Virtuozzo 6

Templates Management Guide

February 17, 2016

Parallels IP Holdings GmbH
Vordergasse 59
8200 Schaffhausen
Switzerland
Tel: + 41 52 632 0411
Fax: + 41 52 672 2010
www.virtuozzo.com

Contents

CHAPTER 1

Introduction

Virtuozzo 6 is a virtualization solution that allows you to run multiple virtual machines and Containers on a single physical server.

This chapter provides general information about Virtuozzo and this guide.

- Virtuozzo basics (p. 5)
- goals and target audience of the guide (p. 6)
- guide organization (p. 6)
- resources to consult to get more information on Virtuozzo (p. 7)
- ways to submit feedback to the documentation team (p. 7)

In This Chapter

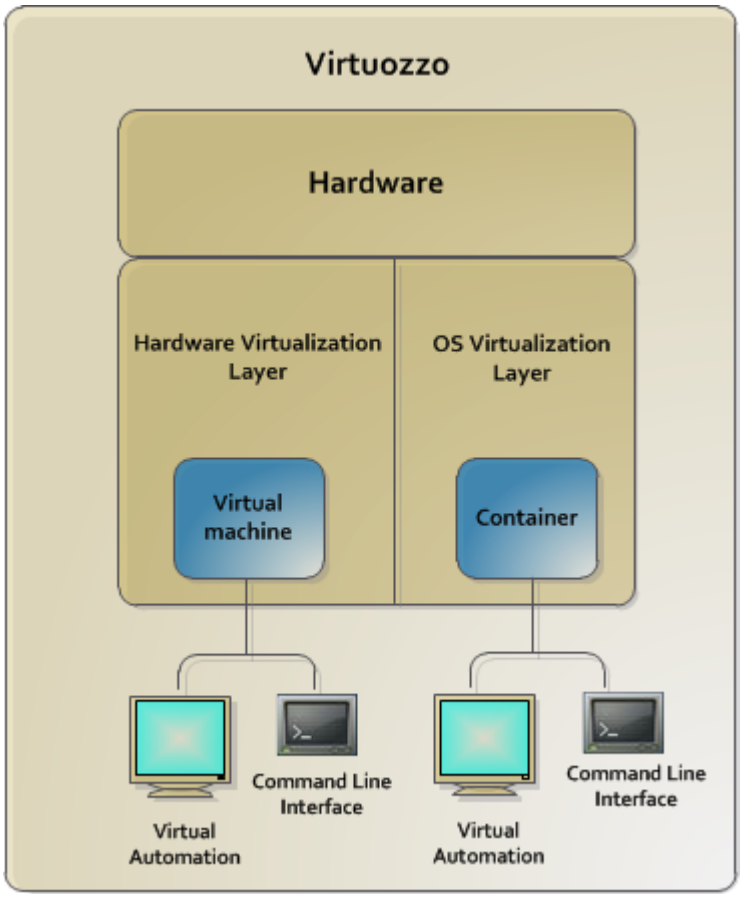
About Virtuozzo 6

Virtuozzo 6 is a virtualization solution that allows you to simultaneously run multiple virtual machines and Containers on a single physical server. With Virtuozzo, you can efficiently share your server's hardware resources among virtual machines and Containers.

Virtuozzo is installed directly on the server hardware and does not need any operating system to function. Once it is installed, Virtuozzo allows you to create virtual machines and Containers and manage them using the following tools:

- **Command-line interface (CLI)**. The command-line interface comprises a set of command-line utilities that you can use to manage virtual machines and Containers, both locally and remotely.
- **Virtual Automation**. Virtual Automation is a remote management tool that allows you to manage physical servers and their virtual machines and Containers with the help of a standard Web browser on any platform.

Graphically, a server with the Virtuozzo software installed can be represented as follows:



About This Guide

This guide is meant to provide complete information on templates, a Virtuozzo technology allowing you to efficiently deploy standard Linux applications inside your Containers and to greatly save your host 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

This guide is organized in the following way:

Chapter 1, Introduction (p. 5), gives an overview of the Virtuozzo product and this guide.

Chapter 2, Templates Overview (p. 8), provides general information on Virtuozzo templates: what templates are, the advantages of their usage in Virtuozzo-based systems, etc.

Chapter 3, Managing EZ Templates (p. 15), provides instructions on managing OS and application EZ templates. You will know how to create and install EZ templates on the host, add them to and remove them from Containers, make OS template caches and update them, etc.

Getting Help

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

- *Virtuozzo 6 Installation Guide*. This guide provides detailed information on installing Virtuozzo on your server, including the pre-requisites and the stages you shall pass.
- *Installing via PXE*. This guide provides information on installing Virtuozzo 6 over a network using a PXE (Preboot Execution Environment) server.
- *Getting Started With Virtuozzo 6*. This guide provides basic information on how to install Virtuozzo on your server, create new Containers and virtual machines, and perform main operations on them. Unlike the *Virtuozzo 6 Installation Guide*, it does not contain detailed description of all the operations needed to install and set Virtuozzo to work (for example, installing Virtuozzo in unattended mode).
- *Virtuozzo 6 User's Guide*. This guide provides comprehensive information on Virtuozzo 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.
- *Virtuozzo Storage Administrator's Guide*. This guide is intended for system administrators interested in deploying Virtuozzo Storage in their networks.
- *Virtuozzo 6 Command Line Reference Guide*. This guide is a complete reference on all Virtuozzo configuration files and command-line utilities.

