktmpl to Create the EZ Template

After you have successfully built a metafile for your OS EZ template, you can use the `vzmktmpl` utility to create the EZ template. When executed, the utility takes the parameters specified in the metafile and creates a new EZ template. Let us take the metafile prepared in the previous subsection and create the `ubuntu` EZ template on its basis. Assuming that the `ubuntu-64.metafile` file is located in the `/root/ubuntu` directory on the server, this can be done as follows:

```
# vzmktmpl /root/ubuntu/ubuntu-64.metafile
Executing(%prep): /bin/sh -e /var/tmp/rpm-tmp.65461
+ umask 022
+ cd /usr/src/redhat/BUILD
+ cd /usr/src/redhat/BUILD
+ rm -rf ubuntu-8.10-x86-ez
...
```

You can also specify a number of additional options during the `vzmktmpl` execution. For example, you can use the `--post-install` option to indicate the path to the script which is to be executed after caching the `ubuntu` OS EZ template. For detailed information on all options that can be passed to `vzmktmpl`, turn to the utility manual pages; script examples can also be found in the `/usr/share/vztt/samples` directory on the server.

Upon the command completion, the created EZ template is put to your current working directory:

```
# ls /root
ubuntu-8.10-x86_64-ez-1.0-1.noarch.rpm
...
```

As can be seen from the example above, a new EZ template with the name of `ubuntu-8.10-x86_64-ez-1.0-1.noarch.rpm` has been successfully created and placed to the `/root` directory. To start using your new OS EZ template (i.e. start creating new Containers on its basis), you should first install it on the server using the `vzpkg install template` command and then cache it with the `vzpkg create cache` command.

Note: Non-base OS EZ templates inherit a number of properties from their base OS EZ templates. So, before installing a non-base OS EZ template, make sure that the corresponding base OS EZ template is installed on the server.

Setting Up Repositories and Proxy Servers for EZ Templates

If you are going to use OS and application EZ templates inside your Containers, you should first have one or several repositories with software packages prepared for your EZ templates. Package repositories are required for the EZ templates functioning due to the fact that these templates do not carry all the necessary package files inside themselves. They contain only information about what packages are included in the corresponding EZ template and from what repository they should be downloaded. In Parallels Server Bare Metal, you can make use of the following package repositories:

- Default repositories. When you install an EZ template on the server, it is preconfigured to use official vendor's file sources (e.g. RPMs from the Fedora web site) and a number of packages provided by Parallels and needed for the correct EZ template functioning. So, you can start using the default repositories right after the EZ template installation on the server. Please note that the default repositories are provided for non-commercial versions of Linux distributions only.
- Local repositories. You can build your own local repositories, which allows you:
  - To greatly save on network bandwidth when deploying package updates to several servers in your network.
  - To organize your own package repository if public repositories provided by Linux OS and application vendors are not compatible with a `vzpkg` tool used to manage EZ templates.

  Note: The `vzpkg` tool supports all the repositories that can be used by the `yum` utility (version 2.4.0 and higher) and the `apt` utility.
- Special repositories used to store software packages for commercial Linux distributions (e.g. Red Hat Enterprise Linux 5 or Suse Linux Enterprise Server 10).

Along with setting up the aforementioned package repositories, you can also create special caching proxy servers and use them to efficiently manage your OS and application EZ templates:

- The `vzpkgproxy` utility allows you to automatically set up any computer on your network to serve as a caching proxy server for EZ templates of non-commercial Linux distributions (e.g. Fedora 9 or CentOS 5).
- The `vzrhnproxy` utility allows you to create special RHN Proxy Servers to effectively manage the software packages included in the RHEL 4 and 5 OS EZ templates.

Managing the Default Repository

When you install an OS EZ template on the server, it is preconfigured to use one or several package repositories storing Linux OS vendor's file sources. The path to the repositories for most OS EZ templates and their application EZ templates is automatically set during the EZ template installation in the `mirrorlist` or `repositories` files located in the `/vz/template/<os_name>/<os_version>/<arch>/config/os/default/` directory. Thus, you do not have to perform any additional operations to start using an installed OS EZ template (i.e. cache it and create Containers on its basis). The only requirement that your system should meet is to have an Internet connection to access the servers where the specified repositories are located.

Let us assume that you wish to use the 32-bit version of Fedora 10 to base your Containers on. To do this, you should install the `fedora-core-10-x86-tmpl-4.0.0-1.swsoft.noarch.rpm` EZ template on the server (if it was not installed during the Parallels Server Bare Metal installation):

```
# vzpkg install template \\
  fedora-core-10-x86-tmpl-4.0-1.swsoft.noarch.rpm
Preparing...                ########################################### [100%]
   1:fedora-core-10-x86-tmpl  ########################################### [100%]
```

After the OS EZ template has been successfully installed, you can see the path to the default repositories storing the Fedora OS packages in the `/vz/template/fedora-core/10/x86/config/os/default/mirrorlist` file. For example:

```
# cat /vz/template/fedora-core/8/x86/config/os/default/mirrorlist
$SW_SERVER/download/mirrors/fedora-core-10
$SW_SERVER/download/mirrors/updates-released-fc10
http://mirrors.fedoraproject.org/mirrorlist?repo=fedora-10&arch=i386
http://mirrors.fedoraproject.org/mirrorlist?repo=updates-released-
f10&arch=i386
```

This file lists the repositories set to handle the Fedora 10 OS EZ template:

- Repositories on the Parallels web server defined in the first two lines. `$SW_SERVER` denotes the string whose value is specified in the `/vz/template/conf/vztt/url.map` file. In our case this string will be the URL of the Parallels web server (http://vzdownload.swsoft.com). The repositories on the Parallels server keep a number of software packages needed for the correct `fedora-core-10-x86` EZ template operation. Along with `$SW_SERVER`, the `url.map` file contains the list of URLs for all Linux distributions supported by Parallels Server Bare Metal and having official repositories.
- Repositories on the Fedora web server defined in the third and fourth lines. These repositories store all RPM packages for the Fedora 10 release.

Notice that the priority according to which software packages are downloaded from the specified servers is determined by the repositories order in the `mirrorlist` file. So, in the example above RPM packages from the Parallels web server will be downloaded first and, after that, all the packages from the Fedora server.

Notes:

1. You can use the `vzpkg info` *`OS_template_name`* `mirrorlist repositories` command to view the mirrors and repositories set to handle the corresponding OS template.

2. Commercial Linux distributions (e.g. Red Hat Enterprise Linux and SUSE Linux Enterprise Server) do not have official repositories. So you should manually create software repositories and add the information on them to the respective files before starting to use OS templates for such distributions. Refer to the Managing Repositories for Commercial Linux Distributions subsection (p. 28) to learn how to create repositories for commercial Linux distributions.

Software packages will be downloaded and installed on the server from the repositories specified in the `mirrorlist` or `repositories` files in one of the following cases:

- When you cache the OS EZ template.
- When updating an existing OS EZ template cache (if there are new packages available in the repository).
- When you add the application EZ template or package to the Container for the first time.
- When you update the EZ templates or software packages inside your Container.

You can easily add your own repositories (e.g. storing unofficial software packages) to be used by your EZ templates. To do this, you should only create the `repositories` file in the `/vz/template/<os_name>/<os_version>/<arch>/config/os/default` directory on the server, if it is not present, and specify the path to the needed repository. For example, to add the extra repository located at http://mirrors.dotsrc.org/jpackage/ and keeping Java packages for Fedora 10, you should perform the following operations:

1. Create the `/vz/template/fedora-core/10/x86/config/os/default/repositories` file on the server, if it is not yet present.
2. Add the following string to the file:

```
http://mirrors.dotsrc.org/jpackage
```

3. Save the file.

Creating a Local Repository

You can also set up a local repository where the packages included in EZ templates will be downloaded and stored. Organizing your own local repository results in less bandwidth consumption and rapid software updates inside your Containers. You may also wish to build a local repository if OS vendors or third-party software developers do not provide repositories for their versions of Linux distributions compatible with the `vzpkg` tool; so, you have to manually find and install new applications or updates inside your Containers.

Note: The `vzpkg` tool supports all the repositories that can be used by the `yum` utility (version 2.4.0 and higher) and the `apt` utility. For detailed information on these utilities, see their man pages.

