Parallels

Parallels Server 4 Bare Metal

Command Line Reference 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

Introduction

In This Chapter

About Parallels Server 4 Bare Metal

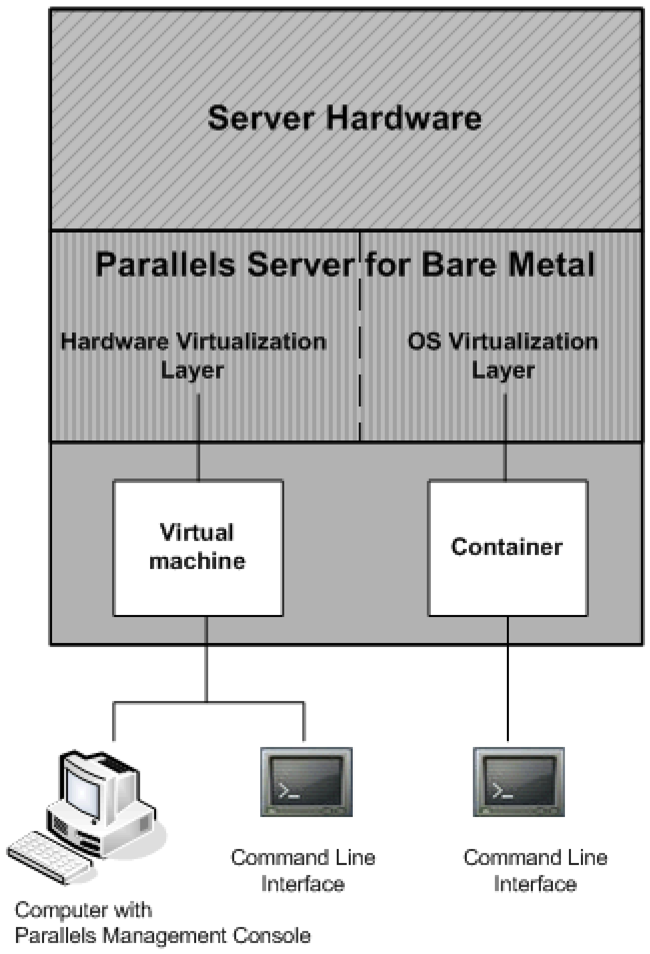
Parallels Server 4 Bare Metal provides you with the possibility to simultaneously run Parallels virtual machines and Containers on the same physical server. Using this software, you can efficiently use your server's hardware resources by sharing them among multiple virtual machines and Containers.

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 physical 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 a complete reference on all Parallels Server Bare Metal configuration files and command-line utilities. It familiarizes you with the way to configure Parallels Server Bare Metal to meet your requirements and to perform various tasks by using the corresponding command-line utilities.

The primary audience for this guide is anyone who is looking for an explanation of a particular configuration option, needs help for a particular command, or is seeking for a command to perform a certain task.

Organization of This Guide

Chapter 1, **Introduction**, gives an overview of the Parallels Server Bare Metal product and this guide.

Chapter 2, **Managing Parallels Server 4 Bare Metal**, provides instructions on Parallels Server Bare Metal configuration files, scripts, and command-line utilities.

Chapter 4, **Managing Containers**, describes Parallels Server Bare Metal command-line utilities that can be used for managing your Containers.

Chapter 5, **Managing Virtual Machines**, focuses on Parallels Server Bare Metal utilities that can used for managing your virtual machines.

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.

Formatting Legend

Format	Meaning
Bold	Parameters that the user must type exactly as shown.
Italic	Parameter values that the user must supply.
Between square brackets. Example: `[`**--name** *`name`*`]`	Optional parameters.

Between curly brackets and/or separated by pipe (|).
 	
Examples:
 	
`ID``|`*`name`*
 	
`{`**-o** *`name`*`|`**-d** *`name`*`}`
	Set of choices from which the user must choose only one.

Parameter followed by the same parameter in brackets with ellipses. Example: *`name`*`[,`*`name`*`...]`	Parameters that can be repeated more than once in the same command line.

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 Server 4 Templates Management 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 the physical server resources (physical memory, disk space, etc.).
- *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

Managing Parallels Server 4 Bare Metal

This chapter provides instructions on configuration files, scripts, and command-line utilities that can used to configure the settings related to the Parallels Server Bare Metal software and the Parallels server.

In This Chapter

Parallels Server Bare Metal Configuration Files

This section explains what configuration parameters Parallels Server 4 Bare Metal has and how they affect the product behavior.

There are a number of files responsible for the Parallels Server Bare Metal system configuration. Most of the files are located in the `/etc` directory on the Parallels server. However, some configuration files are stored on the Backup Node, inside a Container, or on a dedicated server. In case a configuration file is located in a place other than the Parallels server, we point clearly the exact position where it can be found.

A list of configuration files is presented in the table below:

`/etc/vz/vz.conf`	The Parallels Server Bare Metal global configuration file. This file keeps system-wide settings, affecting Container and template default location, global network settings, and so on.
`/etc/vz/conf/<CT_ID>.conf`	The private configuration file owned by a Container numbered `<CT_ID>`. The file keeps Container specific settings – its resource management parameters, location of private area, IP address, and so on.
`/etc/vz/conf/ve-<name>.conf.sample`	Sample files, containing a number of default Container configurations, which may be used as a reference for Container creation. The following samples are shipped with Parallels Server Bare Metal: `basic`, `cpanel`, `confixx`, `slm.plesk`, `slm.256MB`, `slm.512MB`, `slm.1024MB`, `slm.2048MB`. You may also create your new samples customized for your own needs.
`/etc/vz/conf/dists/<distribution_name>.conf`	The configuration files used to determine what scripts are to be run on performing some operations in the Container context (e.g. on adding a new IP address to the Container). These scripts are different from Parallels Server Bare Metal action scripts and depend on the Linux version the given Container is running.
`/etc/vz/conf/networks_classes`	The definition of network classes, used by traffic shaping and bandwidth management in Parallels Server Bare Metal.
`/etc/sysconfig/vzup2date/vzup2date.conf`	This file specifies the default connection parameters for the `vzup2date` utility.
`/<path>/<name>.conf`	This configuration file specifies the default connection parameters for the `vzup2date-mirror` utility. It should be located on the computer where you are planning to run `vzup2date-mirror`.

`/etc/cron.d/vereboot`	The configuration file for the `cron` daemon. Using this file, Parallels Server Bare Metal emulates the "`reboot`" command working inside a Container.
`/etc/vzvpn/vzvpn.conf`	The configuration file used to define the parameters for establishing a private secure channel to the Parallels support team server.
`/etc/vzreport.conf`	The configuration file used to define the parameters for sending your problem report to the Parallels support team.
`/etc/sysctl.conf`	Kernel parameters. Parallels Server Bare Metal adjusts a number of kernel `sysctl` parameters and modifies the default `/etc/sysctl.conf` file.
`/etc/vzredirect.d/*.conf`	These files define the offline management modes for controlling Containers by their administrators.
`/etc/vzlmond.conf`	This configuration file defines the parameters used by the `vzlmond` daemon to collect information on the main Parallels server resources consumption.
`/etc/vzstat.conf`	The file lists the warning and/or error levels for a number of resource control parameters. If a parameter hits the warning or error value, the `vzstat` utility will display this parameter in yellow or red.
`/etc/vzstatrep.conf`	This configuration file is located on the Monitor Node and used by the `vzstatrep` utility when generating statistic reports and graphics on the Parallels server resource consumption and sending these reports to the server administrator.
`/etc/vzbackup.conf`	The global configuration file residing on the Backup Node and determining the global virtual machines and Containers backup settings.
`/etc/vz/pkgproxy/rhn.conf`	The Red Hat Network (RHN) Proxy Server configuration file used by the `vzrhnproxy` utility when setting up the RHN Proxy Server. This file can be located on any computer where the `vzrhnproxy` package is installed.
`/etc/vzpkgpoxy/vzpkgproxy.conf`	This configuration file is used by the `vzpkgproxy` utility when creating special caching proxy servers for OS and application EZ templates. The file can be located on any computer where the `vzpkgproxy` package is installed.
`/etc/vztt/vztt.conf`	This configuration file is used by the `vzpkg` utility when managing OS and application EZ templates.

Global Parallels Server Bare Metal Configuration File

Parallels Server Bare Metal keeps its system wide configuration parameters in the `/etc/vz/vz.conf` configuration file. This file is in shell format. Keep in mind that Parallels Server Bare Metal scripts source this file – thus, shell commands in this file will cause system to execute them under root account. Parameters in this file are presented in the form `PARAMETER=”value”`. Logically all the parameters belong to the following groups: global parameters, logging, disk quota, template, network traffic, Containers, validation and overcommitment, supplementary parameters, and name-based hosting parameters. Below is the description of all the parameters defined in this version of Parallels Server Bare Metal.

Global parameters

Parameter	Description	Default value
`VIRTUOZZO`	This can be either “yes” or “no”. Parallels Server Bare Metal System V startup script checks this parameter. If set to “no”, then Parallels Server Bare Metal modules are not loaded. You might set it to “no” if you want to perform system maintenance and do not want to bring up all Containers on the server.	`yes`
`HTTP_PROXY`	Specifies either the hostname or the IP address of the HTTP proxy server. After setting this parameter and in case you use an HTTP proxy server for handling all HTTP requests, the Parallels Server Bare Metal utilities communicating with the outer world through HTTP (e.g. the `vzreport` utility) will use this server for managing all your HTTP messages (e.g. sending your problem report).	`-`
`ACTIONLOGDIR`	This is the directory where `pctl` keeps a log of its actions in the format suitable for Parallels Server Bare Metal statistics daemon `hwcoll`.	`/vz/actionlog`
`LOCKDIR`	Actions on a Container should be serialized, since two simultaneous operations on the same Container may break its consistency. Parallels Server Bare Metal keeps lock files in this directory in order to serialize access to one Container.	`/vz/lock`
`REMOVEMIGRATED`	Specifies whether the private area and the configuration file of the Container moved to a new server with the `vzmigrate` command should be destroyed on the Source Server (the value of the parameter is set to `yes`) or renamed to have the `.migrated` suffix (the value of the parameter is set to `no`). You may wish to leave the Container private area and the configuration file to make migration faster. This configuration value can be overridden by the `vzmigrate` command-line options.	`no`

VE0CPUUNITS	CPU weight designated for the server itself.	1000
OFFLINE_MANAGEMENT	Specifies whether Containers can be managed by the Container administrator by means of the services indicated in the OFFLINE_SERVICE parameter.	yes
OFFLINE_SERVICE	These services correspond to the names of the files in the /etc/vzredirect.d directory, each file defining at what port the service will be accessible and to what Container the requests coming to this port will be redirected. These services will be accessible to those Containers which have the OFFLINE_MANAGEMENT parameter set to "yes".	vzpp-plesk vzpp
BURST_CPU_AVG_USAGE	The CPU usage limit, in percent, set for the Container. This limit is calculated as the ratio of the current Container CPU usage to the CPU limit (i.e to the value of the CPULIMIT parameter) set for the Container in its configuration file. If the limit is not specified, the full CPU power of the server is considered as the CPU limit. Upon exceeding the BURST_CPU_AVG_USAGE limit, the BURST_CPULIMIT limit is applied to the given Container. This parameter can be redefined by the BURST_CPU_AVG_USAGE parameter set in the Container configuration file.	disabled
BURST_CPULIMIT	The CPU power limit, in per cent, the Container cannot exceed. The limitations set in this parameter are applied to any Container exceeding the limit specified in the BURST_CPU_AVG_USAGE parameter. This parameter can be redefined by the BURST_CPULIMIT parameter set in the Container configuration file.	
VEFORMAT	Determines the VZFS version to be applied to all Containers that will be created on the given server: ▪ If you wish your Containers to use the benefits of the VZFS v2 technology, the value of this parameter should be set to vz4. ▪ If you wish your Containers to be based on VZFS v1, you should make sure that the value of this parameter is set to vz3.	vz4
VZMOUNTS	Defines the partitions which will be automatically mounted by the /etc/init.d/vz script after the server boot. This script will check (by calling the fsck utility) and mount all the partitions specified as the value of this parameter, listed	/vz

in /etc/fstab file on the server, and having the noauto flag set for them in this file.

Logging parameters affect the pctl utility logging behavior.

Parameter	Description	Default value
LOGGING	This parameter defines whether pctl should log its actions.	yes
LOGFILE	File where pctl logs its actions.	/var/log/vzctl.log
LOG_LEVEL	There are three levels of logging defined in the current version of Parallels Server Bare Metal.	0

The table below describes the possible values of the LOG_LEVEL parameter and their meanings:

Log level	Information to be logged
0	Actions of pctl on Containers like start, stop, create, destroy, mount, umount.
1	Level 1 logs events, calls to pctl helper scripts located in /etc/vz/conf (such as vz-start and vz-stop) and situations when the init process of the Container is killed on Container stop after timeout.
2	Level 0 and level 1 logging events, plus template version used for Container creation and calls to mount and quota operations with parameters.

Disk quota parameters allow you to control the disk usage by the Containers:

Parameter	Description	Default value
DISK_QUOTA	DISK_QUOTA defines whether to turn on disk quota for Containers. If set to “no” then disk space and inodes accounting will be disabled.	yes
VZFASTBOOT	This option determines the Container quota reinitialization procedure when the server is booted after an incorrect shutdown. If set to "no", the disk quota is reinitialized for each Container during the server startup and only then are the Containers started, which results in a long server and Containers booting time. When set to "yes", the Container quota reinitialization procedure depends on the Container quota files state: ▪ Those Containers whose quota files (/var/vzquota/quota.<CT_*ID*>) have a "dirty" flag set, meaning that their contents are inconsistent with the real Containers usage, are started without the quota reinitialization. After all the Containers with "dirty" flags are launched, they are restarted one by one to reinitialize their respective quotas. ▪ Those Containers whose quota files are absent from the server or corrupted are started only after their quota has been successfully reinitialized.	no

In general, setting the `VZFASTBOOT` parameter to "yes" allows you to considerably reduce the server and Containers downtime after the incorrect server shutdown.

SLM parameters allow you to control the amount of memory consumed by the Containers:

Note: The SLM parameters are supported only in the Linux distributions running the 2.6 kernel.

Parameter	Description	Default value
`SLM`	If set to "yes", the SLM modules are loaded to the server. It means that the `slmmemorylimit` parameter is supported and can be used to manage the amount of memory consumed by every Container on the server. **Note:** After changing this parameter, restart the Parallels Server Bare Metal service for the changes to take effect.	`yes`
`SLMPATTERN`	Defines the SLM pattern rules for grouping the processes running inside Containers on the server. The default rules are set in the `/etc/vzslm.d/default.conf` file on the server.	`default`

Network traffic parameters define whether you want to account bandwidth consumed by Containers and whether you want to limit bandwidth available to Containers:

Parameter	Description	Default value
`TRAFFIC_SHAPING`	Traffic shaping allows you to limit the bandwidth consumed by Containers for outgoing traffic. If it is set to “yes”, then limitations will be turned on. If you want to use this feature, `TRAFFIC_ACCOUNTING` should be set to “yes” as well.	`no`
`BANDWIDTH`	This is the list of network interfaces on which we want to shape the traffic and their speed in the form of “dev:rate”. The rate is measured in Kbits/s. If you want to shape traffic on more than one interface, set this parameter to “dev1:rate1 dev2:rate2”. For example, for two 100 Mbits/s Ethernet cards, set it to “eth0:102400 eth1:102400”.	`eth0:102400`
`TOTALRATE`	This parameter sets the size of the bandwidth pool for all Containers. It is the upper limit for the bandwidth available to all your Containers and is specified in the form of “dev:class:rate”. The rate is measured in Kbits/s. Containers can consume bandwidth up to this limit in addition to the limit specified by the `RATE` parameter. Default value corresponds to 4 Mbits/s limit for the Class 1 Containers.	`eth0:1:4096`
`RATE`	This parameter is the default bandwidth guaranteed to a Container for outgoing traffic if the Container	`eth0:1:8`

configuration file does not explicitly specify a different value. This value is in the same format as TOTALRATE and its default value is “eth0:1:8”. The rate is measured in Kbits/s. Note that 8 Kbits/s, offered by the default configuration, is the guarantee and the Container cannot consume less than this value and more than the sum of this value and TOTALRATE.

Template parameters allow to configure the template area location.

Parameter	Description	Default value
TEMPLATE	This is the directory where to find templates. It is not recommended to redefine this option since all the templates built by Parallels use the default directory.	/vz/template

Container default parameters either affect new Container creation or represent Container parameters that can be overridden in the Container configuration file:

Parameter	Description	Default value
VE_ROOT	This is a path to the Container root directory where the private area is mounted.	/vz/root/CT_ID
VE_PRIVATE	This is a path to the Container private area, where VZFS keeps its private data. VZFS implementation requires VE_PRIVATE reside within a single physical partition.	/vz/private/CT_ID
CONFIGFILE	The default configuration file sample to be used for the Container creation; it may be overridden with the --config option of the pctl create command.	basic
DEF_OSTEMPLATE	The default OS template to be used for the Container creation; it may be overridden with the --pkgset command-line option for pctl create.	fedora-core-7
IPTABLES	Only those iptables modules will be loaded to the Containers hosted on the server which are indicated as the value of this parameter and only if they are loaded on the server itself as well.	ip_tables ipt_REJECT ipt_tos ipt_limit ipt_multiport iptable_filter iptable_mangle ipt_TCPMSS ipt_tcpmss ipt_ttl ipt_length
VE_ENVIRONMENT	Additional environment variables to be passed to the Container init process. Should be provided as any number of *name=value* pairs separated by spaces.	

Container validation and overcommitment parameters define whether the Container configuration should be validated and the server overcommitment checked on a Container startup:

Parameter	Description	Default value

VE_VALIDATE_ACTION	Defines whether the Container configuration should be validated when a Container is started. If this parameter is set to “warning”, a warning is displayed in case of misconfiguration. If set to “error”, the Container is not started in case of misconfiguration. If set to “fix”, the configuration is automatically corrected.	none
OVERCOMMITMENT_ACTION	Defines whether the server should be checked for the overcommitment of resources when a Container is started. If this parameter is set to “warning”, a warning is displayed in case of overcommitment. If set to “error”, the Container that would cause overcommitment is not started. When checking for overcommitment, the following five parameters are checked.	none
OVERCOMMITMENT_LEVEL_LOWMEM	The percentage of committed memory residing at lower addresses and directly accessed by the kernel.	120
OVERCOMMITMENT_LEVEL_MEMSWAP	The percentage of committed memory available for applications including both RAM and swap space.	90
OVERCOMMITMENT_LEVEL_ALLOCMEM	The allocation memory commitment level is the ratio of the memory size guaranteed to be available for allocation to the capacity of the system.	100
OVERCOMMITMENT_LEVEL_ALLOCMEM_TOT	The number shows how much memory the applications are allowed to allocate in comparison with the capacity of the system.	1000
OVERCOMMITMENT_LEVEL_ALLOCMEM_MAX	This allocation memory commitment level is the ratio of the maximal (among all	60

running Containers) amount of allocated memory to the capacity of the system.

Supplementary parameters define other Parallels Server Bare Metal settings:

Parameter	Description	Default value
`VZWDOG`	Defines whether the `vzwdog` module is loaded on Parallels Server Bare Metal startup. This module is responsible for catching messages from the kernel. It is needed if you configure the serial Monitor Server for Parallels Server Bare Metal.	`no`
`VZPRIVRANGE`	Defines the ID range for the Containers that are allowed to access the <servere> ID stored in the `/proc/vz/hwid` file.	`1 100`
`DUMPDIR`	The directory where the Container dump file created by means of the `pctl suspend` command is to be stored.	`/vz/private /CT_`*`ID`*`/dump`

Container Configuration File

Each Container has its own configuration file, which is stored in the `/etc/vz/conf` directory and has a name like *`CT_ID`*`.conf`. This file has the same format as the global configuration file. The settings specified in this file can be subdivided into the following categories: miscellaneous, networking, backup, resource management parameters, and name-based hosting parameters.

Note: In Parallels Server Bare Metal, you can also configure a number of settings for the server itself by editing the `/etc/vz/conf/0.conf` file. Currently, these settings include the `VERSION` and `ONBOOT` parameters, as well as all parameters listed in the table under the *System parameters* group.

Miscellaneous parameters:

`VERSION`	Specifies the Parallels Server Bare Metal version the configuration file applies to. "2" relates to Parallels Server Bare Metal version 4 and later.
`ONBOOT`	Specifies whether the Container should be started automatically on system startup. Parallels Server Bare Metal automatically starts all Containers that have this parameter set to “yes” upon startup. **Note:** If "yes" is specified as the value of this parameter in the `0.conf` file, all server system management parameters are set on the server boot to the values indicated in this file.
`OFFLINE_MANAGEMENT`	Overrides the `OFFLINE_MANAGEMENT` parameter from the global configuration file.
`OFFLINE_SERVICE`	Overrides the `OFFLINE_SERVICE` parameter from the global configuration file.
`ALLOWREBOOT`	Specifies whether the Container may be restarted with the “reboot” command inside. If omitted or set to “yes”, reboot is allowed. **Note:** To make reboot working, you should uncomment the corresponding line in the `/etc/cron.d/vereboot` file.
`CAPABILITY`	Specifies capabilities inside of the Container. Setting of following capabilities is allowed: `CHOWN`, `AC_OVERRIDE`, `AC_READ_SEARCH`, `FOWNER`, `FSETID`, `KILL`, `SETGID`, `SETUID`, `SETPCAP`, `LINUX_IMMUTABLE`, `NET_BIND_SERVICE`, `NET_BROADCAST`, `NET_ADMIN`, `NET_RAW`, `IPC_LOCK`, `IPC_OWNER`, `SYS_MODULE`, `SYS_RAWIO`, `SYS_CHROOT`, `SYS_PTRACE`, `SYS_PACCT`, `SYS_ADMIN`, `SYS_BOOT`, `SYS_NICE`, `SYS_RESOURCE`, `SYS_TIME`, `SYS_TTY_CONFIG`, `MKNOD`, `LEASE`.
`OSTEMPLATE`	The name of the OS template that was used for creating the Container. You do not have to change this parameter; `pctl` will set it for you upon calling the `pctl create` command (or using the defaults from the global configuration file). The `.` symbol before the OS template name, if specified, indicates that this is an EZ OS template.

`TEMPLATES`	When used in the Container sample configuration file, this parameter defines a list of application templates that should be automatically added to the Container being created on the basis of this sample. So, if the corresponding templates are installed on the server, and the `pctl create` command uses a configuration file with this parameter defined, the templates will be added to the Container immediately upon its creation. When used in the configuration file of an existing Container, this parameter provides a list of templates that have been installed inside the Container by means of either the `pctl create`, `vzpkgadd`, or `vzpkg install` commands. In this case you should not modify this parameter since it is used by template management utilities to track the history of the installed templates. This parameter is omitted if no templates have been applied to the Container.
`VE_ROOT`	Overrides the `VE_ROOT` parameter from the global configuration file.
`VE_PRIVATE`	Overrides the `VE_PRIVATE` parameter from the global configuration file.
`VE_ENVIRONMENT`	Overrides the `VE_ENVIRONMENT` parameter from the global configuration file.
`TECHNOLOGIES`	Determines a set of technologies which should be provided by the Parallels Server Bare Metal kernel for Container operation. Currently, this parameter can contain the information about the following technologies: ▪ The system architecture of the Container (`x86`, `x86_64`, or `i64`). ▪ Whether the Container is based on the OS template supporting the Native POSIX Thread Library (NPTL). In this case, the `nptl` entry is specified as the value of this parameter. ▪ Whether the OS EZ template the Container is based on requires the `sysfs` filesystem support (e.g. the OS EZ template for SUSE Linux Enterprise 10).
`DISABLED`	If set to `yes`, disables the Container making it impossible to start the Container once it was stopped. You can start the disabled Container by setting the value of this parameter to `no` or using the `--force` option with the `pctl set` command.
`DESCRIPTION`	Sets the description for the Container. **Note:** You are allowed to use only symbols in the 'A -z' and '0-9' ranges in your descriptions.
`NAME`	The name assigned to the Container. You can use this name, along with the Container ID, to refer to the Container while performing this or that Container-related operation on the server. Follow the following rules while setting the Container name: ▪ The name should contain the `A-Z`, `a-z`, `0-9`, `\`, `-`, and _ symbols only. ▪ If the name consists of two or more words, it should be quoted (e.g. "My Container 101").
`ORIGIN_SAMPLE`	The configuration sample the Container was based on when created.

`CONFIG_CUSTOMIZED`	Indicates whether any of the Container configuration parameters have been modified as regards its original configuration sample. If this parameter is omitted, its value is considered as "no".
`UUID`	The Container unique identifier. This identifier is used by certain Parallels Server Bare Metal utilities during their execution.
`VEFORMAT`	Displays the VZFS version applied to the Container during its creation: ▪ `vz4` denotes that the Container is based on VZFS v2. ▪ `vz3` denotes that the Container is based on VZFS v1. This parameter is meant for your information only and cannot be changed.

All resource management parameters can be subdivided into the CPU, disk, system, and SLM categories for your convenience. Any parameter can be set with the `pctl set` command and the corresponding option name (in the lower case, e.g. `--kmemsize` for `KMEMSIZE`, etc.). See the **Managing Containers** chapter for more details. The **Typical value** column, if present, specifies a range of reasonable parameter values for different applications, from light to huge heavy loaded Containers (consuming 1/8 of server with 2 GB memory). If the barrier and limit fields are in use, ranges for both thresholds are given.