Feedback

If you spot a typo in this guide, or if you have an opinion about how to make this guide more helpful, you can share your comments and suggestions with us by completing the Documentation Feedback form at http://www.virtuozzo.com/support/usersdoc/.

CHAPTER 2

Templates Overview

A template in Virtuozzo 6 is a set of application and configuration files installed on a host in such a way as to be usable by any Container. Virtuozzo provides tools for creating templates, installing and removing them on/from hosts, adding them to Containers, and so on.

Using templates, you can:

- Securely share RAM among similar applications running in different Containers to save hundreds of megabytes of memory
- 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 Virtuozzo. 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. Virtuozzo 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. Virtuozzo uses application templates to add extra software to 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 Virtuozzo, you can perform the following operations on templates:

- create new application templates
- list the templates currently installed on the host
- install templates on and remove them from the host
- add templates to Containers
- remove templates from the host and from Containers
- migrate templates from one host to another

All these operations are described in **Managing EZ Templates** (p. 15) in detail.

Note: The current version of Virtuozzo does not support using templates in virtual machines.

In This Chapter

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 host 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 host. 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 website 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/ website and storing software packages for 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 host 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 life cycle 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).

Virtuozzo 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 host 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: For details on how to manage software package repositories, see **Setting Up Repository and Proxy Servers for EZ Templates** (p. 21).

EZ Template Directory Structure

All EZ templates and the software packages included in them and installed on the host 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`).
  - *<os_version>* is the version of the Linux distribution specified as *<os_name>* (e.g., 5 or 6).
  - *<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 Centos 6 EZ template, the `/vz/template/centos/6/x86` directory on the host 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 host.
  - 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 host. An 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.
- One or several subdirectories containing the packages comprising 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 host.
  - `post-cache`: this script is executed after installing the packages included in the EZ template on the host.
  - `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 *Virtuozzo 6 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

Listed below are the major differences between OS and application EZ 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 Virtuozzo.
- You may define a list of compatible templates and a list of required templates for application templates.

EZ Template Life Cycle

An EZ template has the following development stages:

1. Any EZ template should be first installed on the host. 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 host, 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 `prlctl 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 downloaded from the network repository, installed on the host, and then 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 host with the `vzpkg remove template` command.

CHAPTER 3

Managing EZ Templates

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

In This Chapter

Creating an EZ Template

Virtuozzo is shipped with a certain number of OS (e.g., `centos-6-x86_64`) and application (e.g., `mysql-centos-6-x86_64`) EZ templates. This is usually sufficient to deploy the main Linux distributions and their applications in Containers. You can, however, create your own OS and application EZ templates and use them to base Containers on or run different applications in Containers.

You can use one of the following ways to create EZ templates:

- Make a template on the basis of an existing template (p. 16). In this case, you configure the parameters of one of the templates installed on your server to meet your needs.
- Make a template from scratch (p. 16). In this case, you first create a new template configuration and then create the template on its basis.

The following subsections describe both ways of creating templates in detail.

Creating a Template from Official Templates

The easiest way to create a template is to base it on one of the official templates shipped by Virtuozzo and adjust its configuration to meet your demands:

1. Install the template you want to use as the basis for your new template (e.g., using the `yum install` command).
2. Create a copy of the directory containing the template configuration files. Depending on whether you want to create a non-base template or a self-contained template, your steps slightly differ:
   - To create a non-base template, create a subdirectory in the `/vz/template/`*`os_name/os_version/arch`*`/config/os` directory and copy to this directory the contents of the `/vz/template/`*`os_name/os_version/arch`*`/config/os/default` subdirectory. For example, to make a non-base template for the official `centos-6-x86_64` template and put its configuration files to the `/customized` directory, you can run this command:

```
# cp -a /vz/template/centos/6/x86_64/config/os/{default,customized}
```

> **Note:** Non-base OS EZ templates inherit some 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.

   - To create a self-contained template, create a subdirectory in the `/vz/template/`*`os_name`* directory and copy to this directory the contents of the subdirectory for the original template (`/vz/template/`*`os_name/os_version`*). For example, to make a self-contained template on the basis of the official `centos-6-x86_64` template and put it to the `/vz/template/centos/6-customized` directory, you can run these commands:

```
# mkdir -p /vz/template/centos-customized/6/x86_64/config
# cp -a /vz/template/centos{,-customized}/6/x86_64/config/os
```

3. Change to the directory containing the configuration files for the new template and adjust them to meet your demands. For example, you can change the list of packages to include in the template by editing the `packages` file or configure the list of mirrors by editing the `mirrorlist` file.

Creating a Template from Scratch

The process of creating a template from scratch includes completing the following tasks:

1. Installing the `vztt-build` package on the server.
2. Making a metafile that will be used as the basis for creating the template.
3. Using the `vzmktmpl` utility to create the template from the metafile.

The following subsections describe these operations in detail.

Installing vztt-build

The first step in creating a template from scratch is to install the `vztt-build` package on your server. This package contains

- template metafile samples. You use metafile samples as the basis for creating a metafile for your own template.
- the `vzmktmpl` utility. You use the `vzmktmpl` utility to create the template on the basis of your metafile.

The `vztt-build` package is available in the remote official repository so you can install it using the following command:

```
# yum install vztt-build
```

Creating Metafiles

In the second step, you make a metafile, a plain text file having the `.metafile` extension and containing a list of parameters for your template. The `vzmktmpl` utility uses this file when creating the template.