The process of setting up your local repository includes the following main stages:

- Obtaining software packages comprising the given Linux distribution. The easiest way of doing it is to copy the necessary packages from your distribution disks or the OS vendor's website.
- Creating the metadata repository from a set of the copied software packages with the `createrepo` utility. This step can be omitted if you are going to create a repository which will be a mirror of a public repository.
- Making your repository accessible for Containers users. You can let Container users access your repository in one of the following ways:
  - By using the `http` protocol. In this case the repository should represent a web site containing software packages for the EZ template.
  - By using the `ftp` protocol. In this case the repository should represent an FTP site containing software packages for the EZ template.
  - By using the `file` protocol. In this case the repository should represent a directory path (e.g. on your local server) containing software packages for the EZ template.

While the first two protocols allow you to remotely (i.e. from servers located in other networks) access the created repository, the last way can be used within your local server only.

Let us assume that you wish to build a local package repository for the Fedora 10 EZ template where the RPM packages for Fedora 10 will be downloaded and stored. The repository will be used by servers from both your local and other networks, and it will allows downloading packages through the `http` protocol. In our example, we presume the following:

- The package repository will be located inside Container 101. You can use any OS template to base the Container on.

  Note: We recommend that you always place your local repositories inside separate Containers not to compromise the server security. In particular, it is of significant importance if you are going to provide access to your repositories through the `http` and `ftp` protocols.

- Container 101 is started. It has the IP address of `123.145.145.123` and can be accessed from other networks.
- The `apache` web server is running inside Container 101 and the default document root for `apache` is `/var/www/html`, i.e. the `apache` web server stores its sites in the `/vz/root/101/var/www/html` directory on the server.

- The `apache` user and group inside Container 101 are `apache`.

To create a local repository for Fedora 10, perform the following operations:

1. Install the `fedora-core-10-x86` OS EZ template (if it is not already installed):

```
# vzpkg list
redhat-el5-x86
# vzpkg install template fedora-core-10-x86-tmpl-4.0.0-1.swsoft.noarch.rpm
Preparing...                ########################################### [100%]
   1:fedora-core-10-x86-tmpl   ########################################### [100%]
# vzpkg list
fedora-core-10-x86
redhat-el5-x86
```

2. Change to the `/vz/root/101/var/www/html` directory, and create two subdirectories within it:

   Note: You can also log in to Container 101 and perform the operations described in Steps 2-8 from inside the Container. In this case your working directory inside Container 101 must be `/var/www/html`, and you will need to install the `createrepo` package inside the Container and grant the Container access to the CD-ROM drive on the server.

   - The subdirectory where the base RPM packages for Fedora 10 will be stored:

```
# mkdir -p download/fedora-core/10/i386/os/Fedora/RPMS
```

   - The subdirectory where the updated versions of RPM packages for Fedora 10 will be stored:

```
# mkdir -p download/fedora-core/updates/10/i386
```

3. Copy all the packages comprising the Fedora 10 distribution (e.g. from your Fedora 10 distribution disks) to the `download/fedora-core/10/i386/os/Fedora/RPMS` directory on the server.

4. Get the updates for Fedora 10, and put them to the `download/fedora-core/updates/10/i386` directory on the server.

5. Install the `createrepo` package on the server:

```
# rpm -Uhv createrepo-0.4.3-1.2.el4.rf.noarch.rpm
Preparing...                ########################################### [100%]
   1:createrepo             ########################################### [100%]
```

6. Change to the `/vz/root/101/var/www/html` directory and create the following metadata repositories:

   - For the Fedora 10 base RPM packages:

```
# createrepo download/fedora-core/10/i386/os
```

   - For the updated versions of the Fedora 10 RPM packages:

```
# createrepo download/fedora-core/updates/10/i386
```

   Creating the package metadata repository may take some time depending on the speed of your processor and hard disk drive.

7. Create a directory for storing mirror site lists. In our case, we will keep them in the `/vz/root/101/var/www/html/download/mirrors` directory:

```
# mkdir -p download/mirrors
```

8. Create the mirror list files and set the path to your local repository. For example, you can do it in the following way:

   - For the Fedora 10 base RPM packages:

```
# echo 'http://123.145.145.123/download/fedora-core/10/i386/os/'\
       > download/mirrors/fedora-core-10
```

   - For the updated versions of the Fedora 10 RPM packages:

```
# echo 'http://123.145.145.123/download/fedora-core/updates/10/i386' >\
      download/mirrors/updates-released-fc10
```

The aforementioned commands create the `fedora-core-10` and `updates-released-fc10` files in the `/vz/root/101/var/www/html/download/mirrors` directory on the server and add the `http://123.145.145.123/download/fedora-core/10/i386/os/Fedora/RPMS` and `http://123.145.145.123/download/fedora-core/updates/10/i386` strings to them, respectively.

9 Open the `/vz/template/conf/vztt/url.map` file on the server for editing (e.g. by using `vi`), and change the value of the `$FC_SERVER` variable as follows:

```
$FC_SERVER    http://123.145.145.123
```

10 Grant the `apache` user and the `apache` group access to the created repositories inside Container 101 by executing the following command on the server:

```
# pctl exec 101 chown -R apache.apache /var/www/html/download
```

So, our local repository is created. From now on, the `vzpkg` tool will obtain RPM packages for the Fedora 10 EZ template and their updates from your local repositories inside Container 101. You can connect to these repositories through the `http` protocol from both remote and local servers. However, you can speed up the process of managing RPM files in your repository (e.g. update EZ templates and RPM packages) for those Containers that reside on your local server (i.e. the server where the repositories are stored). This can be done by specifying the `file` protocol to be used instead of `http` to connect to your created repositories:

1 Open the `/vz/template/fedora-core/10/x86/config/os/default/mirrorlist` file on the server and comment the strings containing `$FC_SERVER`:

```
#$FC_SERVER/download/mirrors/fedora-core-10
#$FC_SERVER/download/mirrors/updates-released-fc10
```

2 Execute the following commands to create the repository files:

- To create the `/vz/template/fedora-core/10/x86/config/os/default/repositories` file on the server and to make it point to the Fedora 10 base RPM packages from your local repository:

```
# echo 'file:///vz/root/101/var/www/html/download/fedora-core/10/\
      i386/os/Fedora/RPMS' > /vz/template/fedora-core/10/x86/\
      config/os/default/repositories
```

- To create the `/vz/template/fedora-core/10/x86/config/os/default/repositories` file on the server and to make it point to the updated versions of the Fedora 10 RPM packages from your local repository:

```
# echo file:///vz/root/101/var/www/html/download/fedora-core/\
      updates/10/i386" >> /vz/template/fedora-core/10/x86/config/os/\
      default/repositories
```

Managing Repositories for Commercial Linux Distributions

Commercial Linux distributions (e.g. Red Hat Enterprise Linux and SUS Linux Enterprise Server) do not have official repositories. So if you are going to run a commercial Linux distribution inside your Containers, you should create a special repository which will store the software packages of this distribution and enable you to update the packages inside your Containers.

In the example below, we will create the repository which will store the RPM packages included in the Red Hat Enterprise Linux 4 distribution. Besides, we will consider the situation explaining to you how to keep your repository up-to-date by getting the updated packages from the Red Hat Enterprise Linux 4 web site. In our example, we presume the following:

- The server where the repository will be located is running Red Hat Enterprise Linux 4 (RHEL 4).
- The package repository will be stored inside Container 111.
- Container 111 can be accessed from other networks.
- The `apache` web server is running inside Container 111 and the default document root for `apache` is `/var/www/html`, i.e. the `apache` web server stores its sites in the `/vz/root/111/var/www/html` directory on the Hardware Node.
- The `apache` user and group inside Container 111 are `apache`.
- The `http` protocol will be used to access the RHEL 4 packages repository.

To create a repository for RHEL 4, you should perform the following operations:

1 Install the `redhat-as4-x86` OS EZ template on the server, if it is not yet installed:

```
# vzpkg list
fedora-core-8-x86
fedora-core-5-x86
# vzpkg install template redhat-as4-x86-tmpl-4.0.0-1.swsoft.noarch.rpm
Preparing...                ########################################### [100%]
   1:redhat-as4-x86-tmpl    ########################################### [100%]
# vzpkg list
fedora-core-8-x86
fedora-core-5-x86
redhat-as4-x86
```

2 Create the Container where the repository storing the RHEL 4 packages will be located and assign an IP address and hostname to it. Let us use the `fedora-core-8-x86` OS EZ template to base your Container on. For example, to create Container 111 having the IP address of `144.134.134.144` and the hostname of `my_repo` for housing the repository, you can execute the following commands:

```
# vzpkg list
fedora-core-8-x86
fedora-core-5-x86
redhat-as4-x86
# ls /vz/template/cache
fedora-core-8-x86.tar.gz
fedora-core-5-x86.tar.gz
# pctl create 111 --ostemplate fedora-core-8-x86 \
                      --ipadd 144.134.134.144 --hostname my_repo
Creating Container private area (fedora-core-8-x86)
Container is mounted
Postcreate action done
Container is unmounted
Container private area was created
```

```
Delete port redirection
Adding port redirection to Container(1): 4643 8443
```

3 Make sure that Container 111 is running and the `httpd` service is started inside the Container:

```
# vzlist -a
   CTID      NPROC STATUS  IP_ADDR         HOSTNAME
     1         42  running 10.163.163.1    localhost
    111         -  stopped 144.134.134.144 my_repo
    ...
# pctl start 111
Starting Container ...
Container is mounted
...
Container start in progress...
# pctl exec 111 service httpd status
httpd is running...
```

Container 111 should be running to be able to perform the commands listed below.