CPU parameters:

Parameter	Description	Typical value
`CPUUNITS`	Guaranteed CPU power. This is a positive integer number, which determines the minimal guaranteed share of the CPU the Container receives. The total CPU power in `CPUUNITS` is its Bogomips number multiplied by 25. Parallels Server Bare Metal reporting tools consider one 1 GHz Intel processor to be approximately equivalent to 50,000 CPU units.	`250…1000`
`CPULIMIT`	Allowed CPU power. This is a positive number indicating the share of the CPU time, in per cent, the Container may never exceed. You can estimate this share as (allowed Container CPUUNITS/CPU power)*100%.	
`CPUS`	The number of CPUs set to handle all the processes inside the given Container. By default, any Container is allowed to consume the CPU time of all processors on the server.	
`BURST_CPU_AVG_USAGE`	The CPU usage limit, in percent, set for the Container. This limit is calculated as the ratio of the current Container CPU usage to the CPU limit (i.e to the value of the `CPULIMIT` parameter) set for the Container in its configuration file. If the limit is not specified, the full CPU power of the server is considered as the CPU limit. Upon exceeding the `BURST_CPU_AVG_USAGE` limit, the `BURST_CPULIMIT` limit is applied to the Container. This parameter redefines the `BURST_CPU_AVG_USAGE` parameter set in the Parallels Server Bare Metal configuration file.	`disabled`

BURST_CPULIMIT	The CPU power limit, in per cent, the Container cannot exceed. The limitations set in this parameter are applied to the Container when it exceeds the limit specified in the `BURST_CPU_AVG_USAGE` parameter. This parameter redefines the `BURST_CPULIMIT` parameter specified in the Parallels Server Bare Metal configuration file.	
Disk parameters:		
DISKSPACE	Total size of disk space that can be consumed by the Container, in 1 Kb blocks.	204800…10485760-204800…11534340
DISKINODES	Total number of disk inodes (files, directories, symbolic links) the Container can allocate.	80000…400000-88000…440000
QUOTATIME	The grace period of the disk quota. It is defined in seconds. The Container is allowed to temporarily exceed its quota soft limits for not more than the `QUOTATIME` period. Specifying `-1` as the value of this setting makes the grace period last 'infinitely'.	0…604800
QUOTAUGIDLIMIT	This parameter defines the maximum aggregate number of user IDs and group IDs for which disk quota inside the given Container will be accounted. If set to 0, the UID and GID quota will be disabled. When managing the quotaugidlimit parameter, keep in mind the following: ▪ Enabling per-user and per-group quotas for a Container requires restarting the Container. ▪ If you delete a registered user but some files with their ID continue residing inside your Container, the current number of ugids (user and group identities) inside the Container will not decrease. ▪ If you copy an archive containing files with user and group IDs not registered inside your Container, the number of ugids inside the Container will increase by the number of these new IDs.	0…500
IOPRIO	The Container priority for disk I/O operations. The higher the priority, the more time the Container has for writing to and reading from the disk. The default Container priority is 4.	0-7
System parameters:		
NUMPROC	Number of processes and threads allowed. Upon hitting this limit, Container will not be able to start a new process or thread.	40…400
AVNUMPROC	Number of processes expected to run in the Container on average. This is informational parameter used by utilities like `vzcfgvalidate` in order to ensure configuration correctness.	0…NUMPROC

NUMTCPSOCK	Number of TCP sockets (PF_INET family, SOCK_STREAM type). This parameter limits the number of TCP connections and, thus, the number of clients the server application can handle in parallel.	40…500
NUMOTHERSOCK	Number of sockets other than TCP. Local (UNIX-domain) sockets are used for communications inside the system. UDP sockets are used for Domain Name Service (DNS) queries, as example. UDP and other sockets may also be used in some very special applications (SNMP agents and others).	40…500
VMGUARPAGES	Memory allocation guarantee, in pages. Applications are guaranteed to be able to allocate memory while the amount of memory accounted as privvmpages does not exceed the configured barrier of the vmguarpages parameter. Above the barrier, memory allocation is not guaranteed and may fail in case of overall memory shortage.	1725…107520
KMEMSIZE	Size of unswappable kernel memory, allocated for internal kernel structures for the processes of a particular Container. Typical amounts of kernel memory is 16…50 Kb per process.	798720…13148160-851968…14024704
TCPSNDBUF	The total size of send buffers for TCP sockets, i.e. the amount of kernel memory allocated for data sent from applications to TCP sockets, but not acknowledged by the remote side yet.	159744…5365760-262144…10458760
TCPRCVBUF	Total size of receive buffers for TCP sockets. Amount of kernel memory, received from remote side but not read by local application yet.	159744…5365760-262144…10458760
OTHERSOCKBUF	Total size of UNIX-domain socket buffers, UDP and other datagram protocols send buffers.	61440…1503232-163840…4063232
DGRAMRCVBUF	Total size of receive buffers of UDP and other datagram protocols.	32768…262144
OOMGUARPAGES	Out-of-memory guarantee, in pages. Any Container process will not be killed even in case of heavy memory shortage if current memory consumption (including both physical memory and swap) until the oomguarpages barrier is not reached.	1725…107520
LOCKEDPAGES	Memory not allowed to be swapped out (locked with the mlock() system call), in pages (one page is 4 Kb).	4…4096
SHMPAGES	Total size of shared memory (including IPC, shared anonymous mappings and tmpfs objects), allocated by processes of a particular Container, in pages.	512…16384
PRIVVMPAGES	Size of private (or potentially private) memory, allocated by an application. Memory that is always shared among different applications is not included in this resource parameter.	3072…151200-3450…1612800
NUMFILE	Number of files opened by all Container processes.	512…8192

`NUMFLOCK`	Number of file locks created by all Container processes.	`50…200 – 60…220`
`NUMPTY`	Number of pseudo-terminals. For example, the `ssh` session, `screen`, the `xterm` application consumes pseudo-terminal resources.	`4…64`
`NUMSIGINFO`	Number of `siginfo` structures (essentially this parameter limits the size of signal delivery queue).	`256…512`
`DCACHESIZE`	Total size of `dentry` and `inode` structures locked in memory. As example, application, first opening the `/etc/passwd` file, locks entries corresponding to `etc` and `passwd` inodes. If a second application opens the `/etc/shadow` file – only entry corresponding to `shadow` is charged, because `etc` is charged already.	`1843200-1966080…4194304`
`PHYSPAGES`	Total size of RAM used by processes. This parameter is used for accounting purposes only. It shows the usage of RAM by the Container. For memory pages used by several different Containers (mappings of shared libraries, for example), only a fraction of a page is charged to each Container. The sum of the `physpages` for all Containers corresponds to the total number of pages used in the system by all accounted users.	`Not limited`
`NUMIPTENT`	The number of IP packet filtering entries.	`12…128`
`MEMINFO`	Customizes the output of the `/proc/meminfo` virtual file inside the Container and sets it to one of the following modes: ▪ `none` (*non-virtualized*). In this case running the `cat /proc/meminfo` command inside the Container will display information about physical memory on the server (total, used, free, shared, etc.), in kilobytes. ▪ `pages:`*`num`* (*virtualized in pages*). Setting the `/proc/meminfo` output to this mode allows you to specify what amount of total memory will be displayed while running the `cat /proc/meminfo` command inside the Container. ▪ `privvmpages:`*`num`* (*virtualized in privvmpages*). Setting the `/proc/meminfo` output to this mode also allows you to arbitrarily specify the amount of total memory to be displayed while running the `cat /proc/meminfo` command inside the Container. As distinct from the previous mode, the amount of memory to be shown in this mode is calculated on the basis of the value of the `PRIVVMPAGES` parameter set in the Container configuration file.	

SLM parameters:

SLMMODE	Defines the behavior of the SLM and UBC parameters in respect of the given Container: ▪ if set to ubc, disables the SLM mode for the Container. ▪ if set to slm, enables the SLM mode for the Container. ▪ if set to all, both the SLM and UBC parameters are supported and can be used to manage the amount of memory which can be consumed by the Container.	all
SLMMEMORYLIMIT	The total amount of memory that can be consumed by the Container, in bytes. This parameter has effect only if the SLMMODE parameter is set to either slm or all.	134217728... 1073741824
SLMPATTERN	Defines the SLM pattern rules for grouping the processes running inside the Container. This parameter overrides the value set in the SLMPATTERN parameter in the Parallels Server Bare Metal global file.	

Network-related parameters allow you to set bandwidth management parameters, hostname and IP addresses that a Container can use as well as to indicate those iptables modules that can be loaded to the Container:

HOSTNAME	If this parameter is specified, then pctl will set the hostname to its value upon the next Container start. This parameter can be omitted. In this case, the Container administrator should configure the hostname manually.
IP_ADDRESS	This is the list of IP addresses, which can be used on Container network interfaces. This list is an argument of the Container start call and it is impossible to assign IP address from inside the Container if the address is not on the list. Any IP address assigned from within the Container will be visible only within the Container.
NAMESERVER	The IP address of the DNS server the Container is supposed to use. More than one server can be specified in the space-separated format.
SEARCHDOMAIN	DNS search domains for the Container. More than one domain can be specified.
NETDEV	The names of physical network adapters that have been moved from the server to the given Container.
IPTABLES	Overrides the IPTABLES parameter from the Parallels Server Bare Metal global configuration file.
NETIF	Specifies a number of parameters for the virtual network adapters existing inside the Container. These parameters include: ▪ ifname: the name of the veth virtual Ethernet interface inside the Container. ▪ mac: the MAC address assigned to the veth virtual Ethernet interface inside the Container. ▪ host_mac: the MAC address assigned to the veth virtual Ethernet interface on the server. ▪ network: the name of the virtual network where the veth virtual network adapter is included.

- `ip`: the IP address(es) assigned to the `veth` virtual network adapter.

`RATE`	If traffic shaping is turned on, then this parameter specifies bandwidth guarantee, in Kb/s, for the Container. The parameters should be set in the form of `“eth0:1:8”`.
`RATEBOUND`	If set to “yes”, the bandwidth guarantee is also the limit for the Container, and the Container cannot borrow the bandwidth from the `TOTALRATE` bandwidth pool.

Backup-related parameters, if present, allow you to specify the number of backups to store. If absent, these parameters are taken from the global backup configuration file or the backup configuration file for a particular server.

`BACKUP_CHAIN_LEN`	An incremental backup parameter. After this number of incremental backups, a full backup is performed.	`7`
`BACKUP_CHAIN_DAYS`	An incremental backup parameter. After this number of days a full backup is performed.	`7`
`BACKUP_KEEP_MAX`	The number of backups to store. Only full and plain full backups are accounted. If a regular backup is being performed that exceeds this number, the oldest backup is automatically deleted. This parameter is effective only if the `-p` option is specified with the `vzbackup` utility. If there is no `-p` option, the number of backups to store is not limited whatever the value of this parameter.	`3`

Linux Distribution Configuration Files

Some Parallels Server Bare Metal utilities (e.g. `pctl`) need to run special scripts inside a Container to perform certain operations on it. However, carrying out one and the same operation inside Containers running different Linux versions may require execution of different actions. This may be caused by the fact that different Linux distributions store files in different locations, use different commands to complete one and the same task, and so on. To distinguish between Containers running different Linux versions and to determine what scripts should be executed while performing the relevant Container-related operations, Parallels Server Bare Metal uses special distribution configuration files located in the `/etc/vz/conf/dists` directory on the server.

There are a number of distribution configuration files shipped with Parallels Server Bare Metal by default (`centos.conf`, `fedora-core.conf`, `gentoo.conf`, etc.). To view all configuration files available on your Parallels Server Bare Metal, you can go to the `/etc/vz/conf/dists` directory and issue the `ls` command. The distribution configuration files will be displayed in the form of *`Linux_Distribution_Name-version`*`.conf` where *`Linux_Distribution_Name`* and *`version`* denote the name of the Linux distribution and its version, respectively (e.g. `fedora-core-7.conf`).

Any distribution configuration file consists of a number of entries in the form of <*`parameter_name`*>=<*`script_name`*> where <*`parameter_name`*> denotes the name of the parameter defining the operation when the script in the right part of the entry is to be executed and <*`script_name`*> is the name of the script to be run on performing the operation defined by the parameter in the left part of the entry. In the current version of Parallels Server Bare Metal, the following parameters are used to define what scripts should be executed for the corresponding Linux version a Container is running:

- `ADD_IP`: the script specified as the value of this parameter has the default name of <*`distribution_name`*>`-add_ip.sh` and is used to configure the network settings during the Container startup and the IP address(es) assignment. The script is launched inside the Container on executing the following commands:

```
pctl start CT_ID
pctl set CT_ID --ipadd <ip_address>
pctl set CT_ID --ipadd <ip_address> --ipdel all
```

- `DEL_IP`: the script specified as the value of this parameter has the default name of <*`distribution_name`*>`-del_ip.sh` and is used to delete an existing IP address from the Container. The script is launched inside the Container on executing the following commands:

```
pctl set CT_ID --ipdel <ip_address>
pctl set CT_ID --ipdel all
```

- `SET_HOSTNAME`: the script specified as the value of this parameter has the default name of <*`distribution_name`*>`-set_hostname.sh` and is used to configure the hostname of the Container. The script is launched inside the Container on executing the following command:

```
pctl set CT_ID --hostname <name>
```

- `SET_DNS`: the script specified as the value of this parameter has the default name of <*`distribution_name`*>`-set_dns.sh` and is used to configure DNS parameters in the `/etc/resolv.conf` file. The script is launched inside the Container on executing the following command:

```
pctl set CT_ID --searchdomain <domain> --nameserver <ip_address>
```

- SET_USERPASS: the script specified as the value of this parameter has the default name of <*distribution_name*>-set_userpass.sh and is used to add a new user or change the current password. The script is launched inside the Container on executing the following command:

```
pctl set CT_ID --userpasswd <user:passwd>
```

- SET_UGID_QUOTA: the script specified as the value of this parameter has the default name of <*distribution_name*>-set_ugid_quota.sh and is used to set up second level quota. The script is launched inside the Container on executing the following command:

```
pctl set CT_ID --quotaugidlimit <num>
```

- POST_CREATE: the script specified as the value of this parameter has the default name of <*distribution_name*>-postcreate.sh and is used to perform certain tasks (e.g. to modify the crontab files) after the Container creation. This script is launched on the server on executing the following command:

```
pctl create CT_ID
```

- POST_MIGRATE: the script specified as the value of this parameter has the default name of <*distribution_name*>-post_migrate.sh and is used to perform certain operations on the Container where the physical server has been successfully migrated. This script is launched inside the Container on executing the following command:

```
vzp2v [options] --ctid CT_ID
```

- GET_V2PMIGRATE_EXCLUDES: the script specified as the value of this parameter has the default name of <*distribution_name*>-v2pmigrate-excludes.sh and is used to generate a list of files and directories which will be excluded from migrating from the Container to the physical server. This script is launched inside the Container on executing the following command:

```
vzv2p [options] --ctid CT_ID
```

The scripts specified in distribution configuration files are located in the /etc/vz/conf/dists/scripts directory on the server and executed on performing the aforementioned operations on the Containers. After an operation has been initiated, the pctl, vzp2v, or vzv2p utility turns to the corresponding Container configuration file, looks for the value of the DISTRIBUTION variable or, if the latter is not present, of the OSTEMPLATE variable in this file, and defines on their basis what Linux version the given Container is running. After that, pctl reads the corresponding configuration file for the determined Linux version from the /etc/vz/conf/dists directory and executes the scripts specified in this file.

Note: If no distribution is specified as the value of the DISTRIBUTION and OSTEMPLATE variables in the Container configuration file or no configuration file for the given Linux version was found in the /etc/vz/conf/dists directory, the default file from this directory is used.

Network Classes Definition File

In Parallels Server Bare Metal, both traffic accounting and bandwidth management are based on network classes. The network classes' definition file (`/etc/vz/conf/networks_classes`) describes network classes that Parallels Server Bare Metal recognizes. Currently, there can be up to 15 classes defined.

The lines in this file have the following format:

```
<class_id> <ip_address>/<prefix_length>
```

where `<class_id>` defines the network class identifier, *`<ip_address>`* defines the starting IP address, and *`<prefix_length>`* defines the subnet mask. In pair *`<ip_address>`* and *`<prefix_length>`* define the range of IP addresses for this class. There may be several lines for each class. Classes should be defined after Class 1 and represent exceptions from the "matching-everything" rule of Class 1. Class 0 has a special meaning and defines the IP ranges for which no accounting is done (this server Container addresses).

The definition of class 1 is required; any class except class 1 can be omitted. However, it is recommended to define class 0 correctly - it will improve performance:

```
# server Container networks
0 192.168.0.0/16
```

In the default Parallels Server Bare Metal installation, only class 1 is defined and it matches any packets:

```
# all IP-addresses ("local" traffic)
1 0.0.0.0/0
```

Several lines may add more networks in the same class:

```
# class 2 - "foreign" traffic
2 10.0.0.0/8
2 11.0.0.0/8
```

Also, inside of one class it is possible to define sub-networks of another class:

```
# inside "foreign" network there
# is a hole with "local" traffic
1 10.10.16.0/24
```

vzup2date Configuration File

The `/etc/sysconfig/vzup2date/vzup2date.conf` file is used by the `vzup2date` utility on the step of connecting to the repository with storing the latest Parallels Server Bare Metal updates.

The parameters in this file are presented on separate lines in the following format:

```
<parameter_name>=<parameter_value>
```

The table below describes these parameters:

Parameter	Description	Example
`Server`	The URL used for the connection.	`https://vzup2date.parallels.com`
`User`	The user name for accessing the update server.	`user1`
`Password`	The password for accessing the update server.	`sample`
`HTTP_PROXY`	The proxy server address, if you use this server.	`http://192.168.1.20`
`HTTP_PROXY_USER`	The user name used by the HTTP proxy server for your authentication.	`peter`
`HTTP_PROXY_PASSWORD`	The password of the user specified in the `HTPP_PROXY_USER` parameter and used for your authentication by the HTTP proxy server.	`2wed45r`
`LocalRepositoryDir`	The path to the local directory on the server where the downloaded Parallels Server Bare Metal updates are stored. By default, the `/vz/vzup2date` directory is used.	`/vz/vzup2date`
`LogFile`	The path to the log file on the server containing the information on Parallels Server Bare Metal updates. By default, the `/var/log/vzup2date.log` file is used.	`/var/log/vzup2date.log`

Not all the possible parameters must be necessarily present in this file. In fact, all the parameters are optional, i.e. if they are missing from this file, the `vzup2date` utility will ask for the user input without suggesting its own variant taken from this file.

vzup2date-mirror Configuration File

The `vzup2date-mirror` configuration file is used by the `vzup2date-mirror` utility for determining the connection parameters of the repository with Parallels Server Bare Metal system and templates updates and deciding what updates to download to the local mirror. The parameters in this file are presented on separate lines in the following format:

```
<parameter_name>=<parameter_value>
```

The options that can be specified in the `vzup2date.conf` file are described in the table below:

Parameter	Description	Example
`Server`	The URL used for the connection. As a rule, this parameter is set automatically and does not need to be modified.	`https://vzup2date.parallels.com`
`User`	The user name for accessing the update server. As a rule, this parameter is set automatically and does not need to be modified.	`user1`
`Password`	The password for accessing the update server. As a rule, this parameter is set automatically and does not need to be modified.	`sample`
`HTTP_PROXY`	The proxy server address, if you use this server.	`http://192.168.1.20`
`HTTP_PROXY_USER`	The user name used by the HTTP proxy server for your authentication.	`peter`
`HTTP_PROXY_PASSWORD`	The password of the user specified in the `HTTP_PROXY_USER` parameter and used for your authentication by the HTTP proxy server.	`2wed45r`
`LocalRepositoryRoot`	The local directory where the mirror is to be located and all the required packages are to be stored after the execution of `vzup2date-mirror`. This parameter can be overwritten by the `local_repo_path` parameter of the `vzup2date-mirror` utility (to learn more about `local_repo_path`, see the vzup2date-mirror subsection).	`/var/www/html`

`Releases`	The list of comma-separated Parallels Server Bare Metal releases or OS templates names. The format of this parameter is different for different types of updates: ▪ For system updates, you should set it in the *`arch`*/*`Parallels Server Bare Metal_release`* format. ▪ For EZ templates updates, you should set it in the *`arch`*/*`EZ_template_name`* format. ▪ For standard templates updates, it should be set in the *`arch`*/*`standard_template_ name`* format. By default, the value of this parameter is set to `all/all` meaning that all available updates for all system architectures will be downloaded from the Parallels Server Bare Metal official repository to your local mirror.	`i386/4.0.0`
`MirrorName`	The name assigned to the mirror. You must specify this parameter for each mirror if you are planning to have several mirrors with different `LocalRepositoryRoot` parameters operating simultaneously on your server (in one Container). These mirror names will be used by the `apache` application to distinguish among the existing mirrors.	`Mirror1`
`HTTPD_CONFIG_ FILE`	The path to the `httpd` configuration file. This file is required for the correct work of the `apache` application. As you can create an HTTP-based mirror only, the `apache` application should be installed on the server and a valid path to `httpd.conf` should be specified. By default, this parameter is set to `/etc/httpd/conf/httpd.conf`. If you have not change the default `httpd.conf` file location, you do not need to change this parameter.	`/etc/httpd/conf/ httpd.conf`

The `vzup2date-mirror` configuration file can also include a section defining the updates approval policy for deploying Parallels Server Bare Metal system updates to the servers in your local network. This section must be opened with the `<ApproveSystemUpdate` *`arch/release`*`>` tag (where *`arch`* denotes the system architecture (e.g. `x86_64`) and *`release`* denotes the Parallels Server Bare Metal release (e.g. `4.0.0`) the specified policy will be applied to) and closed with the `</ApproveSystemUpdate>` tag. This section is optional. If it is absent from the configuration file, all updates downloaded to your local mirror are automatically approved for installation on your servers. The parameters that can be specified in this section are described in the table below:

Parameter	Description
`CU`	The maximum version of Parallels Server Bare Metal kernel updates for the specified architecture/release pair. All Parallels Server Bare Metal kernel updates having higher versions and downloaded to your local mirror will be invisible for the `vzup2date` utility that you will run on the servers in your local network.
`TU`	The maximum version of Parallels Server Bare Metal tools and utilities updates for the specified architecture/release pair. All tools and utilities updates having higher versions and downloaded to your mirror will be invisible for the `vzup2date` utility that you will run on the servers in your local network.
`MU`	Enables (`yes`) or disables (`no`) the `vzup2date` utility to download the next major version update of the Parallels Server Bare Metal software.If this parameter or the whole updates approval mechanism section is omitted, major updates are available to the `vzup2date` utility by default.

vzvpn Configuration File

The `/etc/vzvpn/vzvpn.conf` file is used by the Parallels Support Tool to establish a secure connection (a virtual private network) between your server and the Parallels support server.

The parameters in this file are presented on separate lines in the following format:

```
<parameter_name>=<parameter_value>
```

The table below describes these parameters:

Parameter	Description
`REMOTE_HOST`	Mandatory. The hostname or the IP address of the Parallels support server.
`REMOTE_PORT`	Mandatory. The port number of the Parallels support server to be used for establishing a virtual private network (VPN).
`STARTTMO`	Mandatory. The time, in seconds, during which there will be attempts to start the Parallels Support Tool if it could not be started immediately after its launching.
`INACTIVE`	Mandatory. The time of inactivity, in seconds, after which the connection between your server and the Parallels support server will be closed.
`PING`	Mandatory. The time, in seconds, at the end of which the port of the Parallels support server will be pinged in case no packets have been received from the support server during the time specified.
`PING_EXIT`	Mandatory. The time, in seconds, after a lapse of which the connection between your server and the Parallels support server will be closed in case no ping signals or other packets have been received from the support server during this time.
`HTTP_PROXY=hostname[:`*`port`*`]`	Optional. The hostname or the IP address and the port number of the HTTP proxy server through which a VPN between your server and the Parallels support server is to be established. This parameter overrides the `HTTP_PROXY` parameter set in the `/etc/vz/vz.conf` file on the server. If the `HTTP_PROXY` parameter is not specified in either of the files, the Parallels Support Tool looks for the `http_proxy` environment variable on the server and takes its value for establishing a VPN.
`HTTP_PROXY_USER`	Optional. The user name used by the HTTP proxy server for your authentication.
`HTTP_PROXY_PASSWORD`	Optional. The password of the user specified in the `HTTP_PROXY_USER` parameter and used for your authentication by the HTTP proxy server.

Note: You are not recommended to change any of the aforementioned parameters. Modify them only if you are dead certain of your actions (for example, you have received the corresponding information from Parallels).

vzreport Configuration File

The `/etc/vzreport.conf` file is used by the `vzreport` utility to submit a problem report to the Parallels support team.

The parameters in this file are presented on separate lines in the following format:

```
<parameter_name>=<parameter_value>
```

The table below describes these parameters:

Parameter	Description
`SUBMIT_URI`	The Uniform Resource Identifier (URI) of the Parallels support server to be used to receive and gather your problem reports.
`COLLECTOR_SCRIPT`	The path to the file on your server where the information on your problems reports is collected. This is the same data that is sent to the Parallels support server.
`HTTP_PROXY`	The hostname or the IP address of the HTTP proxy server through which your problem report will be sent to the Parallels support team.
`HTTP_PROXY_USER`	The user name used by the HTTP proxy server for your authentication.
`HTTP_PROXY_PASSWORD`	The password of the user specified in the `HTTP_PROXY_USER` parameter and used for your authentication by the HTTP proxy server.

Not all the possible parameters should be necessarily present in this file. In fact, all the parameters are optional except for the `SUBMIT_URI` parameter which should be specified to tell the `vzreport` utility where to send your problem report.