The following example demonstrates the process of creating a metafile for the CentOS 6 x64 OS template. To create the metafile:

1. Create a new text file with an arbitrary name and the `.metafile` extension (e.g., `centos-6-custom.metafile`).

   The easiest way to create a new metafile is to copy one of the metafile samples from the `/usr/share/vztt/samples` directory and configure it to meet your demands. For example, you can use the `/usr/share/vztt/samples/centos-6-x86_64/metafile` file as the basis for creating your new metafile for CentOS 6.

```
# cp /usr/share/vztt/samples/centos-6-x86_64/metafile /home/user1
```

2. Configure the metafile to meet your demands:

   a. **%osname**. Specify the name of the Linux distribution for which you are creating the OS EZ template:

```
%osname
centos-custom
```

   The specified name will be assigned to the template directory where you will install the template (e.g., in our case, it will be `/vz/template/centos-custom`).

   b. **%osver**. Specify the version of the Linux distribution for which you are creating the template:

```
%osver
6
```

   The specified name will be assigned to the subdirectory that will denote the version of your Linux distribution (e.g., `/vz/template/centos-custom/6`).

   c. **%osarch**. Provide the information about the microprocessor architecture where the OS EZ template is to be run:

```
%osarch
x86_64
```

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

* `x86`: specify if you plan to run the template on x86 platforms.

* `x86_64`: specify if you plan to use the template on x86-64 platforms.

As the template being created is intended for use on x86-64-bit platforms, the `%osarch` parameter is set to `x86_64`.

d **%packages**. Indicate the packages to include in the template.

The names of the packages must correspond to the names of real packages (with or without package versions, for example, `wget` or `wget=1.9.1`) that are stored in the repository used for the OS template (in our case, for CentOS 5). The specified packages will be downloaded from the package repository and installed on the server when you will cache the OS template.

> **Note:** If you are creating a template for Debian-based distributions (e.g., Debian, Ubuntu), you need to specify in this step three parameters: `%packages_0`, `%packages_1`, and `%packages`. This is explained by the fact that the installation of such distributions is carried out in three stages. If you do not know which packages to include, consult the documentation for the respective distribution.

e **%package_manager**. Specify the package manager to use for handling the template.

```
%package_manager
rpm47x64
```

For the full list of package managers, see **Available Package Managers** below.

f **%repositories**. Define the list of repositories storing packages for the template:

```
%repositories
$CE_SERVER/centos/6/os/x86_64
$CE_SERVER/centos/6/updates/x86_64
```

All the listed parameters are mandatory and must be specified in any metafile. You can also set a number of supplementary parameters in your metafile (e.g., `version` and `release`). For detailed information on all available parameters, see the `vzpkg.metafile` manual pages. For metafile samples, go to the `/usr/share/vztt/samples` directory and look for the `metafile` file in the subdirectory corresponding to specific Linux distributions (e.g., `/usr/share/vztt/samples/centos-6-x86_64/metafile`).

3 Save the file.

Metafiles for Application Templates

The example above explains in detail how to create a metafile for an OS template. The process of creating a metafile for an application template is similar to that described above, except for the following:

- Add the `%appname` parameter to the file to specify a name for the application template, for example:

```
%appname
php-customized
```

- Do not include the `%package_manager`, `%upgradable_versions`, and `%distribution` parameters in the file. These parameters are allowed for OS templates only.
- Make sure that the value of the `%osname` parameter corresponds to the name of the directory of the Linux distribution under which the application template is to be run. For example, if you are creating an application template for CentOS 6 installed in the `/vz/template/centos` directory, set this value to `centos`:

```
%osname
centos
```

- Ensure that the value of the `%osver` parameter corresponds to the version of the respective Linux distribution. For example, for CentOS 6 installed in the `/vz/template/centos/6` directory, set this value to 6:

```
%osver
6
```

Available Package Managers

Depending on the Linux distribution for which you are creating the template, set the following values for the `package_manager` parameter:

x86 Linux distributions

- `rpm49db5x86`: Fedora 17
- `rpm49x86`: Fedora 15 and 16
- `rpm47x86`: Red Hat Enterprise Linux 6 and CentOS 6
- `rpm44x86`: Red Hat Enterprise Linux 5 and CentOS 5
- `rpm43x86`: Red Hat Enterprise Linux 3 and 4, CentOS 3 and 4
- `rpmzypp44x86`: SUSE Linux Enterprise Server 11 with Service Pack 2
- `rpm41x86`: SUSE Linux Enterprise Server 10 and SUSE Linux 10.*x*
- `rpm41s9x86`: SUSE Linux Enterprise Server 9
- `rpmzypp49x86`: openSUSE 12.1
- `dpkg`: Debian and Ubuntu

x86-64 Linux distributions

- `rpm49db5x64`: Fedora 17
- `rpm49x64`: Fedora 15 and 16
- `rpm47x64`: Red Hat Enterprise Linux 6 and CentOS 6
- `rpm44x64`: Red Hat Enterprise Linux 5 and CentOS 5
- `rpm43x64`: Red Hat Enterprise Linux 3 and 4, CentOS 3 and 4
- `rpmzypp44x64`: SUSE Linux Enterprise Server 11 with Service Pack 2

- `rpm41x64`: SUSE Linux Enterprise Server 10 and SUSE Linux 10.*x*
- `rpm41s9x64`: SUSE Linux Enterprise Server 9
- `rpmzypp49x64`: openSUSE 12.1
- `dpkgx64`: Debian and Ubuntu

Creating the Template with vzmktmpl

In the last step, you use the `vzmktmpl` utility to create the template. To make a template with this utility, you need to specify the path to