4 Inside Container 111, create a directory where the RPM packages for Red Hat Enterprise Linux 4 will be stored:

```
# mkdir -p /vz/root/111/var/www/html/download/redhat/as4/i386/os/ \
          RedHat/RPMS
```

5 Copy the RPM packages from the RHEL 4 distribution disks to the `/vz/root/111/var/www/html/download/redhat/as4/i386/os/RedHat/RPMS` directory by executing the following command for each of the RHEL 4 CDs:

```
# cp /media/cdrom/RedHat/RPMS*.rpm /vz/root/111/var/www/html/ \
     download/redhat/as4/i386/os/RedHat/RPMS
```

6 Install the `createrepo` package on the Hardware Node:

```
# rpm -Uhv createrepo-0.4.3-1.2.el4.rf.noarch.rpm
Preparing...              ########################################### [100%]
   1:createrepo           ########################################### [100%]
```

7 Create the metadata repository for the RHEL 4 packages with the `createrepo` utility:

```
# createrepo /vz/root/111/var/www/html/download/redhat/as4/ \
                i386/os/RedHat/RPMS
```

Creating the RPM metadata repository may take some time depending on the speed of your processors and hard disk drive.

8 Create a directory for keeping mirror site lists. In our case, mirror site lists will be stored in the `/vz/root/111/var/www/html/download/mirrors` directory on the server:

```
# mkdir -p /vz/root/111/var/www/html/download/mirrors
```

9 Create the `/vz/root/111/var/www/html/download/mirrors/redhat-as4` mirror list file and make it point to the repository inside Container 111 where RPM packages for RHEL 4 are stored:

```
# echo "http://144.134.134.144/download/redhat/as4/i386/os/RedHat/ \
       RPMS/" > /vz/root/111/var/www/html/download/mirrors/redhat-as4
```

This command makes the `/vz/root/111/var/www/html/download/mirrors/redhat-as4` file on the server and adds the `http://144.134.134.144/download/redhat/as4/i386/os/RedHat/RPMS` string to the file.

So, we have just created a repository for your RHEL 4 OS EZ template. Now you can cache the `redhat-as4-x86` EZ template and start creating Containers on its basis. However, if you wish to receive package updates from the RHEL 4 web site, you should additionally perform the following steps:

1. On the server, run the `up2date` utility and register your `up2date` account with RHEL 4. Please consult the `up2date` documentation to complete this task.

   Note: If your server has another Linux OS installed on it (e.g. Fedora 8), you should create a special Container which is to run Red Hat Enterprise Linux 4 and register the `up2date` account from inside this Container.

2. Inside Container 111, create a directory where the updated versions of the RHEL 4 packages will be stored:

```
# mkdir -p /vz/root/111/var/www/html/download/redhat/updates/as4/i386
```

3. Create the `/vz/root/111/var/www/html/download/mirrors/updates-released-as4` mirror list file and make it point to the repository inside Container 111 where the updated versions of the RHEL 4 packages are stored:

```
# echo "http://144.134.134.144/download/redhat/updates/as4/i386/" > \
      /vz/root/111/var/www/html/download/mirrors/updates-released-as4
```

   This command makes the `/vz/root/111/var/www/html/download/mirrors/updates-released-as4` file on the server and adds the `http://144.134.134.144/download/redhat/updates/as4/i386` string to the file.

4. On the server, create an empty RPM database. For example:

```
# mkdir -p /var/repo/redhat-as4
# rpm --initdb --dbpath /var/repo/redhat-as4
# rpm --dbpath /var/repo/redhat-as4 --import /usr/share/ \
              rhn/RPM-GPG-KEY
```

5. Install the RPMs from the official RHEL 4 disks in the created database:

```
# find /vz/root/111/var/www/html/download/redhat/as4/i386/os/ \
      RedHat/RPMS -name '*.rpm' | xargs rpm -ihv --justdb\
      --dbpath /var/repo/redhat-as4 --ignoresize --force --nodeps
```

   Installing all RPM packages for the RHEL 4 distribution may take a rather long run; please wait until the installation process completes. After that, you can start using the `up2date` utility to update the created repository.

   For example, the following session updates the RPM packages in your local repository inside Container 111:

   - Obtain a list of RPM packages

```
# up2date -l --tmpdir=/tmp/up2date --dbpath /var/repo/redhat-as4/ | \
         awk 'BEGIN { stage = 0; } \
          stage == 0 && /^--*$/ {stage = 1; next; } \
          stage == 1 && /^$/ { stage = 2; next } \
         stage == 1 { print $1; }' \
          > /tmp/pkgs-list
```

   - Download them:

```
# cat /tmp/pkgs-list | xargs up2date -d --tmpdir=/tmp/up2date \
     --dbpath /var/repo/redhat-as4/
# rpm -ivh --justdb --dbpath /var/repo/redhat-as4/ --ignoresize \
     --force --nodeps /tmp/up2date/*.rpm
# mv /tmp/up2date/*.rpm /vz/root/111/var/www/html/download/redhat/ \
    updates/as4/i386/
# createrepo /vz/root/111/var/www/html/download/redhat/updates \
            /as4/i386
```

   You can also make a script to automatically perform the aforementioned operations and set this script to be run as a `cron` job.

Creating a Proxy Server for EZ Templates

Along with setting up the package repositories described in the previous subsections, you can create special caching proxy servers allowing to efficiently manage your OS and application EZ templates.

Setting Up a Proxy Server for EZ Templates

Parallels Server Bare Metal allows you to set up special caching proxy servers and use them to efficiently manage your OS and application EZ templates. The following picture demonstrates an example of the network containing two servers and a separate proxy server:

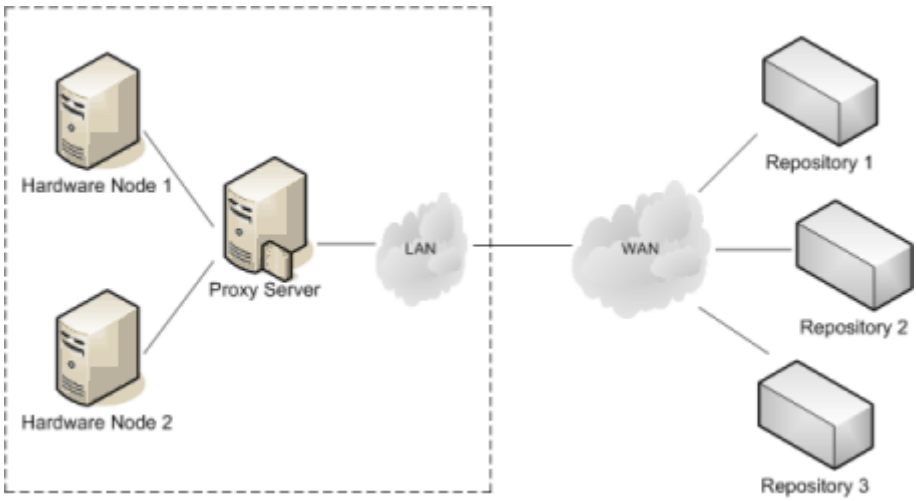


Figure 1: Virtuozzo Network With Caching Proxy Server

The caching proxy server sits between the servers (*Hardware Node 1* and *Hardware Node 2*) keeping a number of EZ templates and the repositories (*Repository 1*, *Repository 2*, and *Repository 3*) storing the packages for these EZ templates. When either server requests certain packages from any repository for the first time, these packages are first downloaded to the proxy server where they are cached and then returned to the corresponding server. From this moment on, if a server requests a package already available in the cache on the proxy server and there are no updates for this package in the official repositories, the package will be immediately downloaded to the server without requesting the repositories on the Internet for this package. At the same time, if one or several updates are available for the requested package in the official repositories, the procedure of handling these updates is identical to that of the main packages, i.e. the found updates are downloaded to the proxy server, cached there, and then retransmitted to the corresponding server.