Kernel Parameters

There is a number of kernel limits that should be set for the Parallels Server Bare Metal software to work correctly. Parallels Server Bare Metal is shipped with a tuned `/etc/sysctl.conf` file. Understanding what parameters were changed is essential for running the required number of Containers. Below is the contents of the `/etc/sysctl.conf` file as shipped with Parallels Server Bare Metal:

```
# On the server we generally need
# packet forwarding enabled and proxy arp disabled
net.ipv4.ip_forward = 1
net.ipv4.conf.default.proxy_arp = 0
# Enables source route verification
net.ipv4.conf.all.rp_filter = 1
# Enables the magic-sysrq key
kernel.sysrq = 1
# TCP Explict Congestion Notification
#net.ipv4.tcp_ecn = 0
# ARP thresholds. First one is num_ve x 3 + 512
# second one is 2 times first one
net.ipv4.neigh.default.gc_thresh2 = 2048
net.ipv4.neigh.default.gc_thresh3 = 4096
# we do not want all our interfaces to send redirects
net.ipv4.conf.default.send_redirects = 1
net.ipv4.conf.all.send_redirects = 0
```

Notice that some parameters of the kernel configuration depends on the maximum number of Containers you plan to run. In the default configuration file, these numbers were calculated under the assumption the maximum Container number is 512. If you plan to run another number of Containers, it is recommended to recalculate `net.ipv4.neigh.default.gc_thresh2` and `net.ipv4.neigh.default.gc_thresh3` parameters as three per Container plus 128…512. Keep the second parameter twice as great as the first one.

To apply the changes issue the following command:

```
# sysctl -p
```

Besides, it makes sense to set `net.ipv4.tcp_use_sg` to 0, since corresponding "Scatter/gather IO" feature is not supported by the `venet` device, used in Parallels Server Bare Metal networking.

It is also worth mentioning that normally you should have forwarding turned on since the server forwards packets destined to or originated from Containers.

Offline Management Configuration Files

The offline management configuration files located in the `/etc/vzredirect.d` directory define various modes of Container offline management by Container administrators. One configuration file describes one offline management mode. In the current Parallels Server Bare Metal version, two files are accessible: `vzpp.conf` and `vzpp-plesk.conf`. The first file defines the Container offline management by means of Parallels Power Panel, and the second one - by means of the same Power Panel with an integrated Plesk control panel.

There are two parameters in each of the files. They are presented on separate lines in the following format:

```
<parameter_name>=<parameter_value>
```

The table below describes these parameters:

Parameter	Description	Example
`PORT`	This port must be entered in the address line of an Internet browser after the Container IP address when managing the Container by means of Parallels Power Panel or the Plesk control panel.	`PORT=8443`
`DST_VEID`	The ID of the Container where the requests coming to the specified port will be redirected.	`DST_VEID=1`

vzlmond Configuration File

The `/etc/vzlmond.conf` file defines the configuration parameters for the `vzlmond` daemon used to periodically check and log the state of your server. The gathered logs can then used by the `vzstatrep` utility to generate statistic reports and graphics on their basis and to send these reports and graphics to the server administrator's e-mail address(es). Detailed information on the `vzstatrep` utility is provided in the `vzstatrep` subsection (p. 82).

The parameters in this file are presented on separate lines in the following format:

```
<parameter_name>=<parameter_value>
```

The table below describes these parameters:

Name	Description	Default Value
`STATS_VMSTAT_PERIOD`	The periodicity, in seconds, with which the `vmstat` utility is run on the server and its output is saved to log files in the directory specified as the value of the `LOGS_DIR` parameter. The `vmstat` output contains information on the server kernel threads, virtual memory, disks, traps, and CPU activity. For more information on `vmstat`, see its man pages.	480
`STATS_FULLDUMP_PERIOD`	The period, in seconds, at the end of which the complete statistics on the server resources consumption is gathered and logged to the directory specified as the value of the `LOGS_DIR` parameter. As distinct from the `vmstat` output, this statistics represents a snapshot of the files contents from the `/proc` directory on the server and contains information on virtually every server resource: the environment of a certain process, the state and configuration of the CPU(s), the number of I/O ports on the server and their configuration, etc. Keep in mind that the amount of disk space needed to store this information may be considerable (about 0,5 Kb per Container). However, you are recommended to set the period to no more than 10 minutes to regularly check and log the current server state and resources consumption.	480
`STATS_NET_PERIOD`	The period, in seconds, after which the server network statistics is collected and logged to the directory specified as the value of the `LOGS_DIR` parameter. The network statistics is gathered separately for each network interface on the server (e.g. `eth0`, `eth1`).	480
`LOGS_DIR`	The name of the directory on the server where the gathered statistics is to be	`/var/log/vzstat`

stored.

All the aforementioned parameters are set to their default values during the Parallels Server Bare Metal installation; so, you do not have to additionally edit any parameter in the `/etc/vzlmond.conf` file to start gathering your server statistics.

vzstat Configuration File

This file (`/etc/vzstat.conf`) lists a number of CPU-, memory-, and disk-related parameters used by the `vzstat` utility. The values assigned to these parameters denote either the warning or the error level for the `vzstat` utility to start displaying these parameters either in the yellow color (the warning level has been hit) or in the red color (the error level has been hit). Moreover, if a parameter has hit the error level, the **CRIT** warning is displayed instead of **OK** after the name of the corresponding subsystem (CPU, Memory, Swap, Net, or Disks).

The table below provides information on the name and the description of all these parameters, on whether they denote the warning or the error level, whether the real parameter value has to be higher or lower than this level in order to invoke an alert, and on the parameters default values:

Parameter	Description	Default Value	Alert When	Alert Type
`LOAD_AVG`	Load average.	30	Higher	Warning
`PROC_RUN`	Number of running processes.	20	Higher	Warning
`PROC_UNINT`	Number of uninterruptable processes (in “D” state).	20	Higher	Warning
`CPU_IDLE`	CPU idle time, in percents.	10	Lower	Warning
`CPU_SYS`	CPU system time, in percents.	50	Higher	Warning
`CPU_LAT_MAX_WARN`	Scheduling latency, in milliseconds (maximum over 5 sec period).	750	Higher	Warning
`CPU_LAT_MAX_ERR`	Scheduling latency, in milliseconds (maximum over 5 sec period).	1000	Higher	Error
`CPU_LAT_AVG_WARN`	Scheduling latency, in milliseconds (5 sec average).	500	Higher	Warning
`CPU_LAT_AVG_ERR`	Scheduling latency, in milliseconds (5 sec average).	750	Higher	Error
`MEM_LAT_MAX_WARN`	Memory allocation latency, in milliseconds (maximum over 5 sec period).	300	Higher	Warning
`MEM_LAT_MAX_ERR`	Memory allocation latency, in milliseconds (maximum over 5 sec period).	500	Higher	Error
`MEM_LAT_AVG_WARN`	Memory allocation latency, in milliseconds (5 sec average).	250	Higher	Warning
`MEM_LAT_AVG_ERR`	Memory allocation latency, in milliseconds (5 sec average).	400	Higher	Error
`MEM_ZONE_ACT_INACT_FREE_WARN`	Size of available memory (free + active + inactive pages), in per cent.	8	Lower	Warning
`MEM_ZONE_ACT_INACT_FREE_ERR`	Size of available memory (free + active + inactive pages), in per cent.	4	Lower	Error

`MEM_ZONE_ACT_INACT_FREE_ABS_WARN`	Size of available memory (free + active + inactive pages), in MB.	4	Lower	Warning
`MEM_ZONE_ACT_INACT_FREE_ABS_ERR`	Size of available memory (free + active + inactive pages), in MB.	2	Lower	Error
`MEM_ZONE_ORDER_GT_0`	Number of pages which are gathered in blocks with order > 0. For example, if current memory distribution looks like: 3*1 1*2 3*4 5*8 .... Then number of pages with order>0 is 1*2 + 3*4 + 5*8 + ...	100	Lower	Warning
`SWAP_FREE_WARN`	Free swap space, in percents.	75	Lower	Warning
`SWAP_FREE_ERR`	Free swap space, in percents.	50	Lower	Error
`SWAP_IN_WARN`	Swap-in activity, in Mb/sec.	0.5	Higher	Warning
`SWAP_IN_ERR`	Swap-in activity, in Mb/sec.	1	Higher	Error
`SWAP_OUT_WARN`	Swap-out activity, in Mb/sec.	0.5	Higher	Warning
`SWAP_OUT_ERR`	Swap-out activity, in Mb/sec.	1	Higher	Error
`SWAP_LAT_MAX_WARN`	Swap-in latency, in milliseconds (maximum over 5 sec period).	750	Higher	Warning
`SWAP_LAT_MAX_ERR`	Swap-in latency, in milliseconds (maximum over 5 sec period).	1000	Higher	Error
`SWAP_LAT_AVG_WARN`	Swap-in latency, in milliseconds (5 sec average).	500	Higher	Warning
`SWAP_LAT_AVG_ERR`	Swap-in latency, in milliseconds (5 sec average).	750	Higher	Error
`DISK_FREE_INODES_WARN`	Free inodes on disk, in percents.	20	Lower	Warning
`DISK_FREE_INODES_ERR`	Free inodes on disk, in percents.	5	Lower	Error
`DISK_FREE_SPACE_WARN`	Free disk space, in percents.	20	Lower	Warning
`DISK_FREE_SPACE_ERR`	Free disk space, in percents.	5	Lower	Error
`CT_FAILCNT_DELTA`	Number of failed UBC resource allocations for particular Container between `vzstat` screen updates (any resource type counts).	1	Higher	Error

vzrmond Configuration File

This file (`/etc/vzrmond.conf`) is the configuration file for the `vzrmond` daemon which is running on the Monitor Server and provides the remote monitoring of servers registered in it and the sending of alerts to the specified e-mail addresses. It also allows you to use external applications for sending alerts (e.g. via ICQ or SMS). The file lists a number of parameters some of which have values that should be provided by the user (from `HOSTS` through `CUSTOM_LIST`). These values are included in double quotes and separated by spaces from each other. The remaining parameters have default values that may be altered by the user. They are not included in quotes.

Parameter	Description	Default value
`HOSTS`	The list of hosts to be monitored delimited by spaces. Both hostnames and IP addresses are allowed.	`“”`
`EMAIL_ADDRESSES`	E-mail addresses to receive the alerts. Must be separated by spaces.	`“”`

`EMAIL_NOTIFICATIONS`	The types of notifications to be sent to the specified e-mail address(es).	`SYSTEM_UP` `SYSTEM_DOWN`
		`DISK_OK`
		`DISK_BAD`
		`INODES_NORM` `INODES_HIGH`
		`HDDBUSY_NORM` `HDDBUSY_HIGH`
		`SSH_UP`
		`SSH_DOWN`
		`VZSTAT_OK`
		`VZSTAT_BAD`
		`LOADAVG_NORM` `LOADAVG_HIGH`
		`UNINT_NORM` `UNINT_HIGH`
		`MEMLATM_NORM` `MEMLATM_HIGH`
		`MEMLATA_NORM` `MEMLATA_HIGH`
		`CPULATM_NORM` `CPULATM_HIGH`
		`CPULATA_NORM` `CPULATA_HIGH`
		`SWAPIN_NORM` `SWAPIN_HIGH`
		`SWAPOUT_NORM` `SWAPOUT_HIGH`

`CUSTOM_ACTION`	The program to send alerts of a	`“”`

	customized type (e.g. via ICQ or SMS).	
CUSTOM_LIST	Options passed as the command-line parameters of the program specified by CUSTOM_ACTION. Must be separated by spaces.	""
POLL_PERIOD	Periodicity of checking up the registered servers, in seconds.	15
CHK_MAX_FAILS	After this number of unsuccessful attempts to reach a server, the "Server is dead" alert is sent.	4
LOAD_AVG	The average number of processes on the server. When this value is exceeded, an alert is sent.	30
PROC_UNINT	The number of uninterruptable sleeping processes (in the "D" state). When this value is exceeded, an alert is sent.	20
CPU_LAT_MAX_ERR	The maximal process scheduling latency, in milliseconds. When this value is exceeded, an alert is sent.	1000
CPU_LAT_AVG_ERR	The average process scheduling latency, in milliseconds. When this value is exceeded, an alert is sent.	750
MEM_LAT_MAX_ERR	The maximal memory allocation latency, in milliseconds. When this value is exceeded, an alert is sent.	500
MEM_LAT_AVG_ERR	The average memory allocation latency, in milliseconds. When this value is exceeded, an alert is sent.	400
SWAP_IN_ERR	The swap in activity, in Mb/s. When this value is exceeded, an alert is sent.	1.0
SWAP_OUT_ERR	The swap out activity, in Mb/s. When this value is exceeded, an alert is sent.	1.0
DISK_FREE_INODES_ERR	The percentage of free disk inodes. When the actual value becomes less than this value, an alert is sent.	5
DISK_FREE_SPACE_ERR	The percentage of free disk space. When the actual value becomes less than this value, an alert is sent.	5

To be able to begin monitoring a server, you should provide the valid values for the HOSTS and EMAIL parameters. If you wish to use an external program for sending alerts about the server state, you should install in on the Monitor Server and provide its name and options in the CUSTOM_ACTION and CUSTOM_LIST parameters. The alert message text will be sent as the standard input for the specified program.

You should increase the value of the `POLL_PERIOD` parameter together with the increase in the number of monitored servers not to create an overload on the Monitor Server. The parameters related to the scheduling latency, memory allocation latency, and swap in/out activity serve to have an alert generated if the system's performance plummets due to the abnormal values of these parameters.

Do not forget to restart the `vzrmond` daemon after you have edited this configuration file.

vzstatrep Configuration File

The `vzstatrep.conf` configuration file located in the `/etc` directory on the Monitor Server is used by the `vzstatrep` utility while trying to generate statistic reports and graphics on the server resource consumption and to send them to your e-mail address. This file has a number of lines in the following format:

```
<parameter_name>="parameter_value"
```

Below is a list of available parameters:

Name	Description
NODES	The IP address or hostname of the server whose logs are to be analyzed. You can set several servers for being processed with the help of the `vzstatrep` utility and separate them by spaces. If no server is specified, the logs of the local server (i.e. of the Monitor Server itself) are analyzed.
STATS_EMAIL	The e-mail address to send the generated statistic reports and graphics to. You can specify several e-mail addresses and separate them by commas or spaces.
GNUPLOT	The path to the `gnuplot` utility on the Monitor Server. By default, the utility is located in the `/usr/bin` directory; however, you may specify another directory for its location (e.g. `/etc/mydir/gnuplot`). `gnuplot` is used by the `vzstatrep` utility to present the server resources consumption in the graphical form. The resources whose graphics are to be generated should be set as the values of the `STATS_PLOT` parameter. For detailed information on the `gnuplot` utility, see its man pages.
MUTT	The path to the `mutt` utility on the Monitor Server. By default, the utility is located in the `/usr/bin` directory; however, you may specify another directory for its location (e.g. `/etc/mydir/mutt`). `mutt` is used by the `vzstatrep` utility to send the generated statistic reports and graphics in the form of attached files via e-mail. For detailed information on the `mutt` utility, see its man pages.
LOGS_DIR	The path to the directory on the server where `vzstatrep` will search for the logs generated by the `vzlmond` daemon and containing the information on the server resources consumption. By default, the `/var/log/vzstat` directory is used. If you have changed the directory where `vzlmond` stores the gathered information, you should specify the full path to this directory as the value of this parameter (e.g. `LOGS_DIR=/my_logs/vzstat`).
STATS_PLOT	Specify the resources parameters whose graphics are to be generated by means of the `gnuplot` utility. You can specify several resources and separate them by spaces. Currently, you can create graphics for the following parameters: ▪ `ve_sum`: the information on the CPU usage for all Containers on the server. ▪ `ve_top`: the information on the CPU usage for 5 Containers with the highest CPU consumption. ▪ `loadavg`: the average number of active processes for the past 1, 5, and 15 minutes. Active processes can be running, i.e. currently executed by the CPU, or runnable, i.e. waiting in the run queue for the CPU. ▪ `io`: the amount of data read from and written to all devices on the server, in kilobytes per second.

- `mem`: the total memory consumption on the server.
- `ints`: the number of interrupts and context switches on the server per second.
- `cpu`: the information on the CPU load on the server.
- `net`: the network information for each network interface on the server.
- `forks`: the number of copies of all processes made on the server during one second.

By default, all the aforementioned resources except for `ve_sum` are plotted.

To start analyzing the logs, creating the server statistic reports and graphics, and receiving e-mail messages with these reports and graphics, you should specify the `NODES` and `STATS_EMAIL` parameters in the `/etc/vzstatrep.conf` file. All the other parameters are automatically set during the `vzrmon` package installation on the Monitor Server.

Backup Configuration File

This file (`/etc/vzbackup.conf`) is in the same format as the global Parallels Server Bare Metal configuration file and per-Container configuration files. All the parameters define the global backup settings, but some of them may be overridden by the per-server configuration file, if the latter exists. Still, other parameters may be further overridden in the configuration file of a particular Container.

All-server parameters:

Parameter	Description	Default value
`BACKUP_DIR`	The backup directory, i.e. the directory where backups are stored.	`/vz/backup`
`BACKUP_TYPE`	The backup type. Among the supported types are "plain full (`F`)", "full (`I`)", and "incremental (`i`)". The default is incremental. If it is impossible to do an "incremental" backup, a "full" backup will be made.	`i`
`BACKUP_NODES`	The hostname of the server whose Containers are to be backed up. You can specify several hostnames of your servers and separate them by spaces. If you wish to back up Containers residing on the Backup server itself, you should specify its hostname as the value of this parameter.	
`BACKUP_MAX_CHLD`	The maximal number of servers to back up in parallel for non-periodic backups.	`1`
`BACKUP_MAX_CHLD_CRON`	The maximal number of servers to back up in parallel for periodic backups.	`3`
`BACKUP_NOTIFY_EMAIL`	The e-mail addresses where to send notifications about the backing up.	
`BACKUP_COMPRESS`	Specifies whether the Containers are to be compressed when being backed up, and with what compression algorithm. When backing up Containers, you can set this option to one of the following values: ▪ `C0`: in this case the Container backup is created without any compression. Using this level of compression, you may greatly reduce the backup creation time; however, the size of the resulting backup file may significantly increase as compared to other compression levels. ▪ `C1`: in this case the Container backup is created with a normal level of compression. ▪ `C2`: in this case the Container backup is created with the high level of compression. The size of the resulting backup file is smaller than that of the backup file compressed in the 'normal' and 'none' modes; however, it takes longer to create	`none`

	the backup file. ▪ `C3`: in this case the Container backup is created with the maximal level of compression. The size of the resulting backup file is the smallest and the time of the backup creation - the longest. This parameter can be overridden by the `-Cg`, `-Cb`, `-Cn` command-line options of the `vzbackup` utility.	
`CRON_BACKUP`	Specifies whether the backing up is performed as a cron job. If set to "yes", the values of the `BACKUP_KEEP_MAX` and `BACKUP_LOADAVG_MAX` parameters in the given file are taken into consideration. This parameter can be overridden by the `-p` or `-j` command-line switch of the `vzbackup` utility.	`no`

Per-Node parameters:

Parameter	Description	Default value
`BACKUP_SSH_OPTS`	Options which are passed to `ssh` when it is used.	`-c blowfish`
`BACKUP_VESTOP`	Defines whether the Containers are to be stopped before their backing up. If set to `–s`, the Containers are stopped by default, otherwise, they are not stopped.	
`BACKUP_EXCL_VES`	Defines those Containers that are to be excluded from the backup list. Container IDs must be given here.	
`BACKUP_LOADAVG_MAX`	The maximal `loadavg` with which backing up is allowed. This parameter is effective only if the `-p` option is specified with the `vzbackup` utility.	`10`
`BACKUP_FINISH_TIME`	The time when the backing up should be stopped and delayed until the next execution, e.g. when running backup scripts at 4am, one can require the backup to be finished before 7am. The backup will continue from the last Container at the next execution. The format is: "`HH:MM`". This parameter is effective only if the `-L` option is specified with the `vzbackup` utility.	`none`
`BACKUP_LIMIT_TIME`	The number of hours after which the backing up should be stopped and delayed until the next execution. The format is: "`HH`". This parameter is effective only if the `-L` option is specified with the `vzbackup` utility.	`none`

Per-Container parameters

Parameter	Description	Default Value
`BACKUP_CHAIN_LEN`	An incremental backup parameter. After this number of incremental backups, a full backup is performed.	`7`

BACKUP_CHAIN_DAYS	An incremental backup parameter. After this number of days a full backup is performed.	7
BACKUP_KEEP_MAX	The number of backups to store. Only full and plain full backups are accounted. If a regular backup is being performed that exceeds this number, the oldest backup is automatically deleted. This parameter is effective only if the `-p` option is specified with the `vzbackup` utility. If there is no `-p` option, the number of backups to store is not limited whatever the value of this parameter.	3

If you want to rewrite the per-server parameters for a particular Parallels server, you should create a new configuration file named `server.conf` and put it to the backup directory (defined by the `BACKUP_DIR` parameter in the global backup configuration file.

vzrhnproxy Configuration File

This file (`/etc/vz/pkgproxy/rhn.conf`) is the configuration file for `vzrhnproxy` - a special utility which can be used on any RHEL-based server (e.g. RHEL 4 or 5, Fedora Core 5 or , CentOS 4 or 5) to create RHN (Red Hat Network) Proxy Servers allowing you to effectively manage the RPM packages included in the RHEL 4 and 5 OS EZ templates.

The parameters in this file are presented on separate lines in the following format:

<*parameter_name*>=<*parameter_value*>

The table below describes these parameters:

Parameter Name	Description
REDHAT_LOGIN	The user name for logging in to Red Hat Network.
REDHAT_PASSWORD	The password of the user specified as the value of the `REDHAT_LOGIN` parameter.
HTTP_PROXY	The hostname or the IP address and the port number of the HTTP proxy server, if you use any to connect to the Internet.
HTTP_PROXY_USER	The user name used by the HTTP proxy server for your authentication.
HTTP_PROXY_PASSWORD	The password of the user specified in the `HTTP_PROXY_USER` parameter and used for your authentication by the HTTP proxy server.
EMAIL	The destination of all tracebacks.
PRE_DOWNLOAD	The names of the packages to be downloaded when running the `vzrhnproxy update` command. The names of the packages listed as the value of this parameter should correspond to the names of real packages in the RHEL repository in Red Hat Network and can be specified as regular expressions (e.g. `perl.*`).

vzpkgproxy Configuration File

This file (`/etc/vzpkgproxy/vzpkgproxy.conf`) is the configuration file for `vzpkgproxy` - a special utility which can be used to create special caching proxy servers allowing you to efficiently manage your OS and application EZ templates.

The parameters in this file are presented on separate lines in the following format:

<*parameter_name*>=<*parameter_value*>

The table below describes these parameters:

Parameter Name	Description
`REPO_DIR`	The path to the directory on the proxy server where the local repository created on the basis of the cached packages is to be stored. By default, this directory has the path of `/var/www/html/download`.
`CACHE_DISABLE`	The IP addresses of the hosts to be excluded from the caching process. It means that the packages requested by servers and received from these hosts will not be cached on the proxy server. By default, the proxy server is configured to cache all packages from all hosts on external networks.

vztt Configuration File

This file (`/etc/vztt/vztt.conf`) is the configuration file used by the `vzpkg` utility when managing OS and application EZ templates.

The parameters in this file are presented on separate lines in the following format:

<*parameter_name*>=<*parameter_value*>

The table below describes these parameters:

Parameter Name	Description
`VZTT_PROXY`	The IP address or hostname of the caching proxy server to be used by the `vzpkg` tool for managing OS and application EZ templates.
`HTTP_PROXY`	The IP address or hostname of the HTPP proxy server address, if you use this server.
`HTTP_PROXY_USER`	The user name used by the HTTP proxy server for your authentication.
`HTTP_PROXY_PASSWORD`	The password of the user specified in the `HTPP_PROXY_USER` parameter and used for your authentication by the HTTP proxy server.
`METADATA_EXPIRE`	Defines the period of time, in seconds, in the course of which the downloaded software packages in the `vzpkg` cache are regarded as 'not obsolete'. During this time, the `vzpkg` utility searches for the EZ template packages in the local cache only (without checking the remote repositories set for EZ templates). By default, this period is set to 86400 seconds (24 hours).
`EXCLUDE`	List of comma-separated packages that are not to be installed or updated during the `vzpkg` execution. The package names should correspond to the name of real packages in the repository and can contain file globs (e.g. `*` and `?`).

Parallels Server Bare Metal Scripts

Overview

Along with Parallels Server Bare Metal configuration files responsible for the Parallels Server Bare Metal system configuration, there are a number of scripts allowing you to customize the Container behavior in different ways. These are the following scripts:

Script Name	Description
`/vz/private/<CT_ID>/scripts/<action>`	Container private action scripts. These scripts allow to run user-defined actions on particular events. The currently defined actions are `start`, `stop`, `mount`, `unmount`.
`/etc/vz/conf/dists/scripts/<script>`	Scripts to be executed on performing certain Container-related operations (e.g. on adding a new IP address to the Container). These operations should be specified in the corresponding Linux distribution configuration file.
`/etc/rc.d/init.d/vz`	The Parallels Server Bare Metal start/stop script. This script is responsible for proper Parallels Server Bare Metal startup and shutdown procedures, including modules loading and Container start/stop procedures.

Parallels Server Bare Metal Action Scripts

There might be situations when you need to do additional actions when a particular Container is started or stopped. For example, if you want to be able to access the server file system (or part of it) from Container 101, then you can bind mount it inside the Container manually from the server. However, after you restart the Container, your mount disappears, and you should manually type the `mount` command again.

Parallels Server Bare Metal allows you to automate procedures like the above by using `Parallels Server Bare Metal` *action scripts*. There are six action scripts defined in the current version of Parallels Server Bare Metal:

global `mount`	This script runs immediately after `pctl` mounts the Container private area. The Container itself is not yet running and the script is running in the server context.
`mount`	This script runs immediately after the global mount script. The Container is still not running, and the scripts is called in the server context.
`start`	After `pctl` has started a Container, it runs the Container `start` script. The script is running already in the Container context.
`stop`	This script runs before the Container is stopped, in the Container context.
`umount`	After the Container has been already stopped, the `umount` script is executed, and the script runs in the server context.
global `umount`	This script runs when `pctl` is about to dismount the Container private area. It also runs in the servercontext.

It is important to understand how `pctl` handles exit codes of action scripts. If exit code is non-zero, then `pctl` will try to undo the action for the `mount` and `start` scripts. In other words, if the `start` script returns an error, then `pctl` will stop Container, and if one of the `mount` scripts fails, then `pctl` will dismount the Container private area. Please note that in this case `pctl` will not execute the `stop` and `umount` scripts at all.