- Metafile that defines all parameters for the template; see **Creating Metafiles** (p. 17) for details on metafiles.
- Scripts that are to be executed at different stages of the template life cycle (e.g., when caching an OS template or adding an application template to a Container). Each template should have its own set of scripts optimized for use with the Linux distribution for which the template is created. To learn what scripts to specify for your template:
  1. Install an official template provided by Virtuozzo for the same Linux distribution for which you are creating the template. For example, if you are making a template for CentOS 6, install the CenOS 6 OS template (use the `yum install` command, if necessary).
  2. Go to the configuration directory (`/vz/template/`*`os_name/os_version/arch`*`/config/os/default`) of the installed template.
  3. Look for the files starting with "`pre-`" and "`post-`" (e.g, `pre-cache` or `post-install`):

```
# ls -l /vz/template/centos/6/x86_64/config/os/default
...
-rwxr-xr-x 1 root root  453 Aug 13 22:12 post-cache
-rwxr-xr-x 1 root root 3399 Aug 13 22:12 post-install
-rwxr-xr-x 1 root root  610 Aug 13 22:12 pre-cache
-rwxr-xr-x 1 root root  943 Aug 13 22:12 pre-upgrade
...
```

     For the full list of scripts, see the **vzmktmpl** section in the *Virtuozzo 6 Command Line Reference Guide*.

Once you have a metafile and the required scripts at hand, copy them to your current working directory and execute the `vzmktmpl` utility. For example, to create a template for CentOS 6, base it on the `centos-6-custom.metafile` file, and include the scripts `post-cache`, `post-install`, `pre-cache`, and `pre-upgrade` in the template, you can run this command:

```
# vzmktmpl --pre-cache pre-cache --pre-upgrade pre-upgrade --post-cache post-cache --
post-install post-install centos-6-custom.metafile
```

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

```
# ls
centos-custom-6-x86_64-ez-1.0-1.noarch.rpm
...
```

To start using the new OS template, install it on the server using the `vzpkg install template` command and cache it with the `vzpkg create cache` command.

Setting Up Repositories and Proxy Servers for EZ Templates

If you are going to use OS and application EZ templates in Containers, you first need to have one or more repositories with software packages prepared for these 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 Virtuozzo, you can make use of the following package repositories:

- Default repositories. When you install an EZ template on the server, it is pre-configured to use official vendor's file sources (e.g., RPMs from the Fedora website) and a number of packages provided by Virtuozzo 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.
- Repositories for storing software packages and updates for Red Hat Enterprise Linux distributions.

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.

Managing Default Repositories

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 your system must meet is to have an active Internet connection to access the servers where the preconfigured repositories are located.

Let us assume that you want to use the 32-bit version of Fedora 16 to base your Containers on. To do this, you install the `fedora-core-16-x86-tmpl-4.0.0-17.swsoft.noarch.rpm` EZ template on the server:

```
# vzpkg install template fedora-core-16-x86-tmpl-4.0.0-17.swsoft.noarch.rpm
Preparing...                ########################################### [100%]
   1:fedora-core-16-x86-tmpl ########################################### [100%]
```

After the OS EZ template has been successfully installed, you can explore the default repositories for Fedora 16 in the `/vz/template/fedora/16/x86/config/os/default/mirrorlist` file. For example:

```
# cat /vz/template/fedora/16/x86/config/os/default/mirrorlist
$SW_SERVER/download/mirrors/fedora-core-16
$SW_SERVER/download/mirrors/updates-released-fc16
http://mirrors.fedoraproject.org/mirrorlist?repo=fedora-16&arch=i386
http://mirrors.fedoraproject.org/mirrorlist?repo=updates-released-f16&arch=i386
```

This file lists the repositories set to handle the Fedora 16 OS EZ template:

- Repositories on the official Web server are defined in the first two lines. `$SW_SERVER` denotes the string whose value is specified in the `/vz/template/conf/vztt/url.map` file. In our case, this string is the URL of the official Web server. The repositories on the official Web server keep a number of software packages needed for the correct operation of the `fedora-core-16-x86` template. Along with `$SW_SERVER`, the `url.map` file contains the list of URLs for all Linux distributions supported by Virtuozzo and having official repositories.
- Repositories on the Fedora web server are defined in the third and fourth lines. These repositories store all RPM packages for the Fedora 16 distribution.

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 official Web server are 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 for specific OS templates.

2. Commercial Linux distributions (e.g., Red Hat Enterprise Linux) do not have official repositories. You need to manually set up software repositories for such distributions before you can start using them. For more information, see to **Setting Up Repositories with vzcreaterepo** (p. 29).

Software packages are downloaded and installed on the server from package repositories when you:

- Cache an OS EZ template.
- Update an existing OS EZ template cache (provided that new packages are available in the repository).
- Add an application EZ template or package to a Container for the first time.

- Update an EZ template or software package in a Container.

You can set up your own repositories (e.g., storing unofficial software packages) for EZ templates. To do this, 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 desired repository. For example, to add the extra repository at http://mirrors.dotsrc.org/jpackage/ and keeping Java packages for Fedora 16:

1. Create the `/vz/template/fedora/16/x86/config/os/default/repositories` file on the server.
2. Add the following string to the file:

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

3. Save the file.

Creating a Local Repository

Along with using pre-configured repositories, you can create local repositories and configure for handling packages for your EZ templates. Organizing your own local repositories results in less bandwidth consumption and rapid software updates to Containers. You may also want to build a local repository if OS vendors or third-party software developers set up repositories that are not compatible with the `vzpkg` tool.

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

The process of setting up a local repository includes the following main steps:

1. 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.
2. 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.
3. 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 website 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 third one can be used within your local server only.