Besides, a special Parallels script is automatically launched on the proxy server at stated intervals. During its execution, the script creates a local repository on the basis of the cached packages (please note that the script does not remove the cached packages; so, they remain intact until you delete them manually). The packages in the local repository are organized in the same way as in the original repository. So, you can make the `vzpkg` tool use the packages from the created local repository on the proxy server, which may be useful, for example, for migrating a Container that is based on an OS EZ template containing one or more outdated packages. In this case the Container migration may fail if the following conditions are simultaneously met:

- The destination server does not have any of the outdated software packages included in the OS EZ template the Container to be migrated is based on.
- The missing packages are not available in the official repositories set to handle the OS EZ template; so, they cannot be downloaded and installed on the destination server. The absence of certain software packages in the official repositories may come out of the fact that these packages have become obsolete in the course of time and have been replaced with newer versions.

You can easily avoid the aforementioned problem by setting up a caching proxy server that will store both the outdated and newer versions of the necessary packages and using this proxy server for handling your OS and application EZ templates.

In general, setting up a proxy server for managing EZ templates has the following advantages:

- Your Internet bandwidth consumption is greatly reduced since all packages are downloaded to the proxy server only once and can then be used by any server on your network.
- You can more rapidly apply software updates to your Containers since the proxy server where the downloaded packages are stored resides in the local network.
- You can always have the software packages included in your OS EZ templates at hand and do not have to worry whether they have been changed in or removed from their original repositories.
- The local repository created by the Parallels script on the proxy server enables you to effectively manage your EZ templates even if some packages are not any more available in the corresponding official repository or the server keeping the original repository went down for some reason or other.

Parallels Server Bare Metal provides you with the `vzpkgproxy` utility allowing to automatically set up any computer on your network to serve as a caching proxy server for your EZ templates. During its installation, the utility performs all the tasks necessary to install, configure, and put into operation your proxy server. `vzpkgproxy` is located in the `/virtuozzo/RPMS` directory of your Parallels Server Bare Metal distribution and can be installed with the `rpm -i` command on any servers (including Containers) meeting the following requirements:

- The Apache `httpd` server, version 2.0.53 and higher, should be installed on the server.
- The `createrepo` package, version 0.4.2 and higher, should be installed on the server.

Please keep in mind that you may also need to install a number of additional packages to satisfy the `vzpkgproxy` dependencies (e.g. the `libstdc++.so.5` package).

Note: If you are going to have one or more Containers running Red Hat Enterprise Server 4 or 5, you should use the `vzrhnproxy` utility to create special RHN Proxy Servers enabling you to effectively manage the packages included in the RHEL 4 and 5 OS EZ templates. Detailed information on this utility is provided in the next subsection.

By default, the proxy server is configured to cache the packages from all external hosts. However, you can modify the `CACHE_DISABLE` parameter in the `/etc/vzpkgpoxy/vzpkgproxy.conf` file on the proxy server to explicitly specify the hosts to be excluded from the caching process. You can also configure the behavior of the proxy server by performing the following operations:

- Modifying the port number on which the `httpd` daemon running on your proxy server is listening by setting the desired port number in the `/etc/httpd/conf.d/vzproxy.conf` file on the proxy server; the default port number is 8080.
- Editing the `REPO_DIR` parameter in the `vzpkgproxy.conf` file to change the path to the directory where the local repository created on the basis of the cached packages will be stored. By default, this directory has the path of `/var/www/html/download`.

- Making the `vzpkg` tool use the packages from the local repository on the proxy server while handling your EZ templates. To do this, you should edit the `/etc/vztt/vztt.conf` file on the server and specify:
  - the URL of the proxy server, the port number where the `httpd` daemon is listening, and the path to the directory where the EZ templates local repository is located as the value of the `VZTT_PROXY` parameter. For example, if your proxy server has the `127.123.123.127` IP address assigned, the `httpd` daemon running on the proxy server is listening on port 8080, and the local repository is stored in `/var/www/html/download`, you should set the `VZTT_PROXY` parameter to `http://127.123.123.127:8080/download`.
  - the URL of the proxy server and the port number where the `httpd` daemon is listening as the value of the `HTTP_PROXY` parameter. For example, you should set this value for the aforementioned proxy server to `http://127.123.123.127:8080`. Keep in mind that you also need to set the `HTTP_PROXY_PASSWORD` and `HTTP_PROXY_USER` parameters in the `/etc/vztt/vztt.conf` file if the access to your proxy server is password-protected.

Setting Up an RHN Proxy Server for RHEL OS EZ Templates

If some of your Containers are to run the Red Hat Enterprise Linux 4 (RHEL 4) or 5 (RHEL 5) distribution, you may wish to create a special caching proxy server - *RHN (Red Hat Network) Proxy Server* - allowing for faster RHEL packages downloads, easier distribution, and lower bandwidth requirements. RHN Proxy Servers can be created using the `vzrhnproxy` utility shipped with Parallels Server Bare Metal. This utility can be installed on any computer (including Containers) running the RHEL 4 and RHEL 5 Linux distributions with the `rpm -i` command.

Notes:

1. You may need to install a number of additional packages to satisfy the `vzrhnproxy` dependencies.

2. You can also try to deploy an RHN Proxy Server on systems running other RHEL-based distributions (e.g. CentOS 5 or Fedora 10), but `vzrhnproxy` has not been extensively tested with them.

Let us assume that you wish to create an RHN Proxy Server on the server with the IP address of `192.168.10.10` that will serve all Containers running the 32-bit version of RHEL 5 and residing on the server with the hostname of `mycomputer1` and the IP address of `192.168.0.125`. To do this, perform the following operations:

1. Log in to the server where you are planning to create the RHN Proxy Server (further referred to as Proxy Server) and make sure the `vzrhnproxy` utility is installed on this server.
2. Specify a valid user name and password you use to log in to Red Hat Network (RHN) as the values of the `REDHAT_LOGIN` and `REDHAT_PASSWORD` parameters, respectively, in the `/etc/vz/pkgproxy/rhn.conf` file on the Proxy Server. These credentials will be used by `vzrhnproxy` in the next step to register your system profile with RHN. For example:

```
# vi /etc/vz/pkgproxy/rhn.conf
REDHAT_LOGIN="user1"
REDHAT_PASSWORD="2WSX0OKM"
...
```

3. Execute the following command on the Proxy Server:

```
# vzrhnproxy register i386 5Server mycomputer1 192.168.0.125
registering for i386-5Server-mycomputer1
...
```

   where `i386` and `5Server` denote the system architecture and the operating system you wish to register with RHN (in our case, we are registering the 32-bit version of the Red Hat Enterprise Linux 5 server).

   During the command execution, `vzrhnproxy` will do the following:

   - Connect to Red Hat Network (available at http://rhn.redhat.com) with the credentials specified in the `rhn.conf` file in the previous step.
   - Create a profile and register it with RHN for the system running the 32-bit version of RHEL 5.
   - Download the headers of the packages comprising the 32-bit RHEL 5 distribution to the Proxy Server.

- Create a pseudo-repository containing the repodata generated on the basis of the downloaded headers.
- Grant the server with the IP address of `192.168.0.125` (i.e. our server) access to the Proxy Server.

4 On the server:

- Open the `/vz/template/conf/vztt/url.map` file for editing (e.g. using `vi`) and change the value of the `$RH_SERVER` parameter as follows:

```
$RH_SERVER    http://192.168.10.10/rhn
```

- Save the file.

From this moment on:

- If the server with the IP address of `192.168.0.125` requests certain packages included in the RHEL 5 distribution for the first time (e.g. while caching the RHEL 5 OS EZ template), this request will be sent to the Proxy Server which, in its turn, will connect to Red Hat Network and retrieve the requested packages. These packages will then be downloaded to the Proxy Server where they are cached and finally returned to the server.
- If the server requests a package already available in the cache on the Proxy Server, the package will be immediately downloaded from the cache to the server.

You can make the Proxy Server serve the requests for RHEL 5 packages from more than one server. To do this, you should specify the IP addresses of the corresponding servers during the `vzrhnproxy register` command execution and properly edit the `/vz/template/conf/vztt/url.map` file on each of these servers (see Step 3 and 4 above). Keep in mind that, while executing the `vzrhnproxy register` command, you should specify the hostname of one server only; this can be the hostname of any server to be handled by the Proxy Server.