Caution: When executing `pctl start`, both `mount` and `start` scripts run. However, if the `start` script fails then neither `stop` nor `umount` scripts will run. As a result, `pctl` might be unable to dismount the Container private area, if you set up additional mounts in the `mount` scripts and dismount them in the `umount` scripts.

The situation with the `umount` and `stop` scripts is similar. If a script returns an error, then the action will not be taken. Be careful since this allows to create Containers that are not stoppable by `pctl`.

The global scripts are named `vps.mount` and `vps.umount` and located in the `/etc/vz/conf` directory on the server. These scripts are called when any Container on the server is started or stopped. So, you should include in these scripts those commands that are common for all Containers and leave Container-specific commands for the scripts belonging to a particular Container. Container-specific action scripts are located in the `/vz/private/`*`CT_ID`*`/scripts` directory and have the `mount`, `start`, `stop`, and `umount` names. For example, the scripts specific for Container 101 will have the following names:

- `/vz/private/101/scripts/mount`
- `/vz/private/101/scripts/start`

- `/vz/private/101/scripts/stop`
- `/vz/private/101/scripts/umount`

For the `mount` and `umount` scripts, the environment passed is the standard environment of the parent (i.e. `pctl`) with two additional variables: `$VEID` and `$VE_CONFFILE`. The first one holds the ID of the Container being mounted (started, stopped, dismounted), and the second one holds the full path to the Container configuration file. It is probably a bit redundant. Parallels introduced both variables for convenience. You can use the following fragment of the code in bash scripts to get access to additional Container information like `$VE_PRIVATE` or `$VE_ROOT` locations:

```
#!/bin/bash
#
# This script sources Container configuration files in the same
# order as pctl does

# if one of these files does not exist then something is
# really broken
[ -f /etc/sysconfig/vz ] || exit 1
[ -f $VE_CONFFILE ] || exit 1

# source both files. Note the order, it is important
. /etc/vz/vz.conf
. $VE_CONFFILE
```

The `start` and `stop` scripts are performed in the Container context. If these scripts call any external commands, these commands are taken from the Container itself. Also note that the `start` script runs before any Container tasks (including `init`), thus the `/proc` file system is not mounted inside the Container at this moment – therefore, applications using an information from `/proc` may be not functional.

Parallels Server Bare Metal Utilities

This section provides information on utilities that can be used to manage Parallels Server Bare Metal parameters.

prlsrvctl

General Syntax

The `prlsrvctl` command-line utility is used to perform management tasks on the Parallels server and Parallels Server Bare Metal. The tasks include getting the Parallels Server Bare Metal information, modifying its preferences, installing a license, obtaining statistics and problem reports, and some others.

Syntax

```
prlsrvctl command [options] [-l,--login user[:passwd]@server] [-v, --verbose
number]
```

Parameters

Name	Description
`command`	The name of the command to execute.
`options`	Command options. See individual commands for available options.
`-l, --login`	Connect to the remote Parallels server and execute a command on it. If this parameter is omitted, the command will be executed on the local server.
user	The name of the user used to log in to the remote server.
passwd	The user password. If the password is omitted, you will be prompted to enter it.
server	The remote server IP address or hostname.
`-v, --verbose` *`number`*	Show verbose output. The greater the *number*, the more verbose output will be produced.

Remarks

To display help, enter `prlsrvctl` on the command-line without any parameters.

Links

Legend (p. 11)

prlsrvctl info

Displays the Parallels server and Parallels Server Bare Metal configuration information.

Syntax

```
prlsrvctl info
```

Remarks

The information returned by the `info` command includes the following:

- Server ID and hostname.
- Parallels Server Bare Metal version number.
- Default directory for storing virtual machine files.
- Parallels Server Bare Metal memory limits.
- Parallels Server Bare Metal minimum allowable security level.
- Default directory for storing virtual machine backups.
- Parallels Server Bare Metal license information.
- Server hardware configuration information.
- Other miscellaneous info.

Links

General Syntax (p. 59), Legend (p. 11)

prlsrvctl install-license

Installs the Parallels Server Bare Metal license on the Parallels server.

Syntax

```
prlsrvctl install-license -k,--key key [-n,--name name] [-c,--company name]
```

Parameters

Name	Description
`-k, --key` *key*	License key.
`-n, --name` *name*	License user name.
`-c,--company` *name*	License company name.

Links

General Syntax (p. 59), Legend (p. 11)

prlsrvctl net

The `prlsrvctl net` command allows you to create and configure Parallels virtual networks. The following subsections describe how to perform individual virtual network configuration tasks.

Creating a New Virtual Network

The `prlsrvctl net add` command can be used to create a new virtual network.

Syntax

```
prlsrvctl net add vnetwork_id [-i,--ifname if] [-m,--mac mac_address]
                  [-t,--type bridged|host-only|shared]
                  [-d,--description description]
```

Parameters

Name	Description
`vnetwork_id`	A user-defined name that will identify the new virtual network.
`-i,--ifname` *if*	The name of a physical network adapter on the Parallels server to which this virtual network should be bound.
`-m,--mac` *mac_address*	The MAC address of a virtual network adapter on the Parallels server to which this virtual network should be bound.
`-t,--type` *value*	The type of the virtual network to create. Possible values are: ▪ `bridged` -- a virtual machine connected to this type of virtual network appears as an independent computer on the network. ▪ `host_only` -- a virtual machine connected to this type of virtual network can access only the Parallels server and the virtual machines connected to the same virtual network. ▪ `shared` -- a virtual machine connected to this type of virtual network uses the Parallels server network connections.
`-d,--description` *description*	A user-defined description of the virtual network.

Links

General Syntax (p. 59), Legend (p. 11)

Modifying a Virtual Network

The `prlsrvctl net set` command allows you to modify an existing virtual network.

Syntax

```
prlsrvctl net set vnetwork_id [-i,--ifname if] [-m,--mac mac_address]
                  [-t,--type bridged|host-only|shared]
                  [-d,--description description]
                  [-n, --name new_name]
```

Parameters

Name	Description
`vnetwork_id`	The name of the virtual network to modify.
`-i,--ifname` *if*	The name of a physical network adapter on the Parallels server to which this virtual network should be bound.
`-m,--mac` *mac_address*	The MAC address of a virtual network adapter on the Parallels server to which this virtual network should be bound.
`-t,--type`	The type of the virtual network to create. Possible values are: ▪ `bridged` -- a virtual machine connected to this type of virtual network appears as an independent computer on the network. ▪ `host_only` -- a virtual machine connected to this type of virtual network can access only the Parallels server and the virtual machines connected to the same virtual network. ▪ `shared` -- a virtual machine connected to this type of virtual network uses the Parallels server network connections.
`-d,--description` *description*	A user-defined description of the virtual network.
`-n, --name` *new_name*	A new name for the virtual network. Use this parameter if you would like to rename the virtual network.

Links

General Syntax (p. 59), Legend (p. 11)

Deleting a Virtual Network

The `prlsrvctl net del` command allows to delete an existing virtual network.

Syntax

```
prlsrvctl net del vnetwork_id
```

Parameters

Name	Description
`vnetwork_id`	The name of the virtual network to delete.

Links

General Syntax (p. 59), Legend (p. 11)

Listing Existing Virtual Networks

The `prlsrvctl net list` command lists the existing virtual networks.

Syntax

prlsrvctl net list

Links

General Syntax (p. 59), Legend (p. 11)

prlsrvctl set

Allows you to modify Parallels Server Bare Metal preferences.

Syntax

```
prlsrvctl set [--mem-limit auto|size]
              [-s,--min-security-level low|normal|high]
              [-c,--cep on|off]
              [--mng-settings allow|deny]
              [{--device device --assignment host|vm}]
              [--backup-storage user[[:passwd]@server[:port]]]
              [--backup-path path]
```

Parameters

Name	Description
`--mem-limit`	Sets the upper limit of the memory size that can be reserved for Parallels Server Bare Metal operation. The following options are available: ▪ `auto` -- if this option is used, the memory size will be calculated automatically. ▪ `size` -- user-defined memory size, in megabytes.
`-s,--min-security-level`	The lowest allowable security level that can be used to connect to the Parallels server. The following options are available: ▪ `low` -- plain TCP/IP (no encryption). ▪ `normal` -- most important data is sent and received using SSL over TCP/IP (user credentials during login, guest OS clipboard, etc.) Other data is sent and received using plain TCP/IP with no encryption. ▪ `high` -- all of the data is sent and received using SSL.
`-c,--cep`	Enables/disables the participation in the Customer Experience Program (CEP). The following options are available: ▪ `on` -- enables CEP. ▪ `off` -- disables CEP.
`--mng-settings`	Allows you to grant or deny permission to new users to modify Parallels Server Bare Metal preferences. By default, only administrators of the host OS can modify Parallels Server Bare Metal preferences. When a new user profile is created (this happens when a user logs in to the Parallels server for the first time), he/she will be granted or denied this privilege based on the default setting. This parameter allows you to set that default setting. Please note that this parameter only affects new users (the users that will be created in the future). The profiles of the existing users will not be modified.
`--device` *device* `--assignment`	Allows you to set the assignment mode for the specified VTd device. The following options are available:

	▪ `host` -- assign the device to the Parallels server. ▪ `vm` -- assign the device to virtual machines.
`--backup-storage`	The default backup server where virtual machine backups will be stored.
user	The name of the user on the backup server.
passwd	The user password.
server	The backup server IP address or hostname.
port	The port number. If omitted, the default port number will be used.
`--backup-path` *path*	The name and path of the default directory on the backup server where virtual machines backups will be stored.

Links

General Syntax, Legend (p. 11)

prlsrvctl shutdown

Shuts down the Parallels Server Bare Metal component responsible for managing virtual machines. No operations on virtual machines are possible.

Syntax

```
prlsrvctl shutdown [-f,--force]
```

Parameters

Name	Description
`-f, --force`	Specifies whether the shutdown operation should be forced. If one or more virtual machines are running, clients are connected, or some tasks are currently in progress, then forcing the shutdown will stop all processes automatically and will shut down the Parallels Server Bare Metal component.

Links

General Syntax, Legend (p. 11)

prlsrvctl user list

Displays the list of Parallels Server Bare Metal users (only those users are displayed that have created at least one virtual machine on the Parallels servers).

Syntax

```
prlsrvctl user list [-o,--output name[,name...]]
```

Parameters

Name	Description
`-o,--output` *name*	Names of the fields to include in the output. The following fields are available: ▪ `name` -- User name. ▪ `mng_settings` -- Indicates whether the user is allowed to modify Parallels Server Bare Metal preferences. ▪ `def_vm_home` -- The user default virtual machine folder. The fields must be specified using the lower case letters.

See Also

`prlsrvctl user set` (p. 67)

Links

General Syntax, Legend (p. 11)

prlsrvctl user set

Allows to modify the profile of a Parallels Server Bare Metal user.

Syntax

```
prlsrvctl user set name|uuid [--def-vm-home path]
                    [--mng-settings allow|deny]
```

Parameters

Name	Description
name	The user name.
uuid	The user UUID (universally unique ID).
`--def-vm-home` *path*	The default virtual machine directory name and path.
`--mng-settings`	Specifies whether the user should be allowed to modify Parallels Server Bare Metal preferences. The available options are: ▪ `allow` ▪ `deny`

See Also

`prlsrvctl user list` (p. 66)

Links

General Syntax, Legend (p. 11)

prlsrvctl statistics

Obtains Parallels Server Bare Metal statistics.

Syntax

```
prlsrvctl statistics [-a, --all] [--loop] [--filter name]
```

Parameters

Name	Description
`-a, --all`	*This parameter is not currently used.*
`--loop`	Subscribes to receive statistics on the periodic basis. Once you execute the command with this option, the statistics will be displayed in your console window every time a new set of values is collected. To unsubscribe, press the Enter key or Ctrl-C in your console window.
`--filter` *name*	*This parameter is not currently used.*

Links

General Syntax, Legend (p. 11)

prlsrvctl problem-report

Generates a problem report and displays it on the screen.

Syntax

```
prlsrvctl problem-report
```

Parameters

The command accepts no parameters.

Remarks

The command collects technical data about Parallels Server Bare Metal and the Parallels server and displays the report on the screen (the output can also be piped to a file). The report can then be directed to the Parallels technical support for analysis.

Links

General Syntax, Legend (p. 11)

vzup2date

The `vzup2date` utility is used to update your Parallels Server Bare Metal software and templates and keep them at the most recent version. It has the following syntax:

```
vzup2date [-s|-z] [-m interactive]
vzup2date [config_options] [-s|-z] -m {batch|messages} \
          <command> [command_options] [filters] [update_IDs]
vzup2date {-?|--help}
```

The `vzup2date` utility can be launched in two modes:

- In the graphical mode: in this case `vzup2date` should be used with the `-s` and `-z` switches only or without any parameters at all. You can also specify the `-m interactive` switch to explicitly indicate that `vzup2date` is to be run in the graphical mode.
- In the command-line mode containing two submodes: the batch submode and the messages submode. To run `vzup2date` in the command-line mode, you should specify either the `-m batch` switch or `-m messages` switch for executing `vzup2date` in the batch or messages submodes, respectively. Both submodes are meant to update Parallels Server Bare Metal in the unattended mode and have the identical syntax; however, they are different in their output. The batch submode output is more user friendly than the messages submode one which is mostly suitable for machine processing.

The following options can be passed to `vzup2date` in both modes - graphical and command-line:

Name	Description
`-s, --system`	Used to check and, if necessary, download and install Parallels Server Bare Metal system updates, i.e newest versions of the Parallels Server Bare Metal core and utilities. If the `-s` is omitted and the `-t` option is not specified either, the `vzup2date` utility looks for the Parallels Server Bare Metal system updates.
`-z`	Used to check and, if necessary, download and install OS and application EZ templates. You should explicitly specify this option to make `vzup2date` look for EZ template updates.

Setting Connection Parameters

If you have not set the necessary connection parameters for the repository with Parallels Server Bare Metal updates in the `/etc/sysconfig/vzup2date/vzup2date.conf` file on the server or wish to redefine any of them, you may specify the following options:

Name	Description
`-R, --repository=`*`path`*	The URL used to connect to the repository with Parallels Server Bare Metal updates. The *`path`* value should be specified in the form of `[`*`protocol`*`://][`*`user`*`:`*`password@`*`]`*`server`*`[:`*`port`*`][`*`/repository_dir`*`]` where: ▪ *`protocol`* indicates what protocol is to be used while connecting to the update server (e.g. `http` or `https`). ▪ *`user:password`* denotes the user name and password used to access the update server. ▪ *`server`* is the IP address or the domain name where the update repository is located. ▪ *`port`* denotes the port number of the update server used for establishing the connection. ▪ *`repository_dir`* specifies the directory on the update server where the required Parallels Server Bare Metal updates are stored.
`--proxy=`*`path`*	The proxy server address, if you use this server. The *`path`* value should be specified in the form of `[`*`protocol`*`://][`*`user`*`:`*`password@`*`]`*`server`*`[:`*`port`*`]` where ▪ *`protocol`* indicates what protocol to use while connecting to the proxy server (e.g. `http` or `https`). ▪ *`user:password`* denotes the user name and password used to log it to the proxy server. ▪ *`server`* is the IP address or the domain name of the proxy server. ▪ *`port`* specifies the port number of the proxy server used for establishing the connection.
`--local-path=`*`path`*	The path to the local directory on the server where the downloaded Parallels Server Bare Metal updates are stored.
`--log-path=`*`path`*	The path to the log file on the server containing the information on Parallels Server Bare Metal updates.
`--save-config`	Use this option to save the specified parameters in the `/etc/sysconfig/vzup2date/vzup2date.conf` file on the server.

Available Commands

The commands that can be used with `vzup2date` in the command-line mode (i.e. while specifying either the `-m batch` switch or the `-m messages` switch) are given in the table below:

Name	Description
`list`	Lists the updates matching the criteria specified in [*`filters`*] and [*`update_IDs`*]. Detailed information on filters and update IDs is given below. If no filters and update IDs are specified, all updates for the OS and application templates installed on the server are displayed.
`show`	Displays detailed information on the updates matching the criteria specified in [*`filters`*], and [*`update_IDs`*]. If no filters and update IDs are specified, information on updates for all OS and application templates installed on the server is shown.
`get`	Checks and downloads Parallels Server Bare Metal updates matching the criteria specified in [*`filters`*], and [*`update_IDs`*] from the Parallels update server to the local directory on the server. The path to the local directory can be set either in the `/etc/sysconfig/vzup2date/vzup2date.conf` file or by specifying the `--local-path` option. If no filters and update IDs are specified, updates for all OS and application templates installed on the server are downloaded to the local directory.
`install`	Checks and, if necessary, downloads and installs Parallels Server Bare Metal updates matching the criteria specified in [*`filters`*], and [*`update_IDs`*]. If no filters and updates IDs are specified, updates for all OS and application templates on the server are downloaded and installed. In some cases, you may need to update the `vzup2date` utility itself. To do this, pass the `--self-update` option to the `vzup2date install` command.
`showconf`	Shows the contents of the `/etc/sysconfig/vzup2date/vzup2date.conf` file on the server.
`install-self-update` *`update_ID`*	Installs updates with the specified ID for the `vzup2date` utility. You may need to update `vzup2date` before you are able to get the latest Parallels Server Bare Metal updates. To display the latest updates for `vzup2date`, you can use the `vzup2date get` command.

All the aforementioned commands (except for `showconf` and `install-self-update`) can be used with the following options:

Name	Description
`--cache`	If specified, `vzup2date` does not search the update server for the update packages that are already available in the local repository directory on the server. When used with the `vzup2date install` command, `vzup2date` does not check the integrity of the update files located in the local repository directory.
`--nosignatures`	If specified, `vzup2date` does not validate digital signatures of the downloaded update packages.

`--status-log-file=`*`path`*	The path to the status log file where the messages on Parallels Server Bare Metal updates will be stored (e.g. `/vz/vzup2date/my_file.log`). Without specifying this option, the messages are sent to `stdout` only. The option can be used in the messages submode only.
`--status-log-prog=`*`path`*	The path to the status log program. This program should accept log messages sent to `stdout`. The option can be used in the messages submode only.
`--status-log-id=`*`ID`*	The ID assigned to the status log file and unique within the given system. This ID will be used as the name of the log file with the `.log` extension created during the `vzup2date` *`command`* execution. By default, this file is located in the `/vz/vzup2date/ipc` directory on the server. The option can be used in the messages submode only.

Note: The `vzup2date install` command has a number of additional options described in the vzup2date install subsection (p. 75).

Update Filters and Update IDs

The `vzup2date` utility allows you to specify what particular Parallels Server Bare Metal updates should be searched for on the update server, download the found updates, and install them on the server. This can be done by using special update filters or by explicitly specifying the update IDs. You can also combine both methods to get the right updates for your Parallels Server Bare Metal installation.

The filters that can be used with `vzup2date` can be divided in two groups:

- The filters used to update Parallels Server Bare Metal system files. They are presented in the table below:

Name	Description
`--major`	Selects the latest major update for your current Parallels Server Bare Metal installation. To see what latest update is available, you can use the `vzup2date list` or `vzup2date show` commands. If you do not specify an update ID for the major Parallels Server Bare Metal update, your installation will be automatically updated to the latest Parallels Server Bare Metal version available on the Parallels update server. Bear in mind that the major Parallels Server Bare Metal release you are updating to might also already have available minor updates (i.e. updates for the Parallels Server Bare Metal core and tools). However, they will not be applied during the major Parallels Server Bare Metal update. Thus, to install the latest Parallels Server Bare Metal version and then to apply minor updates for it, you will need to launch the utility twice.
`--core`	Selects updates available for your current Parallels Server Bare Metal core. While working with the Parallels Server Bare Metal core updates, keep in mind the following: ▪ Each Parallels Server Bare Metal release has its own set of core updates. Therefore, the update to the latest core version is possible only within the given Parallels Server Bare Metal release. ▪ Core updates are cumulative, i.e. the updates with higher versions include the functionality of all previous core updates within the given Parallels Server Bare Metal release (e.g. the `CU-2.6.20` core update includes all functionality of `CU-2.6.18`, `CU-2.6.16`, etc.). ▪ Only the updates for the core version currently installed in your system are shown. For example, if your system is running the 2.6 core version, all core updates for 2.6 will be shown.
`--tools`	Selects updates available for your current Parallels Server Bare Metal utilities. While working with Parallels Server Bare Metal tools updates, keep in mind the following: ▪ Each Parallels Server Bare Metal release has its own set of utility updates. Therefore, the update to the latest utility version is possible only within the given Parallels Server Bare Metal release. ▪ As distinct from the Parallels Server Bare Metal core updates, utility updates are incremental, i.e. they include the new functionality only.

- The filters used to update EZ templates. These are the following filters:

Name	Description

`--update-os`	Selects updates for all OS templates installed on the server.
`--all-os`	Selects all OS templates available on the Parallels update server.
`--update-app-for=`*`OS_List`*	Selects updates for all application templates included in the specified OS templates.
`--update-app`	Selects updates for all application templates on the server.
`--all-app-for=`*`OS_List`*	Selects all application templates available on the update server for the specified OS templates.
`--all-app`	Selects all application templates for all OS templates installed on the server.
`--update`	Selects updates for all OS and application templates installed on the server.

vzup2date install

The `vzup2date install` command is used to install new OS and application templates on the server or to update any of the existing OS and application templates already installed on the server. It has the following syntax:

```
vzup2date  [config_options] [-s|-t|-z] -m {batch|messages} install\
            [options] [filters] [update_IDs]
```

Along with the options which are common for all `vzup2date` commands and described in the previous subsection, you can also use the following options with `vzup2date install`:

Name	Description
`--reboot`	Automatically reboots the server, if needed, after the Parallels Server Bare Metal update completion. For example, the system reboot may be required in the case of updating the Parallels Server Bare Metal core or installing major updates on the server. If the option is omitted, the system will not reboot.
`--loader-autoconfig` `[=bootloader]`	Automatically recognizes and reconfigures the Lilo and GRUB boot loaders after the Parallels Server Bare Metal update completion. You can explicitly specify what boot loader is to be reconfigured by specifying either `GRUB` or `Lilo` as the value of *`bootloader`*.
`--self-update`	Automatically updates the `vzup2date` utility. If an updated version of `vzup2date` is available, this version is downloaded and installed on the server at first. After that, the command is re-launched and the Parallels Server Bare Metal system update is performed.
`--novzpkgcache`	Do not run the `vzpkgcache` utility in case of standard OS templates and the `vzpkg create cache` utility in case of EZ OS templates. By default, a tarball (cache) is automatically created for every OS template or its update installed on the server by using the `vzup2date install` command.

For example, to update to the latest Parallels Server Bare Metal core within your current Parallels Server Bare Metal release and to automatically reconfigure the GRUB boot loader, you can issue the following command:

```
# vzup2date -m batch install --core --loader-autoconfig=grub
Downloading releases information. Please, wait...done
Downloading updates information for 4.0. Please, wait...done
Downloading detailed updates information. Please, wait...done
Checking downloaded packages integrity:
        [##################################################################]
100%
Downloading 16 packages
 1 kernel-doc-2.6.10-021stab028.19.777.i386.r 100%    2MB   2.1MB/s
 2 kernel-headers-2.6.10-021stab028.19.777.i3 100%    1MB   2.0MB/s
 3 vzkernel-2.6.10-021stab028.19.777.athlon.r 100%    5MB   1.9MB/s
 4 vzkernel-2.6.10-021stab028.19.777.i686.rpm 100%    6MB   2.1MB/s
 5 vzkernel-enterprise-2.6.10-021stab028.19.7 100%    6MB   2.0MB/s
 6 vzkernel-entnosplit-2.6.10-021stab028.19.7 100%    6MB   2.2MB/s
 ...
```

vzup2date-mirror

The `vzup2date-mirror` utility is used to create local mirrors of the Parallels official repository storing the latest versions of the Parallels Server Bare Metal software and OS and application templates. The `vzup2date-mirror` utility has the following syntax:

```
vzup2date-mirror [options] [local_repo_path]
```

You can pass the following options to `vzup2date-mirror`:

Name	Description
`-s, --system`	Creates a local mirror of the repository storing the latest versions of the Parallels Server Bare Metal core and utilities. It also can be used to update your existing local mirror. If this option is not specified and the `-z` option is omitted, `vzup2date-mirror` will also make the repository mirror with Parallels Server Bare Metal system files.
`-z, --eztemplates`	Creates a local mirror of the repository storing the latest versions of OS and application EZ templates.
`-c, --config`	The full path to the configuration file that will be used by `vzup2date-mirror` on the step of connecting to the Parallels official repository and downloading new updates. If omitted, the utility uses the default `vzup2date-mirror.conf` file which is located in the `/etc/vzup2date-mirror` directory.
`local_repo_path`	The path to the repository mirror. If omitted, the utility uses the repository mirror whose location is defined in the `vzup2date-mirror` configuration file. Detailed information on `vzup2date-mirror.conf` is provided in the Configuration File for vzup2date-mirror subsection (p. 35).
`-q, --quiet`	Reports only errors during the `vzup2date-mirror` execution.
`-D, --delete`	Automatically deletes obsolete updates during the `vzup2date-mirror` execution.
`--version`	Prints the utility version and exits.
`-h, --help, -?`	Displays the utility usage and exits.