Example 1: Creating a Local Repository for Fedora 16

Let us assume that you want to build a local package repository for the Fedora 16 OS EZ template where the RPM packages for Fedora 16 will be downloaded and stored. The repository will be used by servers from both your local and other networks, and it will allow 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 16, do the following:

1 Install the `fedora-core-16-x86` OS EZ template (if it is not already installed):

```
# vzpkg install template fedora-core-16-x86-tmpl-4.0.0-17.swsoft.noarch.rpm
Preparing...                ########################################### [100%]
   1:fedora-core-16-x86-tmpl ########################################### [100%]
```

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 16 will be stored:

```
# mkdir -p download/fedora-core/16/i386/os/Fedora/RPMS
```

- The subdirectory where the updated versions of RPM packages for Fedora 16 will be stored:

```
# mkdir -p download/fedora-core/updates/16/i386
```

3 Copy all the packages comprising the Fedora 16 distribution (e.g., from your Fedora distribution disks) to the `download/fedora/16/i386/os/Fedora/RPMS` directory on the server.

4 Get the updates for Fedora 16, and put them to the `download/fedora-core/updates/16/i386` directory on the server.

5 Install the `createrepo` package on the server, for example:

```
# 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 16 base RPM packages:

```
# createrepo download/fedora/16/i386/os
```

- For the updated versions of the Fedora 16 RPM packages:

```
# createrepo download/fedora-core/updates/16/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, the lists will be kept 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 this as follows:

- For the Fedora 16 base RPM packages:

```
# echo 'http://123.145.145.123/download/fedora/16/i386/os/' >> download/mirrors/fedora-core-16
```

- For the updated versions of the Fedora 16 RPM packages:

```
# echo 'http://123.145.145.123/download/fedora/updates/16/i386' >> download/mirrors/updates-released-fc16
```

The aforementioned commands create the `fedora-core-16` and `updates-released-fc16` 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/16/i386/os/Fedora/RPMS` and `http://123.145.145.123/download/fedora/updates/16/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:

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

Example 2: Accessing the Repository via the `file` protocol

Once you set up a local repository by following the instructions in the example above, you can start using the `vzpkg` tool to obtain RPM packages for the Fedora 16 EZ template from this repository. You can connect to these repositories via the `http` protocol from both remote and local servers.

You can further speed up the process of distributing RPM packages across your repository for those Containers that reside on your local server (that is, on the server where the repositories are stored). To do this, configure your system to use the `file` protocol instead of `http` to connect to your repositories:

1 Open the `/vz/template/fedora/16/x86/config/os/default/mirrorlist` file on the server and comment the strings containing `$FC_SERVER`:

```
#$FC_SERVER/download/mirrors/fedora-core-16
#$FC_SERVER/download/mirrors/updates-released-fc16
```

2 Execute the following commands to create the repository files:

- To create the `/vz/template/fedora/16/x86/config/os/default/repositories` file on the server and to make it point to the Fedora 16 base RPM packages from your local repository:

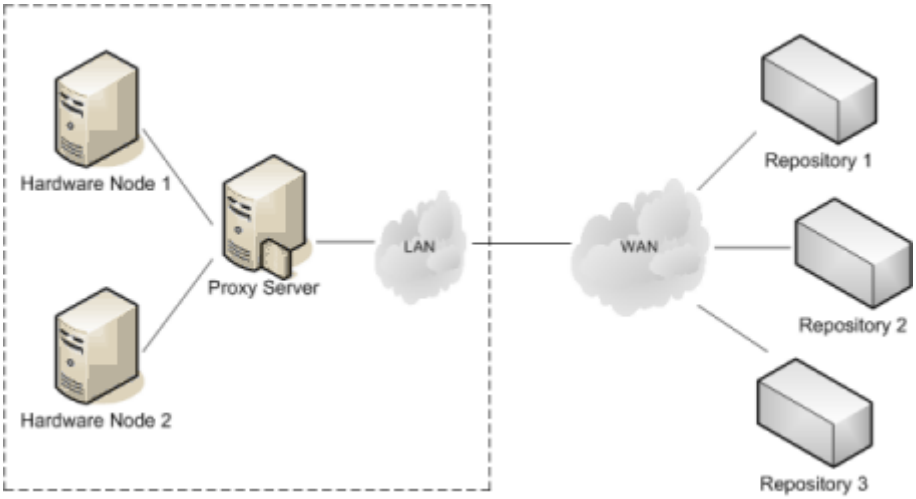
```
# echo 'file:///vz/root/101/var/www/html/download/fedora/16/i386/os/Fedora/RPMS' >>
/vz/template/fedora/16/x86/config/os/default/repositories
```

- To create the `/vz/template/fedora/16/x86/config/os/default/repositories` file on the server and to make it point to the updated versions of the Fedora 16 RPM packages from your local repository:

```
# echo 'file:///vz/root/101/var/www/html/download/fedora/updates/16/i386' >>
/vz/template/fedora/16/x86/config/os/default/repositories
```

Setting Up a Proxy Server for EZ Templates

You can set up special caching proxy servers and use them to efficiently manage your OS and application EZ templates. The following figure illustrates an example network configuration that contains two servers and a separate proxy server:



In this configuration:

- The caching proxy server sits between the servers (*Hardware Node 1* and *Hardware Node 2*) and keeps 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 downloaded to the corresponding server.

- When a server requests a package that is already available in the cache on the proxy server and none of the repositories has updates for the package, this package is immediately downloaded to the server from the proxy server. If any updates are available in the official repositories, the procedure of handling these updates is identical to that of the main packages: The update is downloaded to the proxy server where it is cached and then retransmitted to the corresponding server.
- A special script is automatically run on the proxy server at set intervals. During its execution, the script creates a local repository from the cached packages. All you have to do is to configure the `vzpkg` tool to use packages from this repository instead of obtaining them from remote repositories.