You can also create and register several system profiles with Red Hat Network. For example, if you have one or more servers hosting Containers with the x86-64-bit version of RHEL 5, you may wish to use the Proxy Server for handling the packages included in this RHEL 5 version as well. To do this, you should perform once more Steps 1-4 described above and use the following command on Step 3 to register a new system profile with RHN:

```
# vzrhnproxy register x86_64 5Server mycomputer2 192.168.22.22
registering for x86_64-5Server-mycomputer2
...
```

where `192.168.22.22` is the IP address of the server hosting 64-bit Containers.

To list all system profiles registered with RHN, you can execute the following command on the Proxy Server:

```
# vzrhnproxy list
i386-5Server-mycomputer1
x86_64-5Server-mycomputer2
x86_64-5Server-mycomputer3
```

As you can see, three system profiles are currently registered with RHN: two for servers running the x86-64-bit version of RHEL 5 and one for the server running the 32-bit version of RHEL 5. For each of these profiles, the corresponding pseudo-repository containing the RHEL 5 package repodata exists on the Proxy Server.

After a lapse of time, the repodata (and, consequently, the cache on the Proxy Server) may become obsolete. In this case you can use the `vzrhnproxy update` command to update the repodata in pseudo-repositories on the Proxy Server. For example, the following command will update the repodata in the pseudo-repository corresponding to the `i386-5Server-mycomputer1` profile:

```
# vzrhnproxy update i386-5Server-mycomputer2
```

Preparing an OS Template for Container Creation

OS EZ templates are used to create Containers on their basis. To prepare an OS EZ template for the Container creation, you should:

1. Install the OS EZ template on the server.
2. Cache the installed OS EZ template.

To install a new OS EZ template on the server, use the `vzpkg install template` command. For example, to install the Red Hat Enterprise Linux 5 OS EZ template, you can run this command:

```
# vzpkg install template redhat-el5-x86-ez-4.0.0-4.swsoft.noarch.rpm
Preparing...                ########################################### [100%]
   1:redhat-el5-x86         ########################################### [100%]
# vzpkg list
redhat-el5-x86
```

As you see, the `redhat-el5-x86` EZ template is now installed on the server. The corresponding path is `/vz/template/redhat/el5`. However, before the `redhat-el5-x86` EZ template can be used as a basis for the Container creation, it should first be cached. This can be done by using the `vzpkg create cache` command:

Note: Before you can start caching your OS EZ templates, you may need to set up a package repository for them. So, you have to build a special repository for all commercial versions of the Linux distributions (e.g. Red Hat Linux Enterprise 4 or 5). Detailed information on how to manage package repositories is provided in the Setting Up Repositories and Proxy Servers for EZ Templates section (p. 22).

```
# vzpkg create cache redhat-el5-x86
...
Complete!
Packing cache file redhat-el5-x86.tar.gz ...
Cache file redhat-el5-x86.tar.gz [14M] created.
```

The created tar archive is put to the `/vz/template directory` on the server:

```
# ls /vz/template/cache
redhat-el5-x86.tar.gz
```

After the `redhat-el5-x86` EZ template has been successfully cached, you can start creating Container on its basis. Detailed information on how to create Containers on the basis of OS EZ templates is provided in the *Parallels Server Bare Metal User's Guide*.

Installing Application EZ Templates

The same way as you use an OS EZ template in a Parallels Server Bare Metal system to create any number of Containers on its basis and share its resources, you can use application EZ templates to share package files among any number of Containers. You can then add these applications to any number of Containers.

To install a new application EZ template on the server, you should use the `vzpkg install template` command. For example, to install the `mysql` EZ template intended to be run on the Red Hat Enterprise Linux 5 distribution, you can issue the following command:

Note: If you are running one of the rpm-based Linux distributions (e.g. Red Hat Enterprise Linux 5 or Fedora 10), you can also use the `rpm -Uhv` command to install application EZ templates on your server.

```
# vzpkg install template mysql-redhat-el5-x86-ez-3.0.0-4.swsoft.rpm
Preparing...                ########################################### [100%]
   1:mysql-redhat-el5-x86   ########################################### [100%]
```

The `mysql` EZ template will be installed in the `/vz/template/redhat/el5/x86/config/app/mysql` directory on the server. To make sure that this EZ template has been successfully installed, you can use the `vzpkg list` command:

```
# vzpkg list
redhat-el5-x86                  2009-05-21 02:22:45
redhat-el5-x86  mysql
```

As you see, the `mysql` EZ template is now available on the server and can be added to any number of Containers.

Listing EZ Templates

The `vzpkg list` command allows you to list the EZ templates installed on the server. They may be already used or not used by certain Containers:

```
# vzpkg list
redhat-el5-x86
fedora-core-10-x86       2009-01-24 15:45:15
```

As you see, the `redhat-el5-x86` and `fedora-core-10-x86` EZ templates are available on the server. The characters opposite the `fedora-core-10-x86` EZ template informs you of the date and time when software packages included in the template were updated for the last time. In its turn, the characters absence beside the `redhat-el5-x86` EZ template indicates that the template has not yet been cached at all.

Specifying a Container number as the parameter, `vzpkg list` prints the EZ templates used by the specified Container:

```
# vzpkg list 101
fedora-core-10-x86       2009-01-24 15:45:15
```

Adding an Application EZ Template to a Container

To add an application EZ template to an existing Container, you should use the `vzpkg install` command. To successfully add an application EZ template to a Container, this Container must be running. Otherwise, it is impossible to run the installation process in the Container context.

In the example below, the `mysql` application EZ template meant for the usage with Red Hat Enterprise Linux 5 and already installed on the Parallels server is added to Container 101:

```
# pctl status 101
CTID 101 exists mounted running
# vzpkg list
redhat-el5-x86             2009-05-21 02:21:56
redhat-el5-x86   mysql
...
# vzpkg install 101 mysql
...
Installed:
 mysql                  i386      0:4.1.12-3.RHEL5.1
 mysql-bench            i386      0:4.1.12-3.RHEL5.1
 mysql-devel            i386      0:4.1.12-3.RHEL5.1
...
```

Keeping EZ Templates Up To Date

Parallels Server Bare Metal allows you to update your OS and application EZ templates as follows:

- Update any of the EZ templates installed on the server.
- Update the caches of OS EZ templates installed on the server.
- Update the packages that are included in the EZ templates (OS and application) applied to particular Containers.

All the aforementioned operations are described in the following subsections in detail.

Updating EZ Templates on the Parallels Server

Sometimes, you may need to update one or more EZ templates (either OS or application) installed on your server. The process of updating EZ templates consists in updating one or more EZ template configuration files located in the `/vz/template/<os_name>/<os_version>/<arch>/config` directory on the server. Parallels Server Bare Metal allows you to use one of the following tools to update the EZ templates installed on the server:

- the `vzup2date` utility
- the `vzpkg update template` utility

Updating Templates with vzup2date

The `vzup2date` utility allows you to update any of the EZ templates installed on the server. This utility can also be used to download new EZ templates to the server and install them there. `vzup2date` can be launched in two modes:

- graphical mode
- command line mode

Updating EZ templates in the graphical mode takes place if you have executed the `vzup2date` utility with the `-z` option. After launching the utility, you will be presented with a greeting screen:

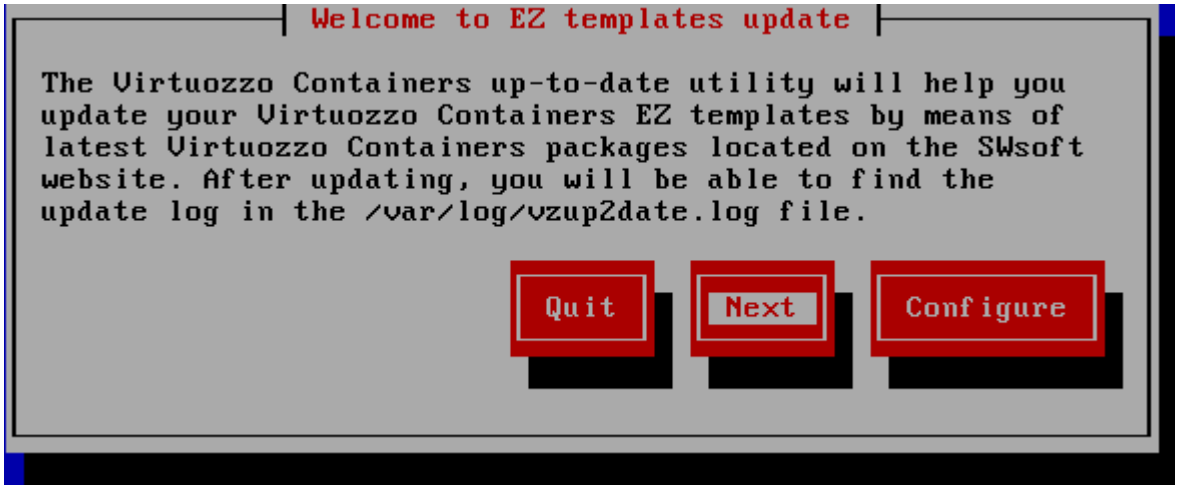


Figure 2: Updating System - Welcome Screen

In this window you can do one of the following:

- Click the Next button to connect to the repository storing the latest EZ templates (either the Parallels default repository or your own one).
- Click the Configure button to configure the parameters used to connect to the EZ templates repository.