When executed, the `vzup2date-mirror` utility completes a number of tasks (connects to the Parallels official repository, creates a special directory and downloads the specified Parallels Server Bare Metal system or templates updates to this directory, etc.) resulting in building a local mirror of the Parallels official repository or some of its parts.

vzlicload

This utility is used to manage Parallels Server Bare Metal licenses on your server. It has the following syntax:

```
vzlicload [options]
```

The utility accepts the following options:

`-p, --product-key`	Installs the Parallels Server Bare Metal license on the server.
`-f, --license-file <file_path>`	The full path to the license file containing the license to be installed on the server.
`-r, --remove`	Removes the license with the specified serial number from the server. You can find out the license serial number using the `vzlicview` utility (see the **vzlicview** subsection (p. 79) for details).
`-i, --stdin`	Makes `vzlicload` use standard input as a license.
`-h, --help`	Prints the usage help and exits.

vzlicupdate

This utility can be used to perform the following license-related operations:

- Activate your Parallels Server Bare Metal installation using a special activation code.
- Update the currently installed license on the server.
- Transfer the license installed on the Source Server with the help of an activation code to the Destination Server.

The `vzlicupdate` utility has the following syntax:

```
vzlicupdate [options]
```

The utility accepts the following options:

`-a, --activate` *`activation_code`*	Activates the Parallels Server Bare Metal installation using the specified activation code. To successfully complete this task, your server must be connected to the Internet.
`-t, --transfer`	Transfers the license activated with the activation code from the Source Server to the Destination Server. Should be run along with the `-a` option on the Destination Server, i.e. on the server where you are planning to transfer the license.
`-s, --server` *`hostname`*`[:`*`port`*`]`	The hostname of the Parallels Key Authentication (KA) server responsible for updating Parallels Server Bare Metal licenses, activating Parallels Server Bare Metal Containers installations, and transferring licenses from the Source Server to the Destination Server. If not specified, the `ka.parallels.com` hostname is used.
`-n, --no-check`	Updates the license currently installed on the server even if it is still valid.
`-v, --verbose`	Sets the log level to its maximum possible value.
`-h, --help`	Prints the utility usage and exits.

When executed without any options, `vzlicupdate` updates the license currently installed on the server. However, you can use the options listed in the table above to complete other license-related tasks.

vzlicview

This utility displays the license contents along with the license status information. It has the following syntax:

```
vzlicview [options]
```

The following options can be used with this utility:

`-p, --product-key <key_number>`	Displays the license information contained in the specified Parallels Server Bare Metal product key.
`-f, --license-file <file>`	Displays the license information from the specified Parallels Server Bare Metal license file.
`-i, --stdin`	Makes `vzlicview` use standard input as a license and display its information.
`-h, --help`	Displays the utility usage and exits.

When executed without any options, the utility returns the contents and status of the license currently installed on the server. The utility can report the following statuses for Parallels Server Bare Metal licenses:

`ACTIVE`	The license installed on the server is valid and active.
`VALID`	The license the utility parses is valid and can be installed on the server.
`EXPIRED`	The license has expired and, therefore, could not be installed on the server.
`GRACED`	The license has been successfully installed on the server, but it has expired and is currently on the grace period (i.e. it is active till the end of the grace period).
`INVALID`	The license is invalid (for example, because of the server architecture mismatch) or corrupted.

vznetcfg

The `vznetcfg` utility is used to manage the following network devices on the server:

- physical and VLAN (Virtual Local Area Network) adapters
- Virtual Networks (VNs)

`vznetcfg` has the following syntax:

```
vznetcfg command
```

Where *`command`* can be one of the following:

Name	Description
`net new <VN_name>`	Creates a new Virtual Network with the name of `<VN_name>` on the server.
`net addif <VN_name>|<interface_name>`	Connects a network device with the name of `<interface_name>` to the Virtual Network having the name of `<VN_name>`. You can join the following network devices to the Virtual Network: ▪ physical network interface cards (NICs) installed on the server ▪ VLAN adapters bound to NICs on the server
`net delif <interface_name>`	Disconnects a network device (either a NIC or a VLAN adapter) with the name of `<interface_name>` from the corresponding Virtual Network.
`net change <old_VN_name> <new_VN_name>`	Changes the Virtual Network name from `<old_network_ID>` to `<new_VN_name>`.
`net del <VN_name>`	Removes the Virtual Network with the name of `<VN_name>` from the server.
`vlan add <parent_interface> <index_number>`	Creates a new VLAN adapter, associates it with the VLAN ID of `<index_number>` (where `<index_number>` can be an arbitrary integer number to be used to uniquely identify the VLAN among other VLANs on the server), and ties it to the `<parent_interface>` physical network adapter on the server.
`vlan del <vlan_adapter_name>`	Removes the VLAN adapter with the name of `<vlan_adapter_name>` from the server. **Note:** A VLAN adapter name is automatically generated by Parallels Server Bare Metal on the basis of the VLAN ID and the name of the physical adapter you specified during the VLAN adapter creation (e.g. `eth0.1`). You can find out the VLAN name using the `vznetcfg if list` command.
`if list`	Lists detailed information on all network devices (NICs, VLAN adapters, etc.) available on the server.
`net list`	Displays detailed information on the Virtual Networks currently existing on the server.

`init all`	Initializes all interfaces (e.g. VLANs and bridges) listed in the `/etc/vz/vznet.conf` file on the server. You may wish to make use of this command when creating startup scripts.
`down all`	Disables all interfaces (e.g. bridges and VLANs) listed in the `/etc/vz/vznet.conf` file on the server.

vzreport

`vzreport` is used to compile a problem report and to automatically send it to the Parallels support team. It has the following syntax:

```
vzreport [options]
```

The following command-line options can be used with the `vzreport` utility:

Name	Description
`-h, --help`	Print usage information.
`-q, --quiet`	Quiet mode. Print error messages only.
`-p, --progress`	Causes `vzreport` to print additional information on its progress.
`-n, --name` *`name`*	The name of the person submitting the problem report.
`-c, --company` *`company`*	The name of the company where the person is working.
`-e, --email` *`e-mail_address`*	The e-mail address to be used to contact the person generating the problem report.
`-s, --subject` *`subject`*	The main subject of the problem report.
`-m, --description` *`problem_description`*	Additional information which, in your opinion, can help solve the problem.

When launched without any options, the `vzreport` utility starts in the full screen mode; however, you can force it to run in the command line mode by specifying an option containing either your contact information (e.g. `-n` denoting your name) or the problem report description (e.g. `-m` used to provide additional information on your problem). Moreover, if you have specified at least one option with your contact information and/or problem description, you should also indicate all the other options.

vzstatrep

`vzstatrep` is run on the Monitor Server and used to analyze the logs collected by the `vzlmond` daemon on one or more servers to generate statistic reports and graphics on the basis of the gathered logs, and to send these reports and graphics to the server administrator's e-mail address(es). The `vzstatrep` utility has the following syntax:

```
vzstatrep [options]
```

The following command-line options can be passed to `vzstatrep`:

Name	Description
`--plot`	Generate graphics for the resources parameters specified as the values of the `STATS_PLOT` parameter in the `/etc/vzstatrep.conf` file on the Monitor Server.
`--sendmail`	Send the statistic report and graphics to the e-mail address(es) specified as the value(s) of the `STATS_EMAIL` parameter in the `/etc/vzstatrep.conf` file on the Monitor Server. If the `--sendmailto` option is omitted, you should obligatorily use this option.
`--sendmailto` *`mail`*	Send the statistic report and graphics to the e-mail address specified as the value of this option. You can set several e-mail addresses and separate them by spaces. If the `--sendmail` option is omitted, you should obligatorily use this option.
`--weekly`	Generate statistic reports and graphics on a weekly basis. By default, `vzstatrep` analyzes the logs and produces the server resources statistics once a day.
`--nodes` *`hostname`*	Analyze the logs from the server whose IP address or hostname is specified as the value of this option. You can set several servers by separating them by spaces and enclosing them in quotes (e.g. `"my_hardware_node1 my_hardware_node2"`). If the option is omitted or its value is not specified, the logs from the servers set as the values of the `NODES` parameter in the `/etc/vzstatrep.conf` file on the Monitor Server are analyzed.

The `vzstatrep` utility generates statistic reports and graphics on the basis of the logs gathered by `vzlmond` (by default, the logs are stored in the `/var/log/vzstat` directory on the server) and containing information on the memory and CPU consumption of the server, network resources on the server, etc. You do not need to perform any additional operations to start using `vzstatrep`. All the necessary parameters can be set during the `vzstatrep` execution by using the aforementioned options. However, if you wish to run the `vzstatrep` utility as a `cron` job and/or free yourself from the necessity to manually specify the needed options each time you wish to run `vzstatrep`, you should edit the `/etc/vzstatrep.conf` configuration file on the Monitor Server and set the parameters values contained in this file. Detailed information on the `/etc/vzstatrep.conf` file is provided in the **vzstatrep Configuration File** subsection (p. 49).

CHAPTER 3

Managing Containers

Parallels Containers can be managed using the `pctl` command-line utility. The utility is installed on the Parallels server during the product installation.

In This Chapter

Matrix of Parallels Server Bare Metal Command-Line Utilities

The table below contains the full list of Parallels Server Bare Metal command-line utilities.

General utilities are intended for performing day-to-day maintenance tasks:

`pctl`	Utility to control Containers.
`vzlist`	Utility to view a list of Containers existing on the server with additional information.
`vzquota`	Utility to control Parallels Server Bare Metal disk quotas.

Container migration tools allow to migrate Containers between servers or within one server:

`vzmigrate`	Utility for migrating Containers from one server to another.
`vzmlocal`	Utility for the local cloning or moving of the Containers.
`vzp2v`	Utility for migrating physical servers to Containers on the Parallels server.
`pmigrate`	Utility for migrating physical servers to Containers and for moving Containers between Parallels servers.

Container backup utilities allow to back up and restore the Container private areas, configuration files, action scripts, and quota information:

`pbackup`	Utility to back up Containers.
`prestore`	Utility to restore backed up Containers.

Template management tools allow the template creation, maintenance and installation of applications into a Container:

`vzpkg`	Utility to manage OS and application EZ templates either inside your Containers or on the server itself.
`vzmktmpl`	Utility to create OS and application EZ templates.
`vzpkgproxy`	Utility to create caching proxy servers for handling OS and application EZ templates.
`vzrhnproxy`	Utility to create RHN proxy servers for handling the packages included in the RHEL 4 and RHEL 5 OS EZ templates.

Supplementary tools perform a number of miscellaneous tasks in the Parallels server and Container context:

`vzfsutil`	Utility for the VZFS optimization and consistency checking.
`vzcache`	Utility to gain extra disk space by caching the files identical in different Containers.
`vzps` `vztop`	Utilities working as the standard `ps` and `top` utilities, with Container-related functionality added.
`vzsetxinetd`	Utility to switch some services between a standalone and `xinetd`-dependent modes.
`vzdqcheck`	Print file space current usage from quota's point of view.

`vzdqdump` `vzdqload`	Utilities to dump the Container user/group quota limits and grace times from the kernel or the quota file or for loading them to a quota file.
`vznetstat`	Utility that prints network traffic usage statistic by Containers.
`vzcpucheck`	Utility for checking CPU utilization by Containers.
`vzmemcheck`	Utility for checking the server and Container current memory parameters.
`vzcalc`	Utility to calculate resource usage by a Container.
`vzcheckovr`	Utility to check the current system overcommitment and safety of the total resource control settings.
`vzstat`	Utility to monitor the server and Container resources consumption in real time.
`vzpid`	Utility that prints Container id the process belongs to.
`vzsplit`	Utility to generate Container configuration file sample, “splitting” the server into equal parts.
`vzcfgscale`	Utility to scale the Container configuration.
`vzcfgvalidate`	Utility to validate Container configuration file correctness.
`vzhwcalc`	Utility to scan the main resources on any Linux server and to save the obtained information to a special file.
`vzmtemplate`	Utility to migrate the installed OS and application templates from the one server to another.

pctl

`pctl` is the primary tool for Container management. To use it, you have to log in to the server as the root user. The syntax of `pctl` is:

```
pctl [--quiet | --verbose] command CT_ID
pctl --version
pctl --help
```

Where *`command`* can be one of the following:

`create`	Used to create Containers.
`delete`	Used to remove a Container.
`destroy`	Like `pctl delete`, this command is also used to remove a Container from the server.
`mount`	Allows mounting the Container private area and executing the Container mount script.
`umount`	Allows dismounting the Container private area and executing the unmount script.
`start`	Starts a Container.
`stop`	Stops a Container.
`restart`	Restarts a Container.
`status`	Displays the Container status.
`set`	Used to set Container parameters, including resource control settings, the location of the Container private area, hostname, IP addresses, and Container root user password.
`unset`	Used to remove Container parameters (resource control settings, IP addresses, etc.) from the configuration file.
`enter`	Provides a way for the server administrator to “enter” a Container without knowing the Container root password. Use this command with caution and never run it on un-trusted Containers.
`exec, exec2`	These two commands allow running arbitrary commands inside a Container without logging in to the corresponding Container. The difference between two is the returned status.
`recover`	Recovers the original state of the Container system and application files in case something has been broken. The user files are left intact.
`quotaon`	Turns the disk quota on for the given Container.
`quotaoff`	Turns the disk quota off for the given Container.
`quotainit`	Initializes the disk quota for the given Container with the parameters taken from the Container configuration file.
`runscript`	Runs shell scripts inside the given Container.
`suspend`	Used to save the state of a running Container in a dump file.
`restore`	Used to restore a Container from its dump file.

Verbosity options can be used with any of the above commands and they are:

`--verbose`	Overrides the `LOG_LEVEL` setting from the Parallels Server Bare Metal global configuration file `/etc/vz/vz.conf` and sets log level to the maximum possible value for this `pctl` session.
`--quiet`	Disables logging to the screen and to the log file.

You can also pass to `pctl` one of the following options:

`--version`	Displays the `pctl` package version currently installed on the server.
`--help`	Displays the usage information on `pctl`.

pctl create

This command is used to create a new Container. It has the following syntax:

```
pctl create <CT_ID> {--pkgset name [--pkgver ver] |
             [--ostemplate name]} [--config name]
             [--private path] [--root path]
             [--name CT_name] [--description CT_desc]
```

With this command, you can create regular Containers. Container ID <*CT_ID*> is required for this command and shall be unique for the server.

Note: Container IDs from 1 to 100 are reserved for internal Parallels Server Bare Metal needs. Do not use IDs from 1 to 100 for your Containers.

Command arguments are as follows:

`--pkgset` *name*	Denotes the package set (OS template) to use when creating the Container. If omitted, this value is taken from the global Parallels Server Bare Metal configuration file.
`--pkgver` *ver*	Optional. Specifies a particular version of OS template (OS template update) to use when creating the Container. If omitted, the latest available version is used.
`--ostemplate` *name*	Denotes the OS EZ template to be used for creating the Container. If omitted, this value is taken from the `DEF_OSTEMPLATE` parameter in the global Parallels Server Bare Metal configuration file.
`--config` *name*	Optional. If this option is given, `pctl` copies the values from the sample Container configuration file located in `/etc/vz/conf` and having the name in the form of `ve-<`*name*`>.conf-sample`. The sample configuration files usually have a number of resource control limits for the Container and some application templates to be added to the Container immediately upon its creation. If you skip this option and the default configuration file name is not specified in the global Parallels Server Bare Metal configuration file, you will have to set resource control parameters for the Container by using the `pctl set` command before you are able to start the Container. The application templates will have to be also added manually.
`--private` *path*	Optional. When used specifies path to the Container private area. This option is used to override default path to private area from the `/etc/vz/vz.conf` configuration file (`VE_PRIVATE` variable). The argument can contain `$VEID` string which will be replaced by numeric Container ID value.
`--root` *path*	Optional. When used specifies path to the mount point of the Container root directory. This option is used to override default path to Container root directory from the `/etc/vz/vz.conf` configuration file (`VE_ROOT` variable). The argument can contain `$VEID` string which will be replaced by numeric Container ID value.
`--skip_app_templates`	Optional. If any application templates are to be installed into the Container upon creation according to the Container sample configuration file, this switch tells the `pctl` utility not to install any templates.

When creating a new Container, you should specify a unique ID for it. There are no restrictions besides uniqueness from the `pctl` standpoint. However, it is advisable to assign different ID ranges to servers in multi-server environments. For example, you can use IDs from 101 to 2000 on the first server, IDs from 2001 to 4000 on the second one and so on. This will help you in tracking down the server where Container was created and will eliminate possibility of Container IDs conflicts when migrating Containers between servers.

pctl delete and pctl destroy

These commands are used to delete a Container, which is no longer needed, from the server. The syntax of the commands is as follows:

```
pctl delete <CT_ID>
pctl destroy <CT_ID>
```

When executed, `pctl delete`/pctl `destroy` physically removes all the files located in the Container private area (specified as the `VE_PRIVATE` variable in the Container configuration file) and renames the Container configuration file in `/etc/vz/conf` from <*CT_ID*>`.conf` to <*CT_ID*>`.conf.destroyed`. It also renames Container action scripts, if any, in a similar manner.

These commands do not take any additional arguments and requires the Container to be stopped and its private area to be dismounted.

pctl start, pctl stop, pctl restart, and pctl status

These four commands have the same syntax and take no obligatory arguments:

```
pctl start   <CT_ID> [--wait]
pctl stop    <CT_ID> [--fast]
pctl restart <CT_ID>
pctl status  <CT_ID>
```

The first command is used to start a Container. It will set up all network interfaces inside the Container, initialize the Container quota, if needed, start the `init` process inside the Container, and exit. You can also make the `pctl start` command wait for all the necessary startup processes to complete and the Container to boot into the default runlevel by passing the `--wait` option to this command.

When starting a Container, `pctl` executes a number of helper scripts located in the `/vz/private/<CT_ID>/scripts` (the first and last scripts in the table) and `/etc/vz/conf` (all the other scripts in the table) directories, namely (in the order of execution):

`mount`	Optional Container mount script. If it exists, then it is executed immediately after mounting the Container private area. If it exits with a non-zero status, then `pctl` dismounts the Container private area and returns the error.
`vz-start`	This script sets up IP traffic accounting for the Container.
`vz-net_add`	This script creates the necessary ARP entries and sets up the necessary routing entries for Container IP addresses.
`ve-alias_add`	This script configures the network interfaces inside the Container.
`ve-veconfig`	This script is called by `pctl` to set a hostname and DNS search domains inside the Container.
`ve-quota`	If the second-level (per user/group) quota is turned on, then `pctl` calls this script to form the correct `/etc/mtab` file inside the Container.
`start`	Optional Container start script. If it exists, then it is executed in the context of a just started Container.

`pctl stop` shuts the Container down. If the Container is not down after a two-minute timeout due to an error in an application, for example, `pctl` will forcibly kill all the processes inside the Container. To avoid waiting for two minutes in case of a corrupted Container, you may use the `--fast` option with this command. The normal shutdown sequence of `pctl stop` is described below in the order of execution:

`stop`	Optional Container stop script. If it exists, then it is executed in the context of the Container prior to any other actions. If it exits with non-zero status, then `pctl` does not stop the Container.
`umount`	Optional Container unmount script. If it exists, then it is executed after stopping the Container but before dismounting its private area.
`vz-stop`	This script deletes routing and IP traffic accounting for the Container.

You should use action scripts (`mount`/`umount` and `start`/`stop`) if you would like to carry out some actions upon the Container startup/shutdown. However, there might be situations when you have to modify other scripts documented above. In this case it is strongly suggested that you create a separate script containing all your modifications and add an invocation of this script to shipped scripts. This will facilitate upgrades to future Parallels Server Bare Metal versions.

The `pctl restart` *`<CT_ID>`* command consecutively performs the stopping and starting of the corresponding Container.

The `pctl status` command shows the current Container state. It outputs the following information: whether the Container private area exists, whether it is mounted and whether the Container is running as in the example below:

```
# pctl status 101
VEID 101 exist mounted running
```

pctl mount and pctl umount

These commands take no additional arguments:

```
pctl mount  <CT_ID>
pctl umount <CT_ID>
```

The first command mounts the Container private area to the Container root directory (`/vz/root/<CT_ID>` on the server) without starting it. Normally, you do not have to use this command as the `pctl start` command mounts the Container private area automatically.

The `pctl umount` command unmounts the Container private area. Usually, there is no need in using this command either because `pctl stop` unmounts the Container private area automatically.

pctl set

This command is used for setting Container parameters. It has the following syntax:

```
pctl set <CT_ID> <setting_name> <value> [--save]
```

An optional `--save` switch tells `pctl` whether to save changes into the Container configuration file `/etc/vz/conf/<CT_ID>.conf`. Practically all Container settings can be changed dynamically without the necessity of Container reboot. The exceptions are `--onboot`, `--quotaugidnum`, `--capability`, `--private`, and `--root`.

The settings specified in this file can be subdivided into the following categories: miscellaneous, networking, and resource management parameters.

Note: In Parallels Server Bare Metal, you can also use the `pctl set` command to specify a number of parameters for the server itself. Currently, these parameters include: `--cpuunits`, `--numproc`, `--numtcpsock`, `--numothersock`, `--vmguarpages`, `--kmemsize`, `--tcpsndbuf`, `--tcprcvbuf`,`--othersockbuf`, `--dgramrcvbuf`, `--oomguarpages`, `--lockedpages`, `--shmpages`, `--privvmpages`, `--numfile`, `--numflock`, `--numpty`, `--numsiginfo`, and `--dcachesize`. Any of these parameters can be set by indicating 0 as the value of `<CT_ID>`.

Miscellaneous settings:

`--onboot yes|no`	This setting requires the `--save` switch. If you set it to “yes” than Parallels Server Bare Metal will automatically start this Container on next system startup. **Note:** If "yes" is specified as the value of this parameter in the `0.conf` file, all server system management parameters are set on the server boot to the values indicated in this file.
`--offline_management yes|no`	Enabling/disabling the direct managing of the Container through a common Internet browser by means of Parallels Power Panels and the Plesk control panel (as defined by the `OFFLINE_SERVICE` parameter in the global or Container configuration file).
`--offline_service service_name`	Defines whether the Container can be managed by means of Parallels Power Panel or Plesk or both. Valid only if the `OFFLINE_MANAGEMENT` parameter is set to "yes". The names of the available services can be taken from the file names (excluding the `.conf` extension) in the `/etc/vzredirect.d` directory on the server.
`--userpasswd user:password`	This setting creates a new user with the specified password in the Container, or changes the password of an already existing user. This command modifies not the Container configuration file, but the `/etc/passwd` and `/etc/shadow` files inside the Container. In case the Container root is not mounted, it is automatically mounted to apply the changes and

	then unmounted.
`--noatime yes|no`	Sets the `noatime` flag (do not update inode access times) on the Container file system. The default is `yes` for a Class 1 Container, and `no` otherwise.
`--devnodes device:r|w|rw|none`	Lets the Container access the specified devices in the specified mode - read-only, write-only, or read-write - or denies any access. For example: `--devnodes hda1:rw` The device must be present in the Container `/dev` directory, otherwise, a new device is automatically created.
`--netdev_add` *`name`*	Moves the specified network device from the server to the given Container. For example: `--netdev_add eth0`
`--netdev_del` *`name`*	Moves the specified network device from the given Container to the server.
`--capability cap:on|off`	Specifies capabilities inside of Container. Setting the following capabilities is allowed: `AC_OVERRIDE`, `AC_READ_SEARCH`, `CHOWN`, `FOWNER`, `FSETID`, `IPC_LOCK`, `IPC_OWNER`, `KILL`, `LEASE`, `LINUX_IMMUTABLE`, `MKNOD`, `NET_ADMIN`, `NET_BIND_SERVICE`, `NET_BROADCAST`, `NET_RAW`, `SETGID`, `SETPCAP`, `SETUID`, `SYS_ADMIN`, `SYS_BOOT`, `SYS_CHROOT`, `SYS_MODULE`, `SYS_NICE`, `SYS_PACCT`, `SYS_PTRACE`, `SYS_RAWIO`, `SYS_RESOURCE`, `SYS_TIME`, `SYS_TTY_CONFIG`.
`--root` *`path`*	This setting does NOT move the root mount point of your Container to a new path. It simply overrides the `VE_ROOT` parameter in the Container configuration file.
`--private` *`path`*	This setting does NOT move the private area of your Container to a new path. It simply overrides the `VE_PRIVATE` parameter in the Container configuration file. You should use this option only if you have manually moved the Container private area to a new place and want to update the Container configuration file.
`--setmode restart|ignore`	This option tells the utility either to restart or not restart the Container after applying any parameters requiring that the Container be rebooted for them to take effect.
`--disabled yes|no`	If set to `yes`, disables the Container making it impossible to start the Container once it was stopped. The disabled Container can be started by passing the `--force` option to `pctl set`.
`--name`	An arbitrary name assigned to the Container. This name can be used, along with the Container ID, to refer to the Container while performing certain Container-related operations on the server. Follow the following rules while specifying the Container name:

	▪ The name should contain the `A-Z`, `a-z`, `0-9`, `\`, `-`, and _ symbols only. ▪ If the name consists of two or more words, it should be quoted (e.g. "My Container 101").
`--description`	This option allows you to set the description for the Container. **Note:** You are allowed to use only symbols in the 'A -z' and '0-9' ranges in your descriptions.
`--bindmount_add [`*`src`*`:]`*`dst`*`[,nosuid,noexec,nodev]`	Mounts a source directory (*`src`*) located on the server to a destination directory (*`dst`*) inside the Container. If the source directory is not specified, mounts the directory to the `/vz/root/`*`CT_ID`* directory. Additional options that can be used with `--bindmount_add` are the following: ▪ `noexec`. Do not allow execution of any binaries on the mounted directory. ▪ `nodev`. Do not interpret character or block special devices on the mounted directory. ▪ `nosuid`. Do not allow set-user-identifier or set-group-identifier bits to take effect.
`--bindmount_del `*`dst`*`|all`	Removes the mount point created by using the `--bindmount_add` option from the Container.

Resource management settings control the amount of resources a Container can consume. If the setting has `bar:lim` after it than this setting requires specifying both barrier and limit values separated by colons.