Setting up a proxy server for managing EZ templates has the following advantages:

1. Your Internet bandwidth consumption is greatly reduced because all packages are downloaded to the proxy server only once and can then be used by any server on your network.
2. You can more rapidly apply software updates to Containers since the proxy server where the downloaded packages are stored resides in the local network.
3. You can always have the software packages included in OS EZ templates at hand and do not have to worry whether they have been changed in or removed from their original repositories. The Virtuozzo script running on the proxy sever does not remove any cached packages; so it is up to you when to delete a specific package, if at all.

Setting Up a Proxy Server

To make a server act as a caching proxy server for EZ templates, you need to install the `vzpkgproxy` package on this server. `vzpkgproxy` is located in the `/virtuozzo/RPMS` directory of your Virtuozzo distribution and can be installed with the `rpm -i` command on servers (including Containers) meeting the following requirements:

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

Keep in mind that you may also need to install a number of additional packages to satisfy the `vzpkgproxy` dependencies.

By default, the caching proxy server does not allow connections from external hosts. To allow external connections, you need to edit the `/etc/httpd/conf.d/vzproxy.conf` file as described below:

1. Locate the following section in the `vzproxy.conf` file.

```
<Proxy *>
        Order deny,allow
        Deny from all
        # Enter all clients here
        Allow from localhost.localdomain
</Proxy>
```

2 Replace `localhost.localdomain` with the hostnames of external servers you want to be able to connect to the proxy server. For example, to allow connections from the `myserver1.com` and `myserver2.com` servers, you can edit the section as follows:

```
<Proxy *>
        Order deny,allow
        Deny from all
        # Enter all clients here
        Allow from myserver1.com myserver2.com
</Proxy>
```

You can also remove the string `Deny from all` and change the string `Allow from localhost.localdomain` to `Allow from all` to allow all external hosts to connect to the proxy server.

Once you edit the `/etc/httpd/conf.d/vzproxy.conf` file, you need to tell the `vzpkg` tool to use the packages from the local repository on the proxy server while handling EZ templates. To do this, edit the `/etc/vztt/vztt.conf` file on the server and specify the following parameters:

- 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.

Configuring Proxy Server Parameters

You can configure a number of proxy server parameters to meet your demands:

- Configure the `CACHE_DISABLE` parameter in the `/etc/vzpkgpoxy/vzpkgproxy.conf` file on the proxy server to specify the hosts to be excluded from the caching process. By default, the proxy server caches the packages from all external hosts that are allowed to connect to the proxy server.
- Configure the port number on which the `httpd` daemon running on the proxy server is listening. To do this, specify the desired port number in the `/etc/httpd/conf.d/vzproxy.conf` file on the proxy server. The default port number is 8080.
- Configure 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 is `/var/www/html/download`.

Setting Up Repositories with vzcreaterepo

Virtuozzo 6 provides you with a special utility, `vzcreaterepo`, to facilitate the process of setting up repositories for RHEL-like distributions, including repositories for Red Hat Enterprise Linux 5.

Note: The current version of `vzcreaterepo` does not support creating repositories for Red Hat Enterprise Linux 6 and all versions of openSUSE and SUSE Linux Enterprise Server.

The process of setting up a repository with `vzcreaterepo` includes the following steps:

1. Installing an OS template for which you plan to set up the repository.
2. Obtaining the Linux distribution files to use as the basis for creating the repository.

   Note: Always try to get the most recent versions of Linux distributions. For example, when creating a repository for Red Hat Enterprise Linux 5 and choosing between two versions 5.7 or 5.8, go with version 5.8. This will help you avoid possible problems with `vzcreaterepo` that is optimized for work with the latest versions of Linux distributions.
3. Creating the repository with the `vzcreaterepo` utility from the prepared distribution files.
4. Configuring the `vzpkg` template management tool to contact the created repository. As a rule, `vzcreaterepo` automatically configures the `vzpkg` tool to work with newly created repositories. If some extra configuration is required, the utility will inform you of the steps to complete.

The example below assumes the following:

- You want to create a repository for the x64 version of Red Hat Enterprise Linux 5.8.
- The `redhat-el5-x86_64` OS template is installed on your server.
- The full path to the RHEL 5.8 ISO image on the server is `/mnt/iso/rhel-server-5.8-x86_64-dvd.iso`.

To set up a repository, you can run this command:

```
# vzcreaterepo -t redhat-el5-x86_64 -i /home/rhel-server-5.8-x86_64-dvd.iso
```

By default, `vzcreaterepo` sets up the repository in the `/vz/template/repos/download/redhat/el5/x86_64` directory. If necessary, you can specify another location for your repository using the `-d` option.

To configure your repository to also store package updates for RHEL 5.8, you need to specify a number of additional options when executing the `vzcreaterepo` command, for example:

```
# vzcreaterepo -t redhat-el5-x86_64 -i /home/rhel-server-5.8-x86_64-dvd.iso --ctid 101
--ip 192.168.98.198 --hostname myrhelupdates.com --nameserver 192.168.0.1 -e
user@mycompany.com
```

Along with setting up a repository for RHEL 5.8 in the `/vz/template/repos/download/redhat/el5/x86_64` directory, this command:

- Creates Container 101 with the IP address of 192.168.98.198 and the hostname of `myrhelupdates.com`.
- Configures for the Container the nameserver with the IP address of 192.168.0.1.
- Asks you for the user name and password to log in to the Red Hat Network (RHN). Use the credentials you provided when registering in RHN.
- Registers the Container in the Red Hat Network with the email address `user@mycompany.com`. Replace `user@mycompany.com` with the email address you specified when setting up your account in the Red Hat Network.
- Downloads updated packages from the RHEL web site to the Container and creates an update repository from these packages.