As soon as you press Next in the Welcome window, the utility will try to connect to the EZ templates repository (either the Parallels default repository or your own one) and, if the connection is successful, display the EZ Templates Selection window listing all EZ templates that have one or more updates available or that are not installed on your Node at all. For example:

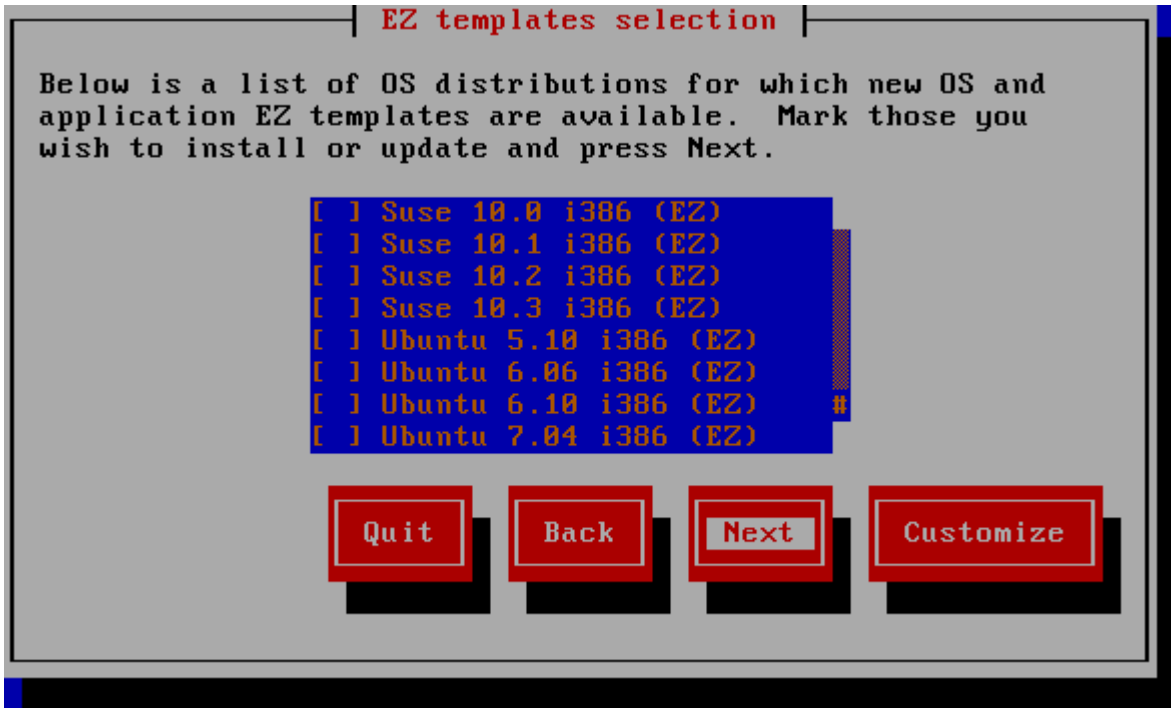


Figure 3: Updating Templates - Selecting Linux Distribution

This window allows you do one of the following:

- If you wish to download and install all available EZ templates/template updates for a certain Linux distribution, select this distribution by placing the cursor beside it and pressing the space bar on your keyboard; then click Next.
- If you wish only certain EZ templates of the corresponding Linux distribution to be installed/updated on the server, place the cursor beside this distribution and press F2 on your keyboard. You will be presented with the Templates selection window where you can select the corresponding EZ templates:

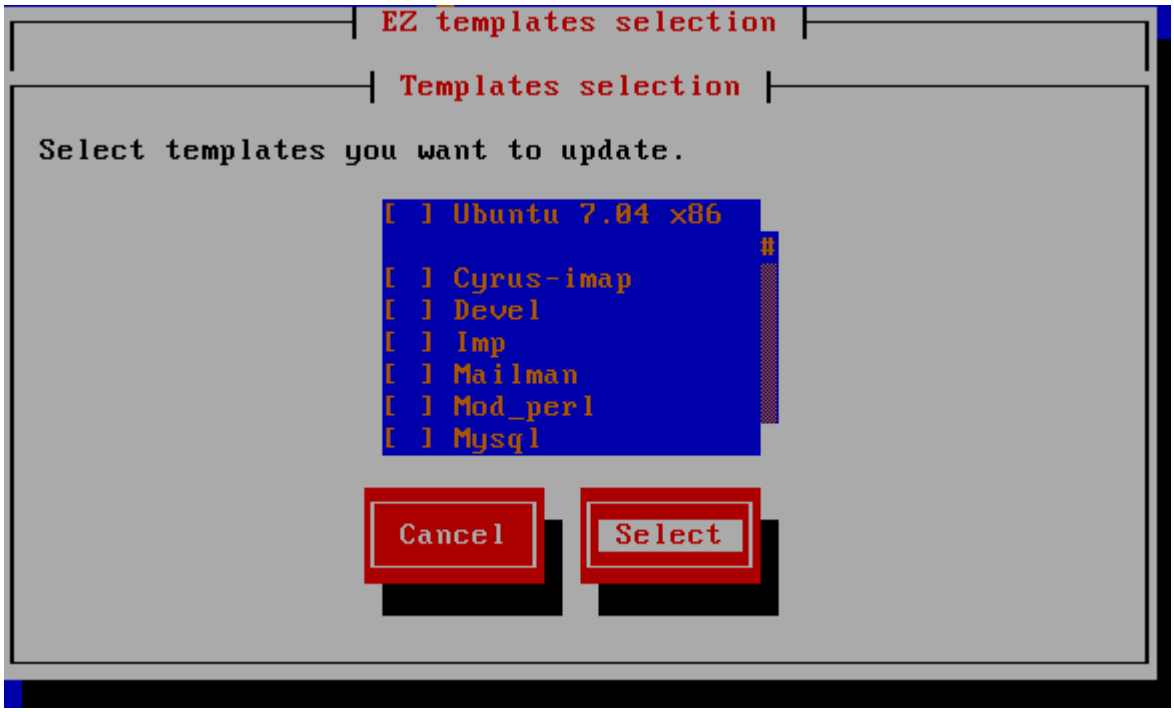


Figure 4: Updating Templates - Selecting EZ Templates

After choosing the right EZ templates, click the Select button to close the displayed window and then click Next to proceed with the wizard.

Note: New application EZ templates for a Linux distribution can be installed on the server only if the corresponding OS EZ template is already available on it.

On the next step, you can review the EZ templates/template updates you selected on the previous step and scheduled for downloading and installing on your server. If you are not satisfied with the chosen templates/template updates, click the Back button to return to the previous step and modify the set of templates; otherwise, click Next to start downloading the templates/template updates to the server.