`--applyconfig `*`name`*	This option lets you set the resource parameters for the Container not one by one, but by reading them from the Container sample configuration file. All Container sample configuration files are located in the `/etc/vz/conf` directory and are named according to the following pattern: `ve-<`*`name`*`>.conf-sample`, so you should specify only the `<`*`name`*`>` part of the corresponding sample name after the `--applyconfig` option. Note that the names of sample configuration files cannot contain spaces. The `--applyconfig` option applies all the parameters from the specified sample file to the given Container, except for the `OSTEMPLATE`, `TEMPLATES`, `VE_ROOT`, `VE_PRIVATE`, `HOSTNAME`, `IP_ADDRESS`, `TEMPLATE`, `NETIF` parameters (if they exist in the configuration sample file).
`--numproc `*`bar`*`:`*`lim`*	Number of processes and threads allowed. Upon hitting this limit, the Container will not be able to start new process or thread. In this version of Parallels Server Bare Metal, the limit shall be set to the same value as the barrier.
`--numtcpsock `*`bar`*`:`*`lim`*	Number of TCP sockets (`PF_INET` family, `SOCK_STREAM` type). This parameter limits the number

	of TCP connections and, thus, the number of clients the server application can handle in parallel. In this version of Parallels Server Bare Metal, the limit shall be set to the same value as the barrier.
`--numothersock` *`bar`*`:`*`lim`*	Number of socket other than TCP. Local (UNIX-domain) sockets are used for communications inside the system. UDP sockets are used for Domain Name Service (DNS) queries, for example. In this version of Parallels Server Bare Metal, the limit shall be set to the same value as the barrier.
`--vmguarpages` *`bar`*`:`*`lim`*	Memory allocation guarantee, in pages (one page is 4 Kb). Applications are guaranteed to be able to allocate memory while the amount of memory accounted as `privvmpages` does not exceed the configured barrier of the `vmguarpages` parameter. Above the barrier, memory allocation may fail in case of overall memory shortage. In this version of Parallels Server Bare Metal, the limit shall be set to the same value as the barrier.
`--kmemsize` *`bar`*`:`*`lim`*	Size of unswappable kernel memory (in bytes), allocated for internal kernel structures of the processes of a particular Container. Typical amounts of kernel memory are 16…50 Kb per process.
`--tcpsndbuf` *`bar`*`:`*`lim`*	Total size (in bytes) of send buffers for TCP sockets – amount of kernel memory allocated for data sent from an application to a TCP socket, but not acknowledged by the remote side yet.
`--tcprcvbuf` *`bar`*`:`*`lim`*	Total size (in bytes) of receive buffers for TCP sockets. Amount of kernel memory received from the remote side but not read by the local application yet.
`--othersockbuf` *`bar`*`:`*`lim`*	Total size in bytes of UNIX-domain socket buffers, UDP and other datagram protocol send buffers.
`--dgramrcvbuf` *`bar`*`:`*`lim`*	Total size in bytes of receive buffers of UDP and other datagram protocols.
`--oomguarpages` *`bar`*`:`*`lim`*	Out-of-memory guarantee, in 4 Kb pages. Any Container process will not be killed even in case of heavy memory shortage if the current memory consumption (including both physical memory and swap) does not reach the `oomguarpages` barrier. In this version of Parallels Server Bare Metal, the limit shall be set to the same value as the barrier.
`--lockedpages` *`bar`*`:`*`lim`*	Memory not allowed to be swapped out (locked with the `mlock()` system call), in 4-Kb pages.
`--shmpages` *`bar`*`:`*`lim`*	Total size of shared memory (including IPC, shared anonymous mappings and `tmpfs` objects), allocated by processes of a particular Container, in 4 Kb pages.
`--privvmpages` *`bar`*`:`*`lim`*	Size in 4 Kb pages of private (or potentially private) memory, allocated by Container applications. Memory that is always shared among different applications is not included in this resource parameter.
`--numfile` *`bar`*`:`*`lim`*	Number of files opened by all Container processes. In this version of Parallels Server Bare Metal, the limit shall be set to the same value as the barrier.

`--numflock` *bar*:*lim*	Number of file locks created by all Container processes.
`--numpty` *bar*:*lim*	Number of pseudo-terminals. For example, `ssh` session, `screen`, `xterm` application consumes pseudo-terminal resource. In this version of Parallels Server Bare Metal, the limit shall be set to the same value as the barrier.
`--numsiginfo` *bar*:*lim*	Number of `siginfo` structures (essentially this parameter limits size of signal delivery queue). In this version of Parallels Server Bare Metal, the limit shall be set to the same value as the barrier.
`--dcachesize` *bar*:*lim*	Total size in bytes of `dentry` and `inode` structures locked in memory. Exists as a separate parameter to impose a limit causing file operations to sense memory shortage and return an error to applications, protecting from excessive consumption of memory due to intensive file system operations.
`--cpuunits` *units*	Allowed CPU power. This is a positive integer number, which determines the minimal guaranteed share of the CPU the Container will receive. You may estimate this share as ((VE CPUUNITS)/(Sum of CPU UNITS across all busy Containers))*100%. The total CPU power depends on CPU, and Parallels Server Bare Metal reporting tools consider one 1 GHz PIII Intel processor to be equivalent to 50,000 CPU units.
`--cpulimit` *percent*	This is a positive number indicating the CPU time, in percent, the corresponding Container is not allowed to exceed.
`--cpus` *num*	If the server has more than one CPU installed, this option allows you to set the number of virtual CPUs to be available to the Container.
`--diskspace` *bar*:*lim*	Total size of disk space consumed by Container, in 1 Kb blocks. When the space used by a Container hits the barrier, the Container can allocate additional disk space up to the limit during grace period specified by the `--quotatime` setting.
`--diskinodes` *bar*:*lim*	Total number of disk inodes (files, directories, symbolic links) a Container can allocate. When the number of inodes used by a Container hits the barrier, the Container can create additional file entries up to the limit during grace period specified by the `--quotatime` setting.
`--quotatime` *seconds*	The grace period of the disk quota. It is defined in seconds. A Container is allowed to temporary exceed barrier values for disk space and disk inodes limits for not more than the period specified with this setting. Specifying `-1` as the value of this setting makes the grace period last 'infinitely'.
`--quotaugidlimit` *num*	This parameter defines the maximum aggregate number of user IDs and group IDs for which disk quota inside the given Container will be accounted. If set to `0`, the UID and GID quota will be disabled. When managing the quotaugidlimit parameter, please keep in mind the following:

- Enabling per-user and per-group quotas for a Container requires restarting the Container.
- If you delete a registered user but some files with their ID continue residing inside your Container, the current number of ugids (user and group identities) inside the Container will not decrease.
- If you copy an archive containing files with user and group IDs not registered inside your Container, the number of ugids inside the Container will increase by the number of these new IDs.

`--ioprio` *`num`*	The Container priority for disk I/O operations. The allowed range of values is 0-7. The greater the priority, the more time the Container has for writing to and reading from the disk. The default Container priority is 4.
`--rate` *`dev`*`:`*`class`*`:Kbits`	If traffic shaping is turned on, then this parameter specifies bandwidth guarantee for the Container. The format is `dev:class:Kbits` where `dev` is the network device to count traffic on, `class` is the network class (group of IP addresses) and the last parameter is traffic bandwidth.
`--ratebound yes|no`	If set to "yes" then the bandwidth guarantee is also the limit for the Container and the Container can not borrow the bandwidth from the `TOTALRATE` bandwidth pool.
`--slmmode ubc|slm|all`	Defines the behaviour of the SLM and UBC parameters in respect of the given Container: ▪ if set to `ubc`, disables the SLM mode for the Container specified, i.e. the `slmmemorylimit` parameter is not supported. So, you can use only the UBC parameters to control the amount of memory which can be consumed by the Container. ▪ if set to `slm`, enables the SLM mode for the Container specified, i.e. the `slmmemorylimit` parameter is supported and can be used to manage the amount of memory which can be consumed by the Container. ▪ if set to `all`, both the `slmmemorylimit` and UBC parameters are supported and can be used to manage the amount of memory which can be consumed by the Container specified. By default, the value of this parameter is set to `all`.
`--slmpattern` *`file_id`*	Sets the SLM pattern rules for the applications grouping from the file specified. By default, the rules set in the `/etc/vzslm.d/default.conf` file on the server are used. This option, if specified, redefines the `SLMPATTERN` parameter set in the global Parallels Server Bare Metal configuration file.
`--slmmemorylimit` *`num`*	The amount of memory that can be consumed by the Container. This parameter unites all memory-related parameters for the given Container and can be viewed with the `top` and `free` utilities from inside the

Container (the parameter value will be shown as the total amount of RAM). It has effect only if the `--slmmode` parameter is set to `slm` or `all`.

The `slmmemorylimit` parameter can be set in different measurement units:

- `bytes`: this measurement unit is used by default;
- `kilobytes`: in this case the parameter value should end up in the `K` suffix (e.g. `1000K`);
- `4 Kb pages`: in this case the parameter value should end up in the `P` suffix (e.g. `150P`);
- `megabytes`: in this case the parameter value should end up in the `M` suffix (e.g. `100M`);
- `gigabytes`: in this case the parameter value should end up in the `G` suffix (e.g. `1G`).

Note: The `slmmemorylimit` parameter is supported only for Containers running the Linux distributions with the 2.6 kernel.

`--meminfo none|pages:`*`num`*`|privvmpages:`*`num`*

Customizes the output of the `/proc/meminfo` virtual file inside the Container and sets it to one of the following modes:

- *Non-virtualized* (`--meminfo none`). In this case running the `cat /proc/meminfo` command inside the Container will display the information about physical memory on the server (total, used, free, shared, etc.), in kilobytes.
- *Virtualized in pages* (`--meminfo pages:`*`num`*). Setting the `/proc/meminfo` output to this mode allows you to manually specify the amount of total memory to be displayed while running the `cat /proc/meminfo` command inside the Container.
- *Virtualized in privvmpages* (`--meminfo privvmpages:`*`num`*). Setting the `/proc/meminfo` output to this mode also allows you to arbitrarily specify the amount of total memory to be displayed while running the `cat /proc/meminfo` command inside the Container. As distinct from the previous mode, the amount of memory shown in this mode is calculated on the basis of the value of the `PRIVVMPAGES` parameter set in the Container configuration file.

`--reset_ub`

Resets the current values of all system parameters of the server to the ones set in the `0.conf` file.

Network related settings allow you to set the hostname, the domain to search when a not fully qualified domain name is used, the DNS server address and the IP addresses that Container can use as well as to indicate those `iptables` modules that can be loaded to the Container:

`--hostname` *`name`*

Sets the hostname to the specified name.

`--ipadd` *`addr`*	Adds an IP address to a list of IP addresses the Container can use and brings up the network interface with this address inside the Container. If used with the `--ifname` option, adds an IP address to the specified Container virtual network adapter.
`--ipdel` *`addr`*`|all`	Allows you to revoke IP address from the Container. If “all” is used instead of IP address than all IP addresses will be revoked. If used with the `--ifname` option, deletes an IP address from the specified Container virtual network adapter.
`--nameserver` *`addr`*	The DNS server IP address for the Container. More than one server may be specified in space-separated format. If used with the `--ifname` option, sets the DNS server for the specified Container virtual network adapter.
`--searchdomain` *`domain`*	The DNS search domain for the Container. More than one domain may be specified.
`--iptables` *`module`*	Only those `iptables` modules will be loaded to the given Container which are indicated. The list of `iptables` modules are loaded to a Container is determined by the list of `iptables` modules loaded on the server at the moment of the Container startup.
`--netif_add` *`name`* `[,`*`mac`*`,`*`host_mac`*`]`	Creates a new `veth` virtual network adapter and assigns the name of *`name`* to the Ethernet interface inside the Container. Along with the Ethernet interface name inside the Container, you can set the following parameters when creating the `veth` adapter: ▪ *`mac`*: the MAC address to be assigned to the `veth` Ethernet interface inside the Container. ▪ *`host_mac`*: the MAC address to be assigned to the `veth` Ethernet interface on the server. Only the Ethernet interface name (*`name`*) is mandatory; all the other parameters, if not specified, are automatically generated by Parallels Server Bare Metal during the `veth` adapter creation.
`--netif_del` *`name`*	Removes the `veth` virtual network adapter with the specified name from the Container.
`--ifname` *`name`*	Specifies the name of the `veth` virtual network adapter whose settings are to be configured. This option can be used along with one of the following options: `--ipadd`, `--ipdel`, `--nameserver`, `--gw`, `--network`, `--dhcp`, `--mac`, `--host_mac`.
`--mac` *`MAC_Address`*	The MAC address to be assigned to the `veth` virtual Ethernet interface inside the Container. Should be used along with the `--ifname` option.
`--host_mac` *`MAC_Address`*	The MAC address to be assigned to the `veth` virtual Ethernet interface on the server. Should be used along with the `--ifname` option.

`--host_ifname` *`name`*	The name to be assigned to the `veth` virtual Ethernet interface on the server. Should be used along with the `--ifname` option.
`--network` *`network_ID`*	Connects the `veth` virtual network adapter to the bridge associated with the specified network ID. Should be used along with the `--ifname` option. You can also use this option to disconnect the `veth` virtual network adapter from the bridge. To this effect, you should specify `""` after the option.
`--dhcp yes|no`	Defines the IP assignment type for the `veth` virtual network adapter: ▪ `yes` enables the dynamic IP address allocation for the Container. ▪ `no` turns off the dynamic IP address allocation for the Container. Should be used along with the `--ifname` option.
`--gw` *`addr`*	Set the default gateway for the `veth` virtual network adapter. Should be used along with the `--ifname` option.

pctl unset

This command is used to remove Container parameters from its configuration file (`/etc/vz/conf/<CT_ID>.conf`). It has the following syntax:

```
pctl unset <CT_ID> <setting_name> --save
```

Depending on the parameter for which the command is executed, `pctl unset` can:

- Either delete the information on the specified parameter from the Container configuration file without making any changes to the Container configuration (e.g. if executed with the `--root` or `--private` parameter).
- Or delete the information on the specified parameter from the Container configuration file and make the corresponding changes to the Container configuration (e.g. disable the offline management if executed with the `--offline_management` parameter or forbid the Container to start on the server boot if executed with the `--onboot` parameter).

This command can be used with the same parameters as `pctl set`. You can view detailed information on all the parameters in the previous subsection.

pctl exec, pctl exec2, and pctl enter

These commands are used to run arbitrary commands inside a Container being authenticated as root on the server. The syntax of these commands is as follows:

```
pctl { exec, exec2 } <CT_ID> <command>
pctl enter <CT_ID>
```

where `command` is a string to be executed in the Container. If `command` is specified as "`-`" then the commands for execution will be read from the standard input until the end of file or "exit" is encountered.

The difference between `exec` and `exec2` is the exit code. `pctl exec` returns 0 in case `pctl` has been able to launch the command and does not take into account the exit code of the command itself. `pctl exec2` returns the exit code of the command executed in the Container.

When using `exec` or `exec2`, you should remember that the shell parses the command-line and, if your command has shell meta-characters in it, you should escape or quote them.

`pctl enter` is similar to `pctl exec /bin/bash`. The difference between the two is that `pctl enter` makes the shell interpreter believe that it is connected to a terminal. As such, you receive a shell prompt and are able to execute multiple commands as if you were logged in to the Container.

However, be aware that `pctl enter` is a potentially dangerous command if you have untrusted users inside the Container. Your shell will have its file descriptors accessible for the Container root in the `/proc` filesystem and a malicious user could run `ioctl` calls on it. Never use `pctl enter` for Containers you do not trust. That is why, `pctl enter` is only supposed to be an off-duty way of connecting to Containers, not a complete replacement of `ssh`. Therefore, it has certain limitations, for example, you cannot establish `ssh` connections while being connected to a Container through `pctl enter`.

pctl recover and pctl reinstall

These commands are used to restore the original state of the Container system and application files (to be more precise, of the VZFS symlinks of the Container private area to system and application templates) in case the Container administrator has inadvertently tampered with them and thereby broken something. These symlinks are restored to the state as they were at the time when the Container was created and/or when other applications were added to the Container afterwards.

The difference between these two commands lies in the way the symlinks are restored. Whereas the `pctl recover` command simply rewrites the original symlinks to the Container private area (leaving the user files intact), the `pctl reinstall` command creates a new private area for the Container and re-writes the Container from scratch using its configuration files (thus retaining the Container IP address, hostname, resource control parameters, and all the other settings). The contents of the Container old private area are then copied to the `/old` directory in the new private area, to retain the user files.

The syntax of these commands is as follows:

```
pctl recover <CT_ID> [options]
pctl reinstall <CT_ID> [options]
```

The available options are listed below:

Option	Description
`--resetpwdb`	Removes the Container user database and creates a clean database as for any new installation.
`--skipbackup`	The contents of the old private area are not saved in the `/old` directory (for the `pctl reinstall` command only).
`--scripts` *`script1 script2 ...`*	Sets the scripts that will be executed during the Container reinstallation. These scripts are used to customize your application templates inside the new Container and bring them to the same state they were inside the old Container. By default, all available scripts are executed.
`--listscripts`	Lists the scripts that will be executed during the Container reinstallation to customize your application templates inside the new Container.
`--desc`	Displays the description of the scripts that will be executed during the Container reinstallation. Should be used together with the `--listscripts` option.

Note: If any of the Container application templates cannot be added to the Container in a normal way, the reinstallation process will fail. This may happen, for example, if an application template was added to the Container using the `--force` option of the `vzpkgadd` command.

pctl quotaon, pctl quotaoff, and pctl quotainit

These commands turn the quota on or off for the particular Container; the `pctl quotainit` command forces the quota to be initialized for the Container, i.e. its disk space and inodes recalculated. The Container ID must be specified after these commands with no additional options:

```
pctl quotaon <CT_ID>
pctl quotaoff <CT_ID>
pctl quotainit <CT_ID>
```

When the quota is turned on or initialized for the specified Container, the quota settings are taken from the Container configuration file. If you wish to change these settings, you should use the `pctl set` command.

pctl suspend and pctl resume

The `pctl suspend` command is used to save the state of a running Container. It has the following syntax:

```
pctl suspend <CT_ID>
```

During the `pctl suspend` execution, the current Container state is saved to a special dump file and the Container itself is stopped. The created dump file is saved to the `Dump` file in the `/vz/private/`*`CT_ID`*`/dump` directory on the server (or in the directory specified as the value of the `DUMPDIR` parameter in the Parallels Server Bare Metal global file).

The `pctl resume` command is used to restore the Container from its dump file created with the `pctl suspend` command. It has the following syntax:

```
pctl resume <CT_ID>
```

When executed, `pctl resume` searches for the `Dump` file in the `/vz/private/`*`CT_ID`*`/dump` directory on the server and restores the Container from this file.You can restore the Container dump file on the Source Server, i.e. on the server where this Container was running before its dumping, or transfer the dump file to another server and restore it there.

Note: Before restoring a Container from its dump file, make sure that the file system on the Destination Server is identical to that at the moment of the Container dumping. Otherwise, the Container restoration may fail.

pctl runscript

The `pctl runscript` command is used to run different shell scripts inside your Containers. It has the following syntax:

```
pctl runscript <CT_ID> <script_path>
```

For the command execution, you should specify the following parameters:

- the ID of the Container where the script is to be run
- the full path to the script on the server

You can execute this command for both running Containers and stopped ones. In the latter case, the corresponding Container will be started before running the specified scripts inside it.

vzlist

The `vzlist` utility is used to list the Containers existing on the given server together with additional information about these Containers. The output and sorting of this information can be customized as needed. The utility has the following syntax:

```
vzlist [-a] [-S] [-o parameter[.specifier] \
[,parameter[.specifier]...]] [-s [-]parameter[.specifier]] \
[-H] [-h hostname_pattern] [CT_ID ...] [-n] [-N name_pattern] \
[CT_ID [CT_ID ...]|-1]
vzlist -L
```

Here follows the description of available options:

Option	Description
`-a, --all`	Lists all the Containers existing on the server. By default, only running Containers are shown.
`-S, --stopped`	Lists only stopped Containers.
`-o` *parameter*`[.`*specifier*`]`	This option is used to display only particular information about the Containers. The parameters and their specifiers that can be used after the `-o` option are listed in the following subsection. To display a number of parameters in a single output, they should be separated with commas, as is shown in the synopsis above.
`-s, --sort` `[-]`*parameter*`[.`*specifier*`]`	Sorts the Containers in the list by the specified parameter. If "`-`" is given before the name of the parameter, the sorting order is reversed.
`-h, --hosthame` *hostname_pattern*	Displays only those Containers that correspond to the specified hostname pattern. The following wildcards can be used: `*`,`?`, and `[]`. **Note:** The last wildcard should be escaped to avoid shell interpretation.
`-H, --no-header`	Do not display column headers.
CT_ID	Displays only the Container with the specified ID. Several Container IDs separated with a space can be specified. If `-1` is given as the Container ID, the utility lists only IDs of the Containers existing on the server, with no additional information.
`-n, --name`	If used without any parameters, displays information on all the Containers on the server together with their names. If you indicate the Container ID after this option, displays information including the Container name on the specified Container only.
`-N, --name_filter` *name_pattern*	Displays only the Container that corresponds to the specified name pattern.
`-i, --netif` <*interface_name*>	Displays the Container whose `veth` virtual Ethernet interface name on the server corresponds to the specified name pattern.
`-d, --description` *desc_pattern*	Displays only the Container whose description corresponds to the specified pattern.

`-L, --list`	Lists all the parameters available to be used with the `-o` option.

vzlist Output Parameters and Their Specifiers

Almost any parameter that can be used after the `-o` and `-s` switches of the `vzlist` utility can be specified by the "dot+letter" combination following the parameter and denoting one of the following things:

Specifier	Description
`.m`	The maximal registered usage of the corresponding resource by the given Container.
`.b`	The barrier on using the corresponding resource set for the given Container.
`.l`	The limit on using the corresponding resource set for the given Container.
`.f`	The number of times the system has failed to allocate the corresponding resource for the given Container.
`.s`	The soft limit on using the corresponding resource set for the given Container.
`.h`	The hard limit on using the corresponding resource set for the given Container.

The following parameters are available for using with the utility:

Parameter	Possible Specifiers	Output Column	Description
`ctid`	none	`CTID`	The Container ID.
`hostname`	none	`HOSTNAME`	The Container hostname.
`ip`	none	`IP_ADDR`	The Container IP address.
`status`	none	`STATUS`	Specifies whether the Container is running or stopped.
`tm`	none	`TM`	Specifies the type of the OS template your Container is based on: ▪ `ST` indicates that the Container is based on a standard OS template. ▪ `EZ` indicates that the Container is based on an EZ OS template.
`ostemplate`	none	`OSTEMPLATE`	Specifies the name of the OS template your Container is based on (e.g. `redhat-el5-x86`).
`kmemsize`	`.m, .b, .l, .f`	`KMEMSIZE`	The size of unswappable kernel memory (in bytes), allocated for internal kernel structures of the processes of a particular Container. Typical amounts of kernel memory are 16…50 Kb per process.
`lockedpages`	`.m, .b, .l, .f`	`LOCKEDP`	The amount of memory not allowed to be swapped out (locked with the `mlock()` system call), in 4-Kb pages.
`privvmpages`	`.m, .b,`	`PRIVVMP`	The size in 4 Kb pages of private (or potentially private) memory, allocated by

	.l, .f		Container applications. Memory that is always shared among different applications is not included in this resource parameter.
shmpages	.m, .b, .l, .f	SHMP	The total size of shared memory (including IPC, shared anonymous mappings and `tmpfs` objects), allocated by processes of a particular Container, in 4 Kb pages.
numproc	.m, .b, .l, .f	NPROC	The number of processes and threads allowed.
physpages	.m, .b, .l, .f	PHYSP	The total size of RAM used by processes. This is accounting-only parameter currently. It shows the usage of RAM by the Container. For memory pages used by several different Containers (mappings of shared libraries, for example), only a fraction of a page is charged to each Container. The sum of the `physpages` usage for all Containers corresponds to the total number of pages used in the system by all accounted users.
vmguarpages	.m, .b, .l, .f	VMGUARP	The memory allocation guarantee, in pages (one page is 4 Kb). Applications are guaranteed to be able to allocate memory while the amount of memory accounted as `privvmpages` does not exceed the configured barrier of the `vmguarpages` parameter. Above the barrier, memory allocation may fail in case of overall memory shortage.
oomguarpages	.m, .b, .l, .f	OOMGUARP	The out-of-memory guarantee, in 4 Kb pages. Any Container process will not be killed even in case of heavy memory shortage if the current memory consumption (including both physical memory and swap) does not reach the `oomguarpages` barrier.
numtcpsock	.m, .b, .l, .f	NTCPSOCK	The number of TCP sockets (`PF_INET` family, `SOCK_STREAM` type). This parameter limits the number of TCP connections and, thus, the number of clients the server application can handle in parallel.
numflock	.m, .b, .l, .f	NFLOCK	The number of file locks created by all Container processes.
numpty	.m, .b, .l, .f	NPTY	The number of pseudo-terminals. For example, `ssh` session, `screen`, `xterm` application consumes pseudo-terminal resource.
numsiginfo	.m, .b, .l, .f	NSIGINFO	The number of `siginfo` structures (essentially this parameter limits size of signal delivery queue).
tcpsndbuf	.m, .b, .l, .f	TCPSNDB	The total size (in bytes) of send buffers for TCP sockets – amount of kernel memory allocated for data sent from an application to a TCP socket, but not acknowledged by the

			remote side yet.
`tcprcvbuf`	`.m, .b, .l, .f`	`TCPRCVB`	The total size (in bytes) of receive buffers for TCP sockets. Amount of kernel memory received from the remote side but not read by the local application yet.
`othersockb`	`.m, .b, .l, .f`	`OTHSOCKB`	The total size in bytes of UNIX-domain socket buffers, UDP and other datagram protocol send buffers.
`dgramrcvbuf`	`.m, .b, .l, .f`	`DGRAMRCVB`	The total size in bytes of receive buffers of UDP and other datagram protocols.
`nothersock`	`.m, .b, .l, .f`	`NOTHSOCK`	The number of socket other than TCP. Local (UNIX-domain) sockets are used for communications inside the system. UDP sockets are used for Domain Name Service (DNS) queries, for example.
`dcachesize`	`.m, .b, .l, .f`	`DCACHESIZE`	The total size in bytes of `dentry` and `inode` structures locked in memory. Exists as a separate parameter to impose a limit causing file operations to sense memory shortage and return an error to applications, protecting from excessive consumption of memory due to intensive file system operations.
`numfile`	`.m, .b, .l, .f`	`NFILE`	The number of files opened by all Container processes.
`numiptent`	`.m, .b, .l, .f`	`NIPTENT`	The number of IP packet filtering entries.
`diskspace`	`.s, .h`	`DQBLOCKS`	The total size of disk space consumed by the Container, in 1 Kb blocks. When the space used by a Container hits the barrier, the Container can allocate additional disk space up to the limit during grace period.
`diskinodes`	`.s, .h`	`DQINODES`	The total number of disk inodes (files, directories, symbolic links) a Container can allocate. When the number of inodes used by a Container hits the barrier, the Container can create additional file entries up to the limit during grace period.
`laverage`	none	`LAVERAGE`	The average number of processes ready to run during the last 1, 5 and 15 minutes.
`cpulimit`	none	`CPULIM`	This is a positive number indicating the CPU time in per cent the corresponding Container is not allowed to exceed.
`cpuunits`	none	`CPUUNI`	Allowed CPU power. This is a positive integer number, which determines the minimal guaranteed share of the CPU the Container will receive. You may estimate this share as ((Container CPUUNITS)/(Sum of CPU UNITS across all busy Containers))*100%. The total CPU power

			depends on CPU, and Parallels Server Bare Metal reporting tools consider one 1 GHz PIII Intel processor to be equivalent to 50,000 CPU units.
`slmmode`	none	`SLMMOD`	The SLM mode defining the behaviour of the SLM and UBC parameters in respect of the given Container. It can be one of the following: ▪ `ubc`: the SLM mode is disabled, which means that the `slmmemorylimit` parameter is not supported and you can use only the UBC parameters to control the amount of memory which can be consumed by the Container. ▪ `slm`: the SLM mode is enabled, which means that the `slmmemorylimit` parameter is supported and can be used to manage the amount of memory which can be consumed by the Container. ▪ `all`: both the `slmmemorylimit` and UBC parameters are supported and can be used to manage the amount of memory which can be consumed by the Container.
`slminst`	none	`SLMINST`	The instant memory usage limit set for the Container, in 4 KB pages.
`slmavg`	none	`SLMAVG`	The average memory usage limit set for the Container, in 4 KB pages.