Notes:

1. An active Internet connection is required to connect to the Red Hat website and register in the Red Hat Network.

2. You must have an active RHN account before running the `vzcreaterepo` command to create a repository with package updates. The account must allow you to download updates for the system for which you are setting up the repository.

Preparing an OS EZ Template for Container Creation

Although you can create Containers without having to prepare OS EZ templates first, doing so provides more control over the entire process and helps avoid problems which might be caused by a slow Internet connection, for example. OS EZ template preparation includes:

1. Installing an OS EZ template on the server.
2. Creating an OS EZ template cache, with or without preinstalled applications.

For details on how to start creating Containers once caching has been done, see the *Virtuozzo 6 User's Guide*.

Installing OS EZ Templates

To install a new OS EZ template on the server, use the `vzpkg install template` command. For example, to install the CentOS 6 EZ (x86) template, you can do this:

```
# vzpkg install template centos-6-x86
Preparing...                ########################################### [100%]
   1:centos-6-x86           ########################################### [100%]
```

You may also want to check that the template has been installed successfully:

```
# vzpkg list
centos-6-x86
```

The `centos-6-x86` EZ template has been installed on the server in the `/vz/template/centos/6` directory.

Note: Before caching OS EZ templates, you may need to set up a local package repository (e.g., for commercial distributions like Red Hat Enterprise Linux). For details on how to manage package repositories, see **Setting Up Repositories and Proxy Servers for EZ Templates** (p. 21).

Creating Clean OS EZ Template Caches

You can use a clean OS EZ template cache to create an empty Container without any custom applications installed. To create a clean OS EZ template cache, use the `vzpkg create cache` command, specifying an OS EZ template to cache:

```
# vzpkg create cache centos-6-x86
```

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

```
# ls /vz/template/cache
centos-6-x86.plain.ploop.tar.lzrw
```

Creating OS EZ Template Caches with Preinstalled Application Templates

You can use an OS EZ template cache with preinstalled application templates to quickly create multiple identical Containers without having to install applications manually or wait until they are installed automatically to each Container after its creation. The best way to create such a cache is:

1. Make a custom sample configuration file with information on the OS EZ template to cache and application EZ templates to preinstall. For example:

```
# cp /etc/vz/conf/ve-basic.conf-sample /etc/vz/conf/ve-centos-6-x86-mysql-devel.conf-sample
```

   Note: If you already have a custom sample configuration file created in Parallels Server Bare Metal 5 or earlier and having application EZ templates specified in it, you can reuse it instead of creating a new one.

2. Add the OS EZ template and application EZ template information to the new configuration file. Each OS and application template name must be preceded by a dot. Multiple consecutive application EZ template names must be separated by white spaces. For example:

```
# cd /etc/vz/conf
# echo 'OSTEMPLATE=".centos-6-x86"' >> ve-centos-6-x86-mysql-devel.conf-sample
# echo 'TEMPLATES=".mysql .devel"' >> ve-centos-6-x86-mysql-devel.conf-sample
```

3. Run the `vzpkg create appcache` command with your configuration file as an option. For example:

```
# vzpkg create appcache --config centos-6-x86-mysql-devel
```

   Note: If the resulting cache already exists, it will not be recreated and you will see a corresponding message.

The resulting archive can be found in the `/vz/template/cache` directory on the Hardware Node. You can check that it exists and includes necessary application templates with the following command:

```
# vzpkg list appcache
centos-6-x86                 2012-07-20 16:51:36
     mysql
     devel
```

Installing Application EZ Templates on the Server

The same way as you use an OS EZ template in a Virtuozzo system to create any number of Containers on its basis and share its resources, you can use application EZ templates to share application files among any number of Containers.

To install a new application EZ template on the server, you can use the `vzpkg install template` command. For example, to install the `mysql` EZ template intended for use on the CentOS 5 distribution, you can execute the following command:

```
# vzpkg install template mysql-centos-6-x86
Preparing...                ########################################### [100%]
   1:mysql-centos-6-x86     ########################################### [100%]
```

The `mysql` EZ template is installed into the `/vz/template/centos/6/x86/config/app/mysql` directory on the server. To check that the installation was successful, you can use the `vzpkg list` command:

```
# vzpkg list
centos-6-x86                  2012-07-21 02:22:45
centos-6-x86  mysql
```

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

Viewing Installed EZ Templates on the Server

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 -O
centos-6-x86
centos-6-x86_64                    2012-07-10 13:16:43
```

Using the `-O` option with the `vzpkg list` command, you can list only the OS EZ templates installed on the server. The time next to an OS EZ template indicates when the template was cached.

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

```
# vzpkg list 101
centos-6-x86_64                    2012-07-10 13:16:43
```

Viewing Available EZ Templates on the Server

The `vzpkg list --available` command allows you to list the EZ templates that are available in the official remote repositories but not installed on the server yet. For example, to list all EZ templates available for installation on your server, you can run this command:

```
# vzpkg list --available
```

The `vzpkg list` command also supports using wildcards. So you can execute the following command to view all MySQL application templates intended for use with different versions of CentOS:

```
# vzpkg list --available mysql-centos*
mysql-centos-4-x86                pvc-templates-32
mysql-centos-4-x86_64             pvc-templates
mysql-centos-5-x86                pvc-templates-32
mysql-centos-5-x86_64             pvc-templates
mysql-centos-6-x86                pvc-templates-32
mysql-centos-6-x86_64             pvc-templates
```

Keeping EZ Templates Up To Date

Virtuozzo allows you to update 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 specific Containers.

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

Updating EZ Templates on the Server

Sometimes, you may need to update one or more EZ templates (either OS or application) installed on your server. The process of updating an EZ template includes updating its configuration files in the `/vz/template/<os_name>/<os_version>/<arch>/config` directory on the server.

The recommended way of updating an installed EZ template is to use the `vzpkg update template` command. When executed, this command tells the `vzpkg` utility to access the official remote repositories and check for available OS and application EZ templates updates. For example, you can run the following command to update the CentOS 5 OS EZ template installed on the server:

```
# vzpkg update template centos-6-x86
```

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 centos-6-x86 redhat-el6-x86
```

simultaneously updates the CentOS 6 and Red Hat Enterprise Linux 6 OS EZ templates installed on the server.

Updating OS EZ Template Caches

Virtuozzo has two types of OS EZ template caches: with and without preinstalled application templates. Each cache type can be updated with a corresponding command.

Note: 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 update the cache. For more information on how to manage repositories for commercial Linux distributions, see **Setting Up Repositories and Proxy Servers for EZ Templates** (p. 21).

Updating OS EZ Template Caches without Preinstalled Applications

If you need to update an OS EZ template cache without preinstalled application templates, use the `vzpkg update cache` command.

When executed, `vzpkg update cache` checks the template cache directory on the server (the default is `/vz/template/cache`) and updates all existing tarballs there. To update a particular cache, specify the corresponding OS EZ template name. For example, to update the cache of the `centos-6-x86` OS EZ template, run the following command:

```
# vzpkg update cache centos-6-x86
```

When executed, `vzpkg update cache` also appends the `-old` suffix to old tarball names (e.g., `centos-6-x86.plain.ploop.tar.lzrw-old`):

```
# ls /vz/template/cache
centos-6-x86.plain.ploop.tar.lzrw centos-6-x86.plain.ploop.tar.lzrw-old
```

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

If the `vzpkg update cache` command cannot find caches for one or more OS EZ templates installed on the server, it creates those in the `/vz/template/cache` directory.

Updating OS EZ Template Caches with Preinstalled Applications

If you need to update an OS EZ template cache with preinstalled application templates, use the `vzpkg update appcache` command with the `--update-cache` option. For example, to update a `centos-6-x86` OS EZ template cache with preinstalled `mysql`, `devel`, and `php`, run the following:

```
# vzpkg update appcache --ostemplate centos-6-x86 --apptemplate mysql,php,devel --
update-cache
```

Updating EZ Templates Packages Inside a Container

Virtuozzo 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. 21).

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.

Adding Application EZ Templates to Containers

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 host is added to Container 101:

```
# prlctl status 101
CTID 101 exists mounted running
# vzpkg list
redhat-el6-x86             2012-05-21 02:21:56
redhat-el6-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
...
```

Copying EZ Templates to Another Server

Virtuozzo 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 `centos-6-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 centos-6-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 `centos-6-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 `centos-6-x86` configuration files from the source server to the destination server. Please keep in mind that the `centos-6-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. 30).

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 `centos-6-x86` OS template has been successfully copied to the destination server, you can run the following command on this server:

```
# vzpkg list
centos-6-x86                20012-6-12 07:05:39
```

Removing an Application from Containers

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-el6-x86                  2012-05-21 02:21:56
redhat-el6-x86   mysql          2012-05-21 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-el6-x86                  2012-05-21 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 Templates From the Server

The `vzpkg remove template` command allows you to remove from the host 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 host:

```
# vzpkg list
redhat-el6-x86              2012-06-16 12:50:17
centos-6-x86                2012-06-18 14:23:12
# vzpkg remove template redhat-el6-x86
redhat-el6-x86 template was removed
# vzpkg list
centos-6-x86                2012-06-18 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 6 from your server:

```
# vzpkg remove template -F redhat-el6-x86 mailman
redhat-el6-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-el6-x86 sitebuilder4 mailman
redhat-el6-x86 sitebuilder4 template was removed
redhat-el6-x86 mailman template was removed
```

In this example the `sitebuilder4` and `mailman` EZ application templates intended to run under RHEL 6 have been successfully removed from the host.

CHAPTER 4

Glossary

This glossary defines terms and spells out abbreviations used in Virtuozzo documentation. References to terms defined elsewhere in the glossary appear in italics.

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, users database, configuration files, applications, system libraries, and so on. Containers share one host 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 or Linux operating systems.

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

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

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

Package set. See *Template*.

Host (physical server or server). A server where the Virtuozzo software is installed for hosting Virtuozzo virtual machines and Containers. Sometimes, it is marked as Container 0.

Virtuozzo license key. A license key that you should install on the physical server to use Virtuozzo. Every physical server must have its own key installed.

Virtuozzo Storage license key. A license key for Virtuozzo Storage to use its functionality.

Memory and IOPS deduplication. A feature introduced in Virtuozzo 6.0. By caching identical files in multiple Containers it helps save memory and IOPS on the Hardware Node.

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 (package set). A set of original application files (packages) repackaged for mounting. 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 Virtuozzo software for managing *Container* memory and some system-related resources.

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

Virtuozzo File System (VZFS). A virtual file system for legacy Containers.

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

Index