After the EZ templates/template updates have been successfully downloaded to the server, the Installing EZ template window is displayed:

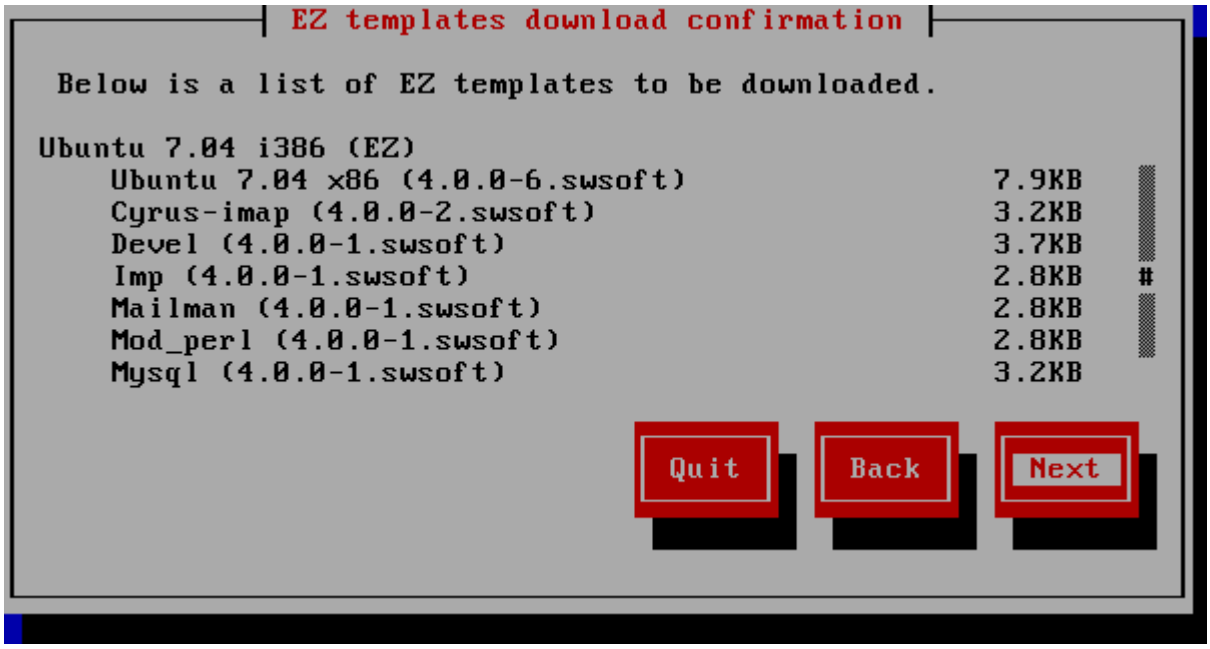


Figure 5: Updating Templates - Viewing EZ Templates to Install

In this window you can view the templates/template updates ready to be installed on your server. If you are installing a new OS EZ template/template update, you can make use of the Run vzpkg cache after installation check box to cache the corresponding OS EZ template/template update right after its installation on the server. By default, all OS EZ templates are just installed on the server without being cached; however, you can select the provided check box and schedule your OS EZ template/template update for caching. Clicking Next starts installing the selected EZ templates/template updates on the server. By the time the wizard finishes you should have updated OS and application EZ templates on your system.

Another way of updating your EZ templates is to run the `vzup2date` utility in the command line mode, which can be done by passing the corresponding commands, switches, and options to `vzup2date`. For example, the following command will update the `fedora-core-10-x86` OS EZ template to the latest version:

```
# vzup2date -z -m batch install fedora-core-10-x86
```

Detailed information on all options that can be passed to the `vzup2date` utility is given in the *Parallels Command Line Reference Guide*.

Updating Templates With vzpkg update template

Another way of updating your EZ templates installed on the server is to use the `vzpkg update template` utility. This utility allows you to update OS or application EZ templates from the corresponding local RPM packages. For example, you can execute the following command to update the CentOS 5 OS EZ template installed on the server from the `centos-5-x86-ez-4.0.0-2.swsoft.noarch.rpm` package located in the `/root` directory:

```
# vzpkg update template /root/centos-5-x86-ez-4.0.0-2.swsoft.noarch.rpm
```

You can update a number of EZ templates at once by specifying the corresponding packages and separating them by spaces. For example, the following command

```
# vzpkg update template /root/centos-5-x86-ez-4.0.0-2.swsoft.noarch.rpm
/root/redhat-el5-x86-ez-4.0.0-4.swsoft.noarch.rpm
```

will simultaneously update the CentOS 5 and Red Hat 5 OS EZ templates installed on the server.

Updating OS EZ Template Caches

With the release of new updates for the corresponding Linux distribution, the created OS EZ template cache can become obsolete. So, Parallels Server Bare Metal provides the `vzpkg update cache` command allowing you to quickly update any of the OS EZ template caches available on the server.

Note: If you are going to update the cache of a commercial OS EZ template (e.g. Red Hat Enterprise Server 5 or SLES 10), you should first update software packages in the remote repository used to handle this OS EZ template and then proceed with updating the EZ template cache. Detailed information on how to manage repositories for commercial Linux distributions is provided in the Setting Up Repositories and Proxy Servers for EZ Templates section (p. 22).

When executed, `vzpkg update cache` checks the `cache` directory in the template area (by default, the template area is located in `/vz/template`) on the server and updates all existing tarballs in this directory. However, you can explicitly indicate the tarball for what OS EZ template should be updated by specifying the OS EZ template name. For example, to update the tarball for the `fedora-core-10-x86` OS EZ template, you should issue the following command:

```
# vzpkg update cache fedora-core-10-x86
Loading "rpm2vzrpm" plugin
Setting up Update Process
Setting up repositories
base0            100% |=========================|  951 B    00:00
base1            100% |=========================|  951 B    00:00
base2            100% |=========================|  951 B    00:00
base3            100% |=========================|  951 B    00:00
...
```

Upon the `vzpkg update cache` execution, the old tarball is renamed by receiving the `-old` suffix (e.g. `fedora-core-10-x86.tar.gz-old`):

```
# ls /vz/template/cache
fedora-core-10-x86.tar.gz  fedora-core-10-x86.tar.gz-old
```

You can also pass the `-f` option to `vzpkg update cache` to remove an existing tar archive and create a new one instead of it.

If the `vzpkg update cache` command does not find a tarball for one or several OS EZ templates installed on the server, it creates tar archives of the corresponding OS EZ templates and puts them to the `/vz/template/cache` directory.

Updating EZ Templates Packages Inside a Container

Parallels Server Bare Metal allows you to update software packages of the OS EZ template a Container is based on and of any application EZ templates applied to the Container. You can do it by using the `vzpkg update` utility. Assuming that Container 101 is based on the `redhat-el5-x86` OS EZ template, you can issue the following command to update all packages included in this template:

```
# vzpkg update 101 redhat-el5-x86
...
  Updating: httpd                ######################### [1/4]
  Updating: vzdev                ######################### [2/4]
  Cleanup : vzdev                ######################### [3/4]
  Cleanup : httpd                ######################### [4/4]

Updated: httpd.i386 0:2.0.54-10.2 vzdev.noarch 0:1.0-4.swsoft
Complete!
Updated:
 httpd                  i386      0:2.0.54-10.2
 vzdev                  noarch    0:1.0-4.swsoft
```

Notes:

1. A Container has to be running in order to update EZ templates inside this Container.

2. If you are going to update the cache of a commercial OS EZ template (e.g. Red Hat Enterprise Server 5 or SLES 10), you should first update software packages in the remote repository used to handle this OS EZ template and then proceed with updating the EZ template cache. Detailed information on how to manage repositories for commercial Linux distributions is provided in the Setting Up Repositories and Proxy Servers for EZ Templates section (p. 22).

As you can see from the example above, the `httpd` and `vzdev` applications have been updated for the `redhat-el5-x86` OS EZ template. If you wish to update all EZ templates (including the OS EZ template) inside Container 101 at once, you should execute the following command:

```
# vzpkg update 101
...
Running Transaction
  Updating  : hwdata               ######################### [1/2]
  Cleanup   : hwdata               ######################### [2/2]

Updated: hwdata.noarch 0:1.0-3.swsoft
Complete!
Updated:
 hwdata                 noarch    0:0.158.1-1
```

In the example above, only the `hwdata` package inside Container 101 was out of date and updated to the latest version.

Creating Historical Mirrors for Backed Up Containers

If you have one or several Containers that are based on OS EZ templates and that were backed up long time ago, you may come across problems when trying to restore them on a destination server other than the source server (i.e. the server where the Containers were hosted during their backing up). This may happen when the following conditions are simultaneously met:

- The destination server does not have one or several software packages included in the OS EZ template the Container being restored is based on.
- The missing packages are not available in the public repositories set to handle the OS EZ template; so, they cannot be downloaded and installed on the destination server. The absence of certain software packages in the public repositories may come out of the fact that some packages that were installed in the Container at the moment of its backing up have become obsolete in the course of time and been replaced with newer versions.

To avoid the aforementioned problems in the future, you can create the so-called 'historical' mirrors which will store an archive of all software packages present in the public repositories and containing the packages installed in the Container during its backing up.

Let us assume that you backed up a number of Containers based on the Fedora 10 OS EZ template. Now to be sure that you always have a repository containing all the necessary RPM packages for your backed up Containers, you wish to create a historical mirror of an official Fedora 8 repository. In our example below, we presume the following:

- The historical mirror will be located inside Container 101. You can use any OS template to base the Container on.

  Note: We recommend that you always place your local repositories inside separate Containers to not compromise the server security. In particular, it is of significant importance if you are going to provide access to your repositories through the `http` and `ftp` protocols.