If a parameter that can be used with a specifier is used without any specifier in the command-line, the current usage of the corresponding resource is shown by default.

vzquota

This command is used to configure and see disk quota statistics for Containers. `vzquota` is also used to turn on the possibility of using per-user/group quotas inside the Container. It allows you to configure per-user or per-group quota inside the Container as well. `pctl` uses `vzquota` internally to configure quotas and you usually do not have to use `vzquota` except for checking the current quota statistics. The syntax of `vzquota` command is as follows:

```
vzquota [options] command <CT_ID> [command-options]
```

General options available to all `vzquota` commands are:

`-v`	Verbose mode. Causes `vzquota` to print debugging messages about its progress. You can give up to two `–v` switches to increase verbosity.
`-q`	Quiet mode. Causes all warning and diagnostic messages to be suppressed. Only fatal errors are displayed.

Parallels Server Bare Metal quota works on a file system sub-tree or area. If this area has additional file systems mounted to its subdirectories, the quota will not follow these mount points. When you initialize quota, you specify the file system sub-tree starting point for the quota. Quota keeps its current usage and settings for a Container in the `/var/vzquota/quota.<CT_ID>` file.

Any quota file has a special flag, which indicates whether the file is “dirty”. The file is dirty when its content can be inconsistent with that of real quota usage. On the Container startup, quota will be re-initialized if the server was incorrectly brought down (for example power switch was hit). This operation may noticeably increase the server startup time.

For both the disk and inodes usage, Parallels Server Bare Metal allows you to set soft and hard limits as well as an expiration time. Upon reaching a soft limit, Parallels Server Bare Metal starts the expiration time counter. When the time is expired, the quota will block the subsequent disk space or inode allocation requests. The hard limit cannot be exceeded.

`vzquota` understands the following commands:

`init`	Before you can use quota, the current disk space and inode usage should be counted. For the `init` command, you must specify all the limits as well as the file tree where you want to initialize the quota.
`drop`	Removes the quota file.
`on`	Turns on quota accounting on the specified quota ID.
`off`	Turns off quota accounting on the specified quota ID.
`setlimit`	Allows you to change quota limits for the running quota.
`setlimit2`	Set the second-level quota parameters.
`stat`	Shows quota statistics for the running quota.
`show`	Shows quota usage from the quota file.

vzquota init

This command is used for counting current usage of disk space and inodes. It has the following syntax:

```
vzquota [options] init <CT_ID> [command-options]
```

The following options are understood by the `vzquota init` command:

`-s, --sub-quotas 1|0`	Optional. If the value used is 1 than per user/group quota is enabled in the Container. By default, user/group quotas are disabled.
`-b, --block-softlimit` *num*	Required. Disk quota block soft limit – amount of 1 Kb blocks allowed for the Container to use. This limit can be exceeded by the Container for the time specified by block expiration time (see below). When expiration time is off, the Container cannot allocate more disk space even if the hard limit is not yet reached.
`-B, --block-hardlimit` *num*	Required. Specifies disk quota block hard limit in 1 Kb blocks. This limit cannot be exceeded by the Container.
`-e, --block-exptime` *time*	Required. Expiration time for excess of the block soft limit. Time can be specified in two formats: ▪ *dd*:*hh*:*mm*:*ss* For example: 30 - 30 seconds; 12:00 - 12 minutes; 20:15:11:00 - 20 days, 15 hours, 11 minutes ▪ `xxA`, where A - h/H(hour); d/D(day); w/W(week); m/M(month); y/Y(year) For instance: 7D - 7 days; 01w - 1 week; 3m – 3 months
`-i, --inode-softlimit` *num*	Required. Inodes soft limit – amount of inodes allowed for the Container to create. This limit can be exceeded by the Container for the time specified by inode expiration time (see below). When expiration time is off the Container cannot create more inodes even if hard limit is not yet reached.
`-I, --inode-hardlimit` *num*	Required. Specifies inodes hard limit. This limit cannot be exceeded by the Container.
`-n, --inode-exptime` *time*	Required. Expiration time for excess of the inode soft limit. Time can be specified in two formats: ▪ *dd*:*hh*:*mm*:*ss* For example: 30 - 30 seconds; 12:00 - 12 minutes; 20:15:11:00 - 20 days, 15 hours, 11 minutes ▪ `xxA`, where A - h/H(hour); d/D(day); w/W(week); m/M(month); y/Y(year) For instance: 7D - 7 days; 01w - 1 week; 3m – 3 months
`-p` *path*	Required. Specifies the path to the Container private area.
`-c` *quota_file*	Optional. Specifies the file to write output of counted disk space and inodes as well as limits. If omitted, the default `/var/vzquota/quota.<CT_ID>` file is used.

vzquota drop

Removes the quota file. The syntax of this command is:

```
vzquota [options] drop <CT_ID> [-f] [-c quota_file]
```

The command checks whether the quota is running for a given Container and if it is, exits with error. An optional `–f` switch can be given to override this behavior and drop quota even if it is running. You can also override the path to the quota file to be dropped with an optional `–c` switch.

vzquota on and vzquota off

These commands are used to turn quota on and off. Their syntax is as follows:

```
vzquota [options] on <CT_ID> [command-options]
vzquota [options] off <CT_ID> [-f] [-c quota_file]
```

`vzquota off` turns the quota off for the file system tree specified in quota file given with an optional `–c` switch. If this switch is omitted, the default `/var/vzquota/quota.<CT_ID>` file is used. This command exits with error if for some reason quota file cannot be accessed and usage statistics could be lost. You can override this behavior by giving an optional `–f` switch.

`vzquota on` accepts the following options:

`-s, --sub-quotas 1|0`	Optional. If the value used is 1 then per user/group quota is enabled in the Container. By default user/group quotas are disabled.
`-u, --ugid-limit` *num*	Optional. Specifies the maximum number of user and group IDs for which usage statistics will be counted in this Container. If this value is 0, user/group quota will not be accounted. The default value is 0.
`-p` *path*	Required. Specifies the path to the Container private area.
`-f`	This option forces recalculation of quota usage even if the quota file does not have dirty flag set on.
`-c` *quota_file*	Optional. Specifies the file to write output of counted disk space and inodes as well as limits. If omitted, the default `/var/vzquota/quota.<CT_ID>` file is used.
`-b, --block-softlimit` *num* `-B, --block-hardlimit` *num* `-e, --block-exptime` *time* `-i, --inode-softlimit` *num* `-I, --inode-hardlimit` *num* `-n, --inode-exptime` *time*	These options are optional for the `vzquota on` command. They are described in the vzquota init subsection.

vzquota setlimit

This command updates limits for the running quota. It requires at least one limit to be specified. It also updates the corresponding quota file with new settings. The syntax of this command is:

```
vzquota [options] setlimit <CT_ID> [command-options]
```

Command options can be:

`-u, --ugid-limit` *num*	Optional. Specifies the maximum number of user and group IDs for which usage statistics will be counted in this Container. If this value is 0, user/group quota will not be accounted. Default value is 0.
`-b, --block-softlimit` *num* `-B, --block-hardlimit` *num* `-e, --block-exptime` *time* `-i, --inode-softlimit` *num* `-I, --inode-hardlimit` *num* `-n, --inode-exptime` *time*	These options are optional for the `vzquota on` command. However, at least one of these options or `-u`, `--ugid-limit` *num* must be specified. These options are described in the vzquota init subsection.
`-c quota_file`	Optional. Specifies the file where to write output of the counted disk space and inodes as well as limits. If omitted, the default `/var/vzquota/quota.<CT_ID>` file is used.

vzquota setlimit2

This command updates the second-level quota parameters for the running quota. It updates the corresponding quota file with new settings. The syntax of this command is:

```
vzquota [options] setlimit <CT_ID> [command-options]
```

You can use the following command options with `vzquota setlimit2`:

`-u, --ugid-limit` *num*	Optional. Specifies the maximum number of user and group IDs for which usage statistics will be counted in this Container. If this value is 0, user/group quota will not be accounted. Default value is 0.
`-b, --block-softlimit` *num* `-B, --block-hardlimit` *num* `-e, --block-exptime` *time* `-i, --inode-softlimit` *num* `-I, --inode-hardlimit` *num* `-n, --inode-exptime` *time*	These options are optional for the `vzquota on` command. These options are described in the vzquota init subsection.
`-c quota_file`	Optional. Specifies the file where to write output of the counted disk space and inodes as well as limits. If omitted, the default `/var/vzquota/quota.<CT_ID>` file is used.

vzquota stat and vzquota show

These commands are used for querying quota statistics. The syntax is as below:

```
vzquota [options] show <CT_ID> [-t] [-f] [-c quota_file]
vzquota [options] stat <CT_ID> [-t] [-c quota_file]
```

The difference between the `vzquota stat` and `vzquota show` commands is that the first one reports usage from the kernel while the second one reports usage as written in the quota file. However, by default `vzquota stat` updates the file with the last kernel statistics. If you do not want to update the quota file, add the `–f` switch to the command.

You can specify an alternative location to the quota file with the `–c` *`quota_file`* switch. Otherwise, the default `/var/vzquota/quota.<CT_ID>` file will be used.

To add information on user/group quota to the above commands output, use the `–t` command line switch.

A typical output of the `vzquota stat` command is shown below:

```
# vzquota stat 101 -t
   resource          usage       softlimit      hardlimit    grace
  1k-blocks         113856         2097152        2097152
     inodes          42539          200000         220000
User/group quota: on,active
Ugids: loaded 33, total 33, limit 100
Ugid limit was exceeded: no

User/group grace times and flags:
 type block_exp_time inode_exp_time  hex_flags
 user                                        0
group                                        0

User/group objects:
 type    ID  resource     usage   softlimit   hardlimit   grace status
 user     0 1k-blocks    113672           0           0         loaded
 user     0    inodes     42422           0           0         loaded
```

This output is suppressed for the sake of simplicity. As can be seen, Container 101 has the same soft and hard limits for disk space and Container can occupy up to 2 Gb of disk space. The current usage is 113 Mb. There are 42,539 inodes used by the Container, it has soft limit of 200,000 inodes and hard limit is set to 220,000. The empty grace column shows that grace period is started neither for inodes nor for disk space.

Per user/group quota is turned on and up to 100 users and groups are counted by the quota. Currently, there are 33 users and groups found in the Container and statistics for root is shown. There are no limits set from within the Container, and the current usage for root is 42,422 inodes and 113 Mb of disk space.

Migration Utilities

pmigrate

Migrating virtual machines and Containers is performed using the `pmigrate` utility. This utility has the following syntax:

```
pmigrate <source_server> <destination_server> [options]
```

<*source_server*> is the Source Server which can be either the server where the virtual machine and Container to be migrated is residing (if you are migrating a virtual machine and Container) or the physical server to be migrated (if you are migrating a physical server). <*destination_server*> is the Destination Server, i.e. the server where the virtual machine and Container or the physical server is to be migrated. If the Source and/or Destination Server is not specified, the operation is performed on the local server.

<*source_server*> and <*destination_server*> consists of two parts:

- <*type*> denotes the type of computer to migrate and can be one of the following:
  - `h` must be specified when migrating a physical server.
  - `c` must be specified when migrating a Container.
  - `v` must be specified when migrating a virtual machine.
- <*address*> denotes the location of computer to migrate and can be one of the following:
  - The computer location if you are migrating a physical server.
  - The computer location and the virtual machine name or Container ID if you are migrating a virtual machine or Container, respectively. The location must be separated from the virtual machine name/Container ID by the slash (`/`).

The location format is as follows:

```
[<user>[:<password>]@]<destination_server_IP_address_or_hostname>[:<destinatio
n_server_port>]
```

The options ([*options*]) you can use with `pmigrate` depend on whether you are migrating a virtual machine or a Container. This section describes the parameters for migrating Containers between Parallels servers and for moving virtual machines and physical servers to Containers. For information on parameters related to migrating virtual machines, refer to **pmigrate** (p. 211).

Container-related parameters

The following options can be used with `pmigrate` when migrating Containers between Parallels servers:

`-s, --nostart`	Do not attempt to start the Container on the Destination Server after its successful migration if the Container was running on the Source Server prior to the migration. This option does not have any effect if the Container was not running on the Source Server.
`-r, --remove-area yes|no`	This option takes precedence of the `REMOVEMIGRATED` setting from the global configuration file. If "yes" is specified,

	then the Container private area and configuration file will be deleted after successful migration. If “no” is specified, the private area and configuration file will be left on the Source Server and have the `.migrated` suffix appended to them.
`-f, --nodeps [=[all][,cpu_check] [,disk_space] [,technologies] [,license][,rate]]`	During its execution, `pmigrate` performs a number of checks on the Destination Server (e.g. it verifies that all OS and application templates required for the Container are present on the Destination Server) and if some checks fail, exits with an error. This option allows you to bypass all checks and migrate the Container. If you specify this option for a running Container, the Container will not be automatically started on the Destination Server. You should manually start it after adding the missing templates. You can additionally use one or several of the following parameters with this option: ▪ `all`: do not perform any checks on the Destination Server. ▪ `cpu_check`: do not check the CPU capabilities of the Destination Server. ▪ `disk_space`: do not check the amount of disk space on the Destination Server. ▪ `technologies`: do not check a set of technologies provided by the Parallels Server Bare Metal kernel on the Destination Server (see the description of the `TECHNOLOGIES` parameter in the Container Configuration File subsection for details). ▪ `license`: do not check the license installed on the Destination Server. ▪ `rate`: do not check the value of the `RATE` parameter in the Parallels Server Bare Metal global file.
`-b, --batch`	Normally, you do not have to specify this option. It is used by Parallels Server Bare Metal scripts and changes the screen output to a computer-parsable form.
`--ssh=<`*`ssh_options`*`>`	Additional options to be passed to `ssh` while connecting to the Destination Server. **Note:** Do not specify the Destination Server hostname as an option of `--ssh`.
`--keep-dst`	Do not remove the 'synched' Container private area on the Destination Server if some error occurred during the migration. This option allows you to prevent `pmigrate` from the repeated 'synching' the Container private area if the first migration attempt failed for some reason or other.
`--online`	Migrates the running Container with zero downtime. By default, the 'iterative online migration' type is used. During the migration: ▪ The main amount of Container memory is transferred to the Destination Server. ▪ The Container is 'dumped' and saved to an image file. ▪ The image file is transferred to the Destination Server where it is 'undumped'.

	Using this type of online migration allows you to attain the smallest service delay. To not use the 'iterative online migration' type, supply the `--noiter` option.
`--lazy`	Can be used only together with the `--online` option. Speeds up the zero downtime migration process if your Container is running a number of memory-consuming applications. This option allows you to decrease the size of the image file storing all Container private data and transferred to the Destination Server by leaving the main amount of memory in a locked state on the Source Server and swapping this memory from the Source Server on demand. While using the `--lazy` option, pay attention to the following: ▪ The migration type is set to "lazy" only if you additionally supply the `--noiter` option. ▪ If the `--noiter` option is not supplied, the migration type is set to 'lazy + iterative'.
`--noiter`	Can be used only together with the `--online` option. Sets the migration type to 'simple'. This option cannot be used together with the `--require-realtime` option.
`--require-realtime`	Can be used only together with the `--online` option. Forces `pmigrate` to move the Container by using the 'iterative online migration' type. If this migration type cannot be carried out for some reason or another, the command will fail and exit. This option cannot be used together with the `--noiter` option. If the default 'iterative online migration' type cannot be carried out, and this option is omitted , `pmigrate` will try to move your Container by making use of other types: the 'simple online migration' type or the 'lazy online migration' type (depending on the presence of the `--lazy` option).
`--readonly`	Just copy the specified Container to the Destination Server without making any changes to the Container on the Source Server.
`--dry-run`	Simulate the same operations as `pmigrate` completes without specifying this option (connects to the Destination Server, verifies that all OS and application templates required for the Container are present on the Server, etc.); however, the Container itself is not moved to the Destination Server.

The following options can be used with `pmigrate` when migrating a physical server or a virtual machine to a Container:

Name	Description
`-c`	Mandatory. The full path to the configuration file on the Parallels server that was created on the physical server by means of the `vzhwcalc` utility. You can specify only the name of the configuration file if you run the `vzp2v` utility from the directory where this file is located.
`-q`, `--quota`	Optional. The partition on your physical server which has any user and/or user groups quotas imposed on it. This partition will be

	migrated to the Container together with all quotas imposed on it. Moreover, these quotas will be applied to the entire Container after the server migration.
`-z, --eztmpl`	Optional. The EZ OS template to be used to create the Container. You may list all OS templates installed on the Parallels server together with their updates by executing the `vzpkg list` command. If an OS template is not specified, the `mkvzfs` command is executed during the Container creation which makes an empty private area with the name of `/vz/private/`*`CT_ID`* on the Parallels server. This private area is then used to copy all the physical server files to it.
`-d, --dist`	Optional. The Linux version your physical server is running. The name of the version specified should coincide with the name of the corresponding distribution configuration file located in the `/etc/vz/conf/dist` directory on the Parallels server. For example, if you specify `rhel-5` as the value of this option, the `rhel-5.conf` file should be present in the `/etc/vz/conf/dist` directory on the Parallels server. You must obligatorily set this option, if there is no `DISTRIBUTION` variable specified in the server configuration file. In case the `DISTRIBUTION` variable is set in the configuration file and you have specified the `-d` option, the latter takes precedence.
`--exclude`	Optional. The path to the directories and files which will be excluded from copying to the Container. This option allows you to avoid migrating the data you do not need. To gain more understanding on this option, please consult the man pages for the `rsync` utility from where it was borrowed. **Note**: We strongly recommend that you exclude the directories you were informed of while running the `vzhwcalc` utility on the physical server.
`-S, --srvstop`	Optional. The services to be stopped for the time of the physical server migration. We recommend that you stop all the services on the physical server except for the critical ones (e.g. the `sshd` service that is needed to provide communication between the physical server and the Parallels server) before the migration. This will prevent the running services from modifying any files being moved.

vzmigrate

This command can be used along with the `pmigrate` utility for moving Containers from one Parallels server to another with minimal or zero downtime. It has the following syntax:

```
vzmigrate [options] Destination_Server {Container_list}
```

`{CT_list}` is a list of `<CT_ID>[:<new_CT_ID>]` pairs. A new Container ID parameter is needed in case both the Source Server (the one where you run the `vzmigrate` command) and the Destination Server have a Container with the ID of `<CT_ID>`. You can specify multiple Containers at once for migration.

Note: For more information on `pmigrate`, refer to Migrating Virtual Machines and Containers (p. 116).

The following options can be used with `vzmigrate`:

`-s, --nostart`	Do not attempt to start the Container on the Destination Server after its successful migration if the Container was running on the Source Server prior to the migration. This option does not have any effect if the Container was not running on the Source Server.
`-r, --remove-area yes|no`	This option takes precedence of the `REMOVEMIGRATED` setting from the global configuration file. If "yes" is specified, then the Container private area and configuration file will be deleted after successful migration. If "no" is specified, the private area and configuration file will be left on the Source Server and have the `.migrated` suffix appended to them.
`-f, --nodeps [=[all][,cpu_check] [,disk_space] [,technologies] [,license][,rate]]`	During its execution, `vzmigrate` performs a number of checks on the Destination Server (e.g. it verifies that all OS and application templates required for the Container are present on the Destination Server) and if some checks fail, exits with an error. This option allows you to bypass all checks and migrate the Container. If you specify this option for a running Container, the Container will not be automatically started on the Destination Server. You should manually start it after adding the missing templates. You can additionally use one or several of the following parameters with this option: ▪ `all`: do not perform any checks on the Destination Server. ▪ `cpu_check`: do not check the CPU capabilities of the Destination Server. ▪ `disk_space`: do not check the amount of disk space on the Destination Server. ▪ `technologies`: do not check a set of technologies provided by the Parallels Server Bare Metal kernel on the Destination Server (see the description of the `TECHNOLOGIES` parameter in the Container Configuration File subsection for details). ▪ `license`: do not check the license installed on the Destination Server.

	▪ `rate`: do not check the value of the `RATE` parameter in the Parallels Server Bare Metal global file.
`-b, --batch`	Normally, you do not have to specify this option. It is used by Parallels Server Bare Metal scripts and changes the screen output to a computer-parsable form.
`--ssh=<`*`ssh_options`*`>`	Additional options to be passed to `ssh` while connecting to the Destination Server. **Note:** Do not specify the Destination Server hostname as an option of `--ssh`.
`--keep-dst`	Do not remove the 'synched' Container private area on the Destination Server if some error occurred during the migration. This option allows you to prevent `vzmigrate` from the repeated 'synching' the Container private area if the first migration attempt failed for some reason or other.
`--online`	Migrates the running Container with zero downtime. By default, the 'iterative online migration' type is used. During the migration: ▪ The main amount of Container memory is transferred to the Destination Server. ▪ The Container is 'dumped' and saved to an image file. ▪ The image file is transferred to the Destination Server where it is 'undumped'. Using this type of online migration allows you to attain the smallest service delay. To not use the 'iterative online migration' type, supply the `--noiter` option.
`--lazy`	Can be used only together with the `--online` option. Speeds up the zero downtime migration process if your Container is running a number of memory-consuming applications. This option allows you to decrease the size of the image file storing all Container private data and transferred to the Destination Server by leaving the main amount of memory in a locked state on the Source Server and swapping this memory from the Source Server on demand. While using the `--lazy` option, pay your attention to the following: ▪ The migration type is set to "lazy" only if you additionally supply the `--noiter` option. ▪ If the `--noiter` option is not supplied, the migration type is set to 'lazy + iterative'.
`--noiter`	Can be used only together with the `--online` option. Sets the migration type to 'simple'. This option cannot be used together with the `--require-realtime` option.
`--require-realtime`	Can be used only together with the `--online` option. Forces `vzmigrate` to move the Container by using the 'iterative online migration' type. If this migration type cannot be carried out for some reason or another, the command will fail and exit. This option cannot be used together with the `--noiter` option.

	If the default 'iterative online migration' type cannot be carried out, and this option is omitted , `vzmigrate` will try to move your Container by making use of other types: the 'simple online migration' type or the 'lazy online migration' type (depending on the presence of the `--lazy` option).
`--readonly`	Just copy the specified Container to the Destination Server without making any changes to the Container on the Source Server.
`--dry-run`	Simulate the same operations as `vzmigrate` completes without specifying this option (connects to the Destination Server, verifies that all OS and application templates required for the Container are present on the Server, etc.); however, the Container itself is not moved to the Destination Server.

vzmlocal

Moving/copying a Container within one and the same server consists in changing/adding the Container ID, private area, and root paths. Thus, you may use the `vzmlocal` utility either to change the ID and/or the private area path and/or the root path of any existing Container(s) or to clone a Container, i.e. to create a complete copy of an existing Container with different ID and paths. It has the following syntax:

```
vzmlocal <source_CT_ID>[:<dest_CT_ID> \
[:<dest_private>[:dest_root]] [...]
vzmlocal -C <source_CT_ID>:<dest_CT_ID> \
[:<dest_private>[:dest_root]] [...]
vzmlocal -h
```

The options are the following:

`-h`	Display the utility help.
`-C`	Clone the source Container instead of moving it.

You should specify the source Container ID (*<source_CT_ID>*) and the destination Container ID (*<dest_CT_ID>*). Specifying the destination Container private area path (*<dest_private>*) and root path (*<dest_root>*) is optional; it allows you to override the default paths - `/vz/private/`*<dest_CT_ID>* and `/vz/root/`*<dest_CT_ID>*, correspondingly.