- Container 101 is started and has the IP address of `123.145.145.123` assigned to it.
- The mirror will be created on a web server, i.e. it can be accessed from other networks through the `http` protocol.
- The `apache` web server is installed and running inside Container 101; the default document root for `apache` is `/var/www/html`, i.e. the `apache` web server stores its sites in the `/vz/root/101/var/www/html` directory on the server.
- The `apache` user and group inside Container 101 are `apache`.

To create a historical mirror for Fedora 10, you should perform the following operations:

1. Change to the `/vz/root/101/var/www/html` directory and create the `fed8mirror` subdirectory within it:

```
# cd /vz/root/101/var/www/html
# mkdir fed10mirror
```

   The `fed10mirror` subdirectory will store an archive of the Fedora 10 repository.

2. Change to the `fed10mirror` subdirectory

```
# cd fed10mirror
```

and execute the following command:

```
# rsync -av http://ftp.rhd.ru/pub/fedora/linux/releases/10/Everything/i386/os
```

This command will make a copy of the entire Fedora 8 repository located at http://ftp.rhd.ru/pub/fedora/linux/releases/10/Everything/i386/os. Keep in mind that it may take a rather long run to copy all RPM packages to Container 101 depending on your bandwidth and the load on the Fedora mirror server.

Note: You can use any alternative Web or FTP site containing the Fedora 10 repository instead of the one indicated above.

3. Add your historical mirror to a list of repositories to be checked while performing operations on EZ templates related to Fedora 10 (in particular, while restoring Fedora-based Container backups). You can do it as follows:

   - Create the `/vz/template/fedora-core/10/x86/config/os/default/repositories` file on the server, if it is not yet present.
   - Add the following string to the file:

```
http://123.145.145.123/var/www/html/fed8mirror
```

   - Save the file.

Copying EZ Templates to Another Server

Parallels Server Bare Metal allows you to copy the installed OS and application EZ templates from one server to another using the `vzmtemplate` utility. For example, you can copy the `fedora-core-10-x86` OS template installed on the source server to the destination server with the IP address of `192.168.0.9` by executing the following command:

```
# vzmtemplate -z root@192.168.0.9 fedora-core-10-x86
root@192.168.0.9's password:
Connection to destination server (192.168.0.9) is successfully established
Copying Template ".fedora-core-10-x86"
...
```

During the command execution, `vzmtemplate` will do the following:

1. Ask you for the password of the `root` user on the destination server.
2. Check whether the `fedora-core-10-x86` OS EZ template already exists on the destination server. If this templates is installed on the Destination Node, the command will exit.
3. Copy the `fedora-core-10-x86` configuration files from the source server to the destination server. Please keep in mind that the `fedora-core-10-x86` OS EZ template is not removed from the source server.
4. Run the `vzpkg create cache` command on the destination server to cache the OS EZ template and prepare it for the Container creation. Detailed information on this command is provided in the Preparing an OS EZ Template for Container Creation section (p. 37).

Note: If you are going to copy an application EZ template, make sure that the corresponding OS EZ template (i.e. the OS template with which the application EZ template can be used) is installed on the destination server. Otherwise, the operation will fail.

To check that the `fedora-core-10-x86` OS template has been successfully copied to the destination server, you can run the following command on this server:

```
# vzpkg list
fedora-core-10-x86                  2009-6-12 07:05:39
```

Removing an Application from a Container

The `vzpkg remove` command allows you to remove one or several application EZ templates from a Container. A session below shows you an example how to remove the `mysql` EZ template from Container 101:

```
# vzpkg list 101
redhat-el5-x86                  2007-05-21 02:21:56
redhat-el5-x86   mysql             2007-05-21 05:36:42
# vzpkg remove 101 mysql -w
vzpkg remove 101 mysql
Removed:
 mysql
 mysql-server
 mysql-devel
 mysql-bench
 perl-DBD-MySQL
# vzpkg list 101
redhat-el5-x86                  2007-05-21 02:21:56
```

You can see that the `mysql` EZ template has been successfully removed from Container 101. The `-w` option tells the `vzpkg remove` command to also delete from the Container all packages having interdependencies with `mysql`.

Removing EZ Template From the Parallels Server

The `vzpkg remove template` command allows you to remove from the Parallels server those EZ OS and application templates that you do need any more. The process of removing an EZ template includes deleting all the RPM packages comprising this template and all the caches available for this template (for EZ OS templates only). Please keep in mind that the template to be removed should not be applied to any Container. Otherwise, the template deletion will fail. A session below demonstrates how to remove the `redhat-el5-x86` EZ OS template from the Parallels server:

```
# vzpkg list
redhat-el5-x86                    2009-02-16 12:50:17
fedora-core-10-x86                2009-02-18 14:23:12
# vzpkg remove template redhat-el5-x86
redhat-el5-x86 template was removed
# vzpkg list
fedora-core-10-x86                2009-02-18 14:23:12
```

To remove an EZ application template from the server, you should additionally specify the `-F` option after the `vzpkg remove template` command. This option denotes the EZ OS template with which the EZ application template is compatible. For example, the following command can be used to remove the `mailman` EZ application template that is intended for running under RHEL 5 from your server:

```
# vzpkg remove template -F redhat-el5-x86 mailman
redhat-el5-x86 mailman template was removed
```

You can also remove several EZ templates at once by specifying their names after `vzpkg remove template` and separating them by spaces. However, when handling application templates, keep in mind that you can delete only those application templates that relate to one and the same EZ OS template. For example:

```
# vzpkg remove template -F redhat-el5-x86 sitebuilder4 mailman
redhat-el5-x86 sitebuilder4 template was removed
redhat-el5-x86 mailman template was removed
```

In this example the `sitebuilder4` and `mailman` EZ application templates intended to run under RHEL 5 have been successfully removed from the Parallels server.

Glossary

Application template. A template used to install a set of applications in *Containers*. See also *Template*.

Container (or *regular Container*). A virtual private server, which is functionally identical to an isolated standalone server, with its own IP addresses, processes, files, its own users database, its own configuration files, its own applications, system libraries, and so on. Containers share one *Parallels server* and one OS kernel. However, they are isolated from each other. A Container is a kind of 'sandbox' for processes and users.

Guest operating system (Guest OS). An operating system installed inside a virtual machine and Container. It can be any of the supported Windows, Linux, or Mac operating systems.

Hardware virtualization. A virtualization technology allowing you to virtualize physical servers at the hardware level. Hardware virtualization provides the necessary environment for creating and managing Parallels virtual machines.

Operating system virtualization (or *OS virtualization*). A virtualization technology allowing you to virtualize physical servers at the operating system (kernel) level. OS virtualization provides the necessary environment for creating and managing Parallels Containers.

OS template (or *Operating System template*). A template used to create new *Containers* with a pre-installed operating system. See also *Template*.

Package set. See *Template*.

Parallels Management Console. A Parallels Server Bare Metal management and monitoring tool with graphical user interface. Parallels Management Console is cross–platform and can run on Microsoft Windows, Linux, and Mac computers.

Parallels Server. A hardware virtualization solution that enables you to efficiently use your physical server's hardware resources by sharing them between multiple virtual machines created on this server.

Parallels server (or *physical server* or *server*). A server where the Parallels Server Bare Metal software is installed for hosting Parallels virtual machines and Containers. Sometimes, it is marked as Container 0.

Parallels Server Bare Metal license. A special license that you should install on the physical server to be able to start using Parallels Server Bare Metal. Every physical server must have its own license installed.

Parallels Virtuozzo Containers for Linux. An operating system virtualization solution allowing you to create multiple isolated Containers on a single physical server to share hardware, licenses, and management effort with maximum efficiency.

Private area. A part of the file system storing *Container* files that are not shared with other *Containers.*

Template (or *package set*). A set of original application files (packages) repackaged for mounting over Virtuozzo File System. There are two types of templates. OS Templates are used to create new *Containers* with a pre-installed operating system. Application templates are used to install an application or a set of applications in *Containers*.

UBC. An abbreviation of *User Beancounter*.

User Beancounter. The subsystem of the Parallels Server Bare Metal software for managing *Container* memory and some system-related resources.

Virtual Environment (or *VE*). An obsolete designation of a *Container*.

Virtuozzo File System (*VZFS*). A virtual file system for mounting to Container private areas. VZFS symlinks are seen as real files inside *Containers*.

Virtual machine (VM). A computer emulated by Parallels Server Bare Metal. Like a Container, a virtual machine is functionally identical to an isolated standalone computer, with its own IP addresses, processes, files, its own users database, its own configuration files, its own applications, system libraries, and so on. However, as distinct from Containers, virtual machines run their own operating systems rather than sharing one operating system kernel.

Index