Notes:

1. You may perform a number of copying/moving operations by a single invocation of the `vzmlocal` utility.

2. You may run the `vzmlocal` utility on both running and stopped Containers.

vzp2v

`vzp2v` can be used along with the `pmigrate` utility to migrate a physical server to a Container on your Parallels server. It has the following syntax:

```
vzp2v [user[:password]@]address[:port] [options]
```

Note: For more information on `pmigrate`, refer to Migrating Virtual Machines and Containers (p. 116).

The options that can be used with the `vzp2v` utility are listed in the table below:

Name	Description
`--ctid`	Mandatory. The ID of the Container that will be created on the Parallels server and where the physical server will be migrated. You can specify any unoccupied ID.
`-c`	Mandatory. The full path to the configuration file on the Parallels server that was created on the physical server by means of the `vzhwcalc` utility. You can specify only the name of the configuration file if you run the `vzp2v` utility from the directory where this file is located.
`-q`, `--quota`	Optional. The partition on your physical server which has any user and/or user groups quotas imposed on it. This partition will be migrated to the Container together with all quotas imposed on it. Moreover, these quotas will be applied to the entire Container after the server migration.
`-z, --eztmpl`	Optional. The EZ OS template to be used to create the Container. You may list all OS templates installed on the Parallels server together with their updates by executing the `vzpkg list` command. If an OS template is not specified, the `mkvzfs` command is executed during the Container creation which makes an empty private area with the name of `/vz/private/CT_ID` on the Parallels server. This private area is then used to copy all the physical server files to it.
`-t, --ostmpl`	Optional. The OS template to be used to create the Container. You can list all OS templates installed on the Parallels server together with their updates by executing the `vzpkgls` command. If an OS template is not specified, the `mkvzfs` command is executed during the Container creation which makes an empty private area with the name of `/vz/private/CT_ID` on the Parallels server. This private area is then used to copy all the physical server files to it.
`-d`, `--dist`	Optional. The Linux version your physical server is running. The name of the version specified should coincide with the name of the corresponding distribution configuration file located in the `/etc/vz/conf/dist` directory on the Parallels server. For example, if you specify `rhel-5` as the value of this option, the `rhel-5.conf` file should be present in the `/etc/vz/conf/dist` directory on the Parallels server. You must obligatorily set this option, if there is no `DISTRIBUTION` variable specified in the server configuration file. In case the `DISTRIBUTION` variable is set in the configuration file and you have specified the `-d` option, the latter takes precedence.
`--exclude`	Optional. The path to the directories and files which will be excluded

	from copying to the Container. This option allows you to avoid migrating the data you do not need. To gain more understanding on this option, please consult the man pages for the `rsync` utility from where it was borrowed. **Note**: We strongly recommend that you exclude the directories you were informed of while running the `vzhwcalc` utility on the physical server.
`-S, --srvstop`	Optional. The services to be stopped for the time of the physical server migration. We recommend that you stop all the services on the physical server except for the critical ones (e.g. the `sshd` service that is needed to provide communication between the physical server and the Parallels server) before the migration. This will prevent the running services from modifying any files being moved.
`-h, --help`	Prints information on the utility options.
`--usage`	Prints usage information.

Backing-Up Utilities

Any Container is defined by its private area, configuration files, action scripts, and quota information. Backing up these components allows you to restore the whole Container on any Parallels Server Bare Metal-based system at any time in case the Container gets broken.

pbackup

The `pbackup` utility is run on the so-called Backup Server. It connects via SSH to the servers where some or all Containers are to be backed up and puts the tarballs into the directory defined in the `/etc/vzbackup.conf` global backup configuration file (by default, this directory is `/vz/backup`). Later on, the Container backups may be restored from this directory. It has the following syntax:

```
pbackup [backup_options] SERVER1 ... [CT options]
```

You may specify any number of servers names or IP addresses in the command-line. You may also enter these names as the value of the `BACKUP_NODES` parameter in the global backup configuration file to avoid the necessity to specify them in the command-line. In this case, you shall specify the `–a` option instead.

Note: While the following two subsections provide the complete reference on the `pbackup` and `prestore` utilities, many of their options can be specified in the `/etc/vzbackup.conf` configuration file to be used as the default ones.

The backup options are the following:

`-F`	Force a plain full backup. Plain full backups of Containers do not allows creating incremental backups on their basis.
`-I`	Force a full backup.
`-i`	Make an incremental backup or, if no full backups are available, a full backup.
`-Cg`	Compress the resulting Container backups with the `gzip` algorithm. This option takes precedence of the `BACKUP_COMPRESS` parameter in the backup configuration file.
`-Cb`	Compress the resulting Container backups with the `bzip2` algorithm. This option takes precedence of the `BACKUP_COMPRESS` parameter in the backup configuration file.
`-Cn`	Do not compress the resulting Container backups. This option takes precedence of the `BACKUP_COMPRESS` parameter in the backup configuration file.
`-a`	Back up all servers specified in the global backup configuration file.
`-c` *`CONFIG`*	Use an alternative backup configuration file.
`-s`	The Containers are to be stopped before their backing up. In this case, if a client tries to access the Containers during their downtime, a temporary "busy" page is shown. This option takes precedence of the `BACKUP_VESTOP` parameter in the backup configuration file.
`-z`	Use the VZFS tracking facility to decrease the Container downtime. With this option, the backing up is performed in two stages. On the first stage, the Container is backed up while still running. On the second stage, it is stopped, and the changes in the Container file system that have been made during the first stage are added to the backup. Valid only if used together with the `-s` option.
`-n`	The Containers are NOT to be stopped before their backing up. This option takes precedence of the `BACKUP_VESTOP` parameter in the backup configuration file.

`-p`	Apply time and load restriction rules for periodic backups. These rules are defined by the `BACKUP_KEEP_MAX` and `BACKUP_LOADAVG_MAX` parameters in the backup configuration file (`/etc/vzbackup.conf`). Without this option, these parameters do not take effect. This option is useful if you invoke `pbackup` in an unattended mode as a cron job. It overrides the `CRON_BACKUP` parameter in the global backup configuration file.
`--desc` *`DESCRIPTION`*	The description of the backup archive.
`-j`	Is opposite to the `-p` switch. Turns off the periodic backup mode and disregards the `BACKUP_KEEP_MAX` and `BACKUP_LOADAVG_MAX` parameters in the backup configuration file (`/etc/vzbackup.conf`). It overrides the `CRON_BACKUP` parameter in the global backup configuration file.
`-L`	Make use of the `BACKUP_FINISH_TIME` and `BACKUP_LIMIT_TIME` in the backup configuration file (`/etc/vzbackup.conf`).
`-rm-tag` *`tag`*	Create a backup and then remove the backup with the specified tag. You can learn the tags of the existing backups on the server by using, for example, the `prestore -l` command.
`-rm-old`	Create a backup and then remove the oldest backup of the specified server/Container(s).
`--ssh-opts` *`OPTIONS`*	Options to be passed to `ssh`. See examples in the backup configuration file.
`--vzcache`	Back up not the Containers themselves, but the cache area of the server (`/vz/template/vzcaches`).

The Container options define the list of Containers to be backed up:

`-e` *`CT1`*`...`	The Containers to back up on the server. Containers can be specified using both their IDs (e.g. `101` or `102`) and their names (e.g. `comp1` or `comp2`). If the `--vzcache` option is specified, not the Containers themselves, but their caches will be backed up.
`-x` *`CT1`*`...`	The Containers that need not be backed up (Containers to exclude). Containers can be specified using both their IDs (e.g. `101` or `102`) and their names (e.g. `comp1` or `comp2`). If the `--vzcache` option is specified, not the Containers themselves, but their caches will be excluded from backing up.

It is sufficient to specify only one Container option: either `–e`, or `–x`; or to do without any Container options if all the Containers from the specified server(s) are to be backed up.

prestore

The `prestore` utility is also run on the Backup Server. It uses the Container backups stored on the Backup Server to restore them to their original servers (or to any other location if the `–d` option is specified). The syntax of the command is the following:

```
prestore [restore_options] server1 ... [CT_options]
```

You may specify any number of servers (their names or IP addresses) whose Containers were at one time backed up and now need to be restored.

The restore options are the following:

`-c CONFIG`	Use an alternative backup configuration file.
`-d NODE`	The Destination Server to restore the Containers to. If no Destination Server is specified, the Containers are restored to the server from which they were originally backed up.
`-t TAG`	Specify the tag of any intermediary incremental backup to be restored.
`-r TAG`	Remove the backup tagged by the TAG value. This option is valid only if a single Container is specified and if the TAG value specifies the last incremental backup.
`--rm-prev TAG`	Remove the backup tagged by the TAG value together with all the previous backups. This option is valid only if a single Container is specified and if the TAG value specifies the last incremental backup.
`-l`	Do not restore any Containers. Show the information on the Containers available to be restored.
`-f`	Show the full information on the backed up Containers (only if the `-l` option is specified).
`--chain`	Show a tag chain (only if the `–l` option is specified).
`--desc config`	Display the configuration file inside the archive with the specified tag (only if the `-l` and `-t` options are specified).
`--single-tar`	Indicates that a single tar is coming (`stdin`).
`--skip-check-vzcache`	Do not restore the backups of the Container cached files stored in the `/vz/backup/CT_ID/vzcache` directory on the Backup Server. For example, you should use this option while restoring a Container with cached files to a Destination Server other than the Backup Server.
`--vzcache`	Restore the `vzcache` template area keeping the Container cached files and located in the `/vz/template/vzcaches` directory on the server.

The Container options define a list of Containers to be restored:

`-e CT1...`	The Containers to be restored on the server. Any Container can be specified using both its IDs (e.g. `101` or `102`) and its names (e.g. `comp1` or `comp2`).
`-x CT1...`	The Containers that need not be restored (Containers to exclude). Any Container can be specified using both its IDs (e.g. `101` or `102`) and its names (e.g. `comp1` or `comp2`).

It is sufficient to specify only one Container option: either `–e`, or `–x`; or to do without any Container options if all the Containers from the specified servers are to be restored.

EZ Template Management Utilities

The following utilities can be used to perform EZ templates-related operations:

- `vzmktmpl`. This utility is used to create OS and application EZ templates.
- `vzpkgproxy`. This utility is used to set up a caching proxy server meant for handling your OS and application EZ templates.
- `vzrhnproxy`. This utility is used to set up a Red Hat Network (RHN) Proxy Server meant for handling the packages included in the RHEL 4 OS EZ template.
- `vzmtemplate`. This utility is used to migrate the installed OS EZ templates from the Source Server to the Destination Server. Detailed information on this utility is provided in the vzmtemplate subsection (p. 174).
- `vzpkg`. This utility is used to perform to manage OS and application EZ templates either inside your Containers or on the server itself. This tool can also be used to manage standard software packages (e.g. the `mysql.rpm` package) inside a Container. The syntax of `vzpkg` is:

```
vzpkg command [options] <CT_ID>
vzpkg --help
```

Where *`command`* can be one of the following:

`install template`	Used to install OS and application EZ templates on the server.
`update template`	Used to update OS and application EZ templates installed on the server.
`remove template`	Used to remove OS and application EZ templates from the server.
`list`	Used to list EZ templates or software packages either on the server or inside a particular Container.
`info`	Used to get information on any EZ templates or software packages available on the server or inside the Container.
`status`	Used to display the updates for the packages installed inside the Container.
`install`	Used to add application EZ templates to or to install software packages inside the Container.
`update`	Used to update application EZ templates and software packages inside the Container.
`link`	Used to replace the real application files inside your Container(s) with the symlinks to the same files on the server.
`remove`	Used to remove application EZ templates or software packages from the Container.
`create cache`	Used to create a tarball (cache) for the given OS EZ template.
`update cache`	Used to update the existing tarball (cache) for the given OS EZ template.
`remove cache`	Used to remove a tarball (cache) for the given OS EZ template.
`localinstall`	Used to install a software package inside a Container from the corresponding file on the server.

`localupdate`	Used to update the software packages installed inside your Container(s) by means of the `vzpkg install` or `vzpkg localinstall` commands.
`upgrade`	Used to upgrade an OS EZ template the Container is based on to a newer version.
`fetch`	Used to download packages included in EZ templates to the server and to store them in the `vzpkg` local cache.
`clean`	Used to remove all locally cached data from the template directories on the server.
`update metadata`	Used to update the local metadata on the server.

vzpkg install template

This command is used to install an OS or application EZ template on the server from the corresponding package. It has the following syntax:

```
vzpkg install template [options] <path_to_package> ...
```

You can use the following options with this command:

Name	Description
`-q, --quiet`	Disables logging to the screen and to the log file.
`-f, --force`	Forces the EZ template installation the server.

When executed, the `vzpkg install template` command installs an application or OS template on the server from the specified package (*`<package>`*). You can install a number of templates at once by specifying the corresponding packages and separating them by spaces.

vzpkg update template

This command is used to update an OS or application EZ template on the server from the corresponding package. It has the following syntax:

```
vzpkg update template [options] <path_to_package> ...
```

This command can be used with the following options:

Name	Description
`-q, --quiet`	Disables logging to the screen and to the log file.
`-f, --force`	Forces the EZ template update.

When executed, the `vzpkg update template` command updates an application or OS EZ template on the server from the specified file (*`<package>`*). You can update a number of templates at once by specifying the corresponding packages and separating them by spaces.

vzpkg remove template

This command is used to remove an OS or application EZ template that you do need any more from the server. It has the following syntax:

```
vzpkg remove template [options] [-F OS_template] <template_name> ...
```

You can pass the following options to this command:

Name	Description
`-q, --quiet`	Disables logging to the screen and to the log file.
`-f, --force`	Forces the EZ template deletion from the server.

When executed, the `vzpkg remove template` command removes the specified OS EZ template (<*template_name*>) from the server. To delete an application EZ template, you should additionally specify the name of the OS EZ template (*OS_template*) under which this application template is to be run.

vzpkg list

The `vzpkg list` command is used to list the EZ templates or software packages installed on the server or already added to a particular Container. It has the following syntax:

```
vzpkg list [options] <CT_ID>|<CT_NAME> [...]
vzpkg list [options] [<OS_template>|<CT_ID>|<CT_NAME> [...]]
```

If you indicate one or more Container IDs or names, the command will list the EZ templates applied to the specified Containers. If you indicate one or more OS EZ templates, `vzpkg list` will display a list of application EZ templates available for these OS EZ templates. Without the *<CT_ID>/<CT_NAME>* and <*OS_template*> arguments, the utility lists all EZ templates available for Containers on the server.

The following options can be used with the `vzpkg list` command:

Name	Description
`-p, --package`	If the <*CT_ID*> or <*CT_NAME*> argument is given, the command lists all software packages installed inside the Container. If the <*OS_template*> argument is given, the command lists all packages included in the OS EZ template. Without the <*CT_ID*>/<*CT_NAME*> and <*OS_template*> arguments, `vzpkg list` displays all packages available on the server.
`-O, --os`	If the <*CT_ID*> or <*CT_NAME*> argument is given, the command displays the OS EZ template the Container is based on. Without the <*CT_ID*>/<*CT_NAME*> argument, `vzpkg list` lists all OS EZ templates installed on the server.
`-A, --app`	If the <*CT_ID*> or <*CT_NAME*> argument is given, the command displays the application EZ templates installed inside the Container. If the <*OS_template*> is given, the command shows the application EZ templates which can be used with the OS EZ template specified. Without the <*CT_ID*>/<*CT_NAME*> and <*OS_template*> arguments, `vzpkg list` lists all application EZ templates installed on the server.
`-C, --cache`	Lists the packages included in the specified EZ template or applied to the specified Container from the local `vzpkg` cache. You can omit this parameter if the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file. Should be used along with the `-p` option.
`-r, --remote`	If the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file, you should use this option to make `vzpkg list` list the packages included in the specified EZ template or applied to the specified Container in the remote repositories. Should be used along with the `-p` option.
`-u, --custom-pkg`	Displays a list of packages that are applied to the specified Container but absent from the repository set to handle the EZ template(s) where these packages are included.
`-i, --pkgid`	Displays the ID assigned to the EZ template instead of its name; these IDs are unique within the given system. If the <*CT_ID*> or <*CT_NAME*> argument is given, the command shows the IDs of the EZ templates available inside the Container. If the <*OS_template*> argument is

	given, the command displays the IDs of the OS EZ template specified and all its EZ application templates. Without the `<`*`CT_ID`*`>/<`*`CT_NAME`*`>` and `<`*`OS_template`*`>` arguments, the IDs of all EZ templates installed on the server are shown.
`-S, --with-summary`	In addition to listing the EZ templates available either inside the Container (if the `<`*`CT_ID`*`>` or `<`*`CT_NAME`*`>` argument is given) or installed on the server (if the `<`*`CT_ID`*`>/<`*`CT_NAME`*`>` argument is omitted), this option makes `vzpkg list` display the summary information on the corresponding EZ templates/packages.
`-c, --cached`	This option has no effect if the `<`*`CT_ID`*`>` or `<`*`CT_NAME`*`>` argument is given. If used for listing the EZ templates available on the server, it makes `vzpkg list` omit all application and OS EZ templates for which the cache has not been created (by running the `vzpkg create cache` command). In other words, with this option on, `vzpkg list` will list only the OS EZ templates ready to be used for the Container creation.
`-d, --debug <`*`num`*`>`	Sets the debugging level to one of the specified values (from 0 to 10). 10 is the highest debug level and 0 sets the debug level to its minimal value.
`-q, --quiet`	Disables logging to the screen and to the log file.

vzpkg info

This command provides information on the EZ templates or software packages installed on the server or applied to the Container. The syntax of the utility is as follows:

```
vzpkg info [-F <OS_template>|<CT_ID>|<CT_NAME>] -q|-d <app_template>
           [<parameters> ...]
vzpkg info -p [-C|-r] [-F <OS_template>|<CT_ID>|<CT_NAME>] -q|-d
           <package_name> [<parameters> ...]
```

The available options are described in the following table:

Name	Description
`-F, --for--os <os_name>|<CT_ID>`	Displays information on the application EZ template or the software package (if the `-p` option is specified) included in the specified OS EZ template or applied to the indicated the Container.
`-p, --package`	If the `<CT_ID>` or `<CT_NAME>` argument is given, the command lists all software packages installed inside the Container. If the `<OS_template>` argument is given, the command lists all packages included in the OS EZ template.
`-C, --cache`	Displays the information on the specified package from the local `vzpkg` cache. You can omit this parameter if the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file.
`-r, --remote`	If the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file, you should use this option to make `vzpkg info` get the information on the specified package from the remote repositories set for handling the EZ template where this package is included.
`-d, --debug <num>`	Sets the debugging level to one of the specified values (from 0 to 10). 10 is the highest debug level and 0 sets the debug level to its minimal value.
`-q, --quiet`	Disables logging to the screen and to the log file.

While executed, `vzpkg info` parses the subdirectories and files located in the `/vz/template/<os_name>/<os_version>/<arch>/config` directory and containing the EZ template meta data. To run the command, you should specify either the OS EZ template name or the Container ID. In either case, detailed information on the corresponding OS EZ template is displayed. You can also use the `-F` option to get the necessary information on any application EZ template included into the OS EZ template or applied to the Container.

By default, `vzpkg info` displays all meta data on the EZ template/package specified. However, you can reduce the amount of the output information by using special parameters (`<parameters>`) listed in the table below:

Name	Description
`name`	The name of the EZ template/package.
`packages`	The packages included in the EZ template. For EZ templates only.
`repositories`	The repository where the packages comprising the EZ template are stored. For EZ templates only.
`mirrorlist`	The URL to the file containing a list of repositories from where the packages comprising the EZ template are to be downloaded. For EZ

	templates only.
`distribution`	The Linux distribution on the basis the OS EZ template has been created or under which the application EZ template is to be run. For EZ templates only.
`summary`	Brief information on the EZ template/package.
`description`	Detailed information on the EZ template/package. As distinct from `summary`, it can contain additional data on the EZ template/package.
`technologies`	Displays the following information: ▪ The microprocessor architecture where the EZ template is to be used (`x86`, `x86`, `ia64`); ▪ Specifies whether the EZ template can be used only on the servers with the Native POSIX Thread Library (NPTL) support. In this case the `nptl` entry is displayed after the `vzpkg info` execution. For EZ templates only.
`version`	The version of the software package.
`release`	The release of the software package.
`arch`	The system architecture where the EZ template/package is to be used. It can be one of the following: ▪ `x86` if the EZ template/package is to be used on 32-bit platforms; ▪ `x86_64` if the 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` if the EZ template is to be used on IA-64-bit platforms (i.e. on servers with the Itanium 2 processor installed).
`config_path`	Displays the path to the EZ template configuration directory containing the template meta data where the meta data for the base OS EZ template are stored (the default directory path is `/vz/template/<os_name>/<os_version>/<arch>/config/os/default`).
`package_manager_type`	The packaging system used to handle the packages included in the specified EZ template. It can be one of the following: ▪ `rpm` for RPM-based Linux distributions (Fedora Core, Red Hat Enterprise Linux, etc.); ▪ `dpkg` for Debian-based Linux distributions (e.g. Debian and Ubuntu). For EZ templates only.
`package_manager`	The package manager type used to manage the packages included in the specified EZ template. It can be one of the following: *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) and CentOS 4, 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, Fedora 7 and 8;
- `rpm43x64`: Red Hat Enterprise Linux 3 and 4 (with the 2.6 kernel and NPTL support) and CentOS 4, 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.

vzpkg status

This command is used to check the status of the packages either installed inside a Container or included in an OS EZ template. It has the following syntax:

```
vzpkg status [options] <CT_ID>|<CT_NAME>|<OS_template>
```

You can use the following options with `vzpkg status`:

Option	Description
`-C, --cache`	Makes the `vzpkg status` command look for available updates in the local `vzpkg` cache only. You can omit this parameter if the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file.
`-r, --remote`	If the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file, you should use this option to make `vzpkg status` look for the package updates in the remote repositories set for handling the corresponding EZ template.
`-d, --debug <num>`	Sets the debugging level to one of the specified values (from 0 to 10). 10 is the highest debug level and 0 sets the debug level to its minimal value.
`-q, --quiet`	Disables logging to the screen and to the log file.

When executed, the command performs the following operations:

- Checks all the packages installed inside the specified Container or included in the specified OS EZ template.
- Checks the repository used to install/update packages inside the Container/OS EZ template.
- Compares the packages in the repository with those inside the Container/OS EZ template.
- Lists the found packages updates for the Container/OS EZ template, if any, or informs you that the Container/OS EZ template is up-to-date.

Note: The `vzpkg status` command can be executed for running Containers only.

vzpkg install

This command is used to add an application EZ template to or install a software package inside a Container. It has the following syntax:

```
vzpkg install [options] <CT_ID>|<CT_NAME> <object> [...]
```

The `vzpkg install` command will add an object (`<object>`) which can be either an application EZ template or a standard software package to the Container with the ID of *`<CT_ID>`* or with the name of *`<CT_NAME>`*. You can specify a number of objects to be applied to your Container.

When executed, `vzpkg install` automatically handles the interdependencies among the packages to be installed inside a Container and ensures that all dependencies are satisfied. If the package dependencies cannot be resolved, the installation process will fail and the corresponding message will be displayed.

Options available to this command are:

Name	Description
`-p, --package`	Tells the `vzpkg install` command to install software packages instead of EZ templates.
`-f, --force`	Forces the EZ template/package installation.
`-C, --cache`	Makes the `vzpkg install` command look for the packages included in the EZ template in the local `vzpkg` cache only. If there is a package not available locally, the command will fail. You can omit this parameter if the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file.
`-r, --remote`	If the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file, you should use this option to make `vzpkg install` look for the packages in the remote repositories set for handling the corresponding EZ template.
`-n, --check-only`	Simulates the same operations as `vzpkg install` completes without specifying this option (downloads the software packages to the server, handles the package interdependencies, etc.); however, the packages themselves are not installed in the specified the Container.
`-d, --debug <num>`	Sets the debugging level to one of the specified values (from 0 to 10). 10 is the highest debug level and 0 sets the debug level to its minimal value.
`-q, --quiet`	Disables logging to the screen and to the log file.

By default, the specified object is treated by `vzpkg install` as an application EZ template. However, you can use the `-p` option to explicitly specify that the object should be considered as a software package.

Installing packages by using the `vzpkg install` command has the following main advantages:

- The dependencies for software packages are automatically checked during the installation process; so, you do not have to bother any more on how to resolve possible package dependencies.
- During the package installation, only symlinks to the installed files on the server are created and added to the Container private area, which allows you to noticeably save disk space inside your Containers.
- You can update the installed packages by running the `vzpkg update` command. For the information on this command, please see the next subsection.

Note: A Container has to be running in order to apply an application EZ template to or install a package inside this Container.

vzpkg update

The `vzpkg update` command is used to update either the OS EZ template a Container is based on or any of the application EZ templates inside the Container. Besides, you can use this command to update the software packages installed inside your Container by means of the `vzpkg install` command. It has the following syntax:

```
vzpkg update [options] <CT_ID>|<CT_NAME> [<object> [...]]
```

The following options can be used with `vzpkg update`:

Name	Description
`-C, --cache`	Makes the `vzpkg update` command look for the package updates in the local `vzpkg` cache only. You can omit this parameter if the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file.
`-r, --remote`	If the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file, you should use this option to make `vzpkg update` look for the package updates in the remote repositories set for handling the corresponding EZ templates.
`-p, --package`	Updates the packages installed inside the Container by using the `vzpkg install` command.
`-f, --force`	Forces the EZ template/package update procedure.
`-n, --check-only`	Simulates the same operations as `vzpkg update` completes without specifying this option (downloads the updated packages to the server, handles their interdependencies, etc.); however, the packages themselves are not updated.
`-d, --debug <num>`	Sets the debugging level to one of the specified values (from 0 to 10). 10 is the highest debug level and 0 sets the debug level to its minimal value.
`-q, --quiet`	Disables logging to the screen and to the log file.

Without any option specified, `vzpkg update` updates all EZ templates (including the OS EZ template) inside the Container with the ID of <*CT_ID*> or the name of <*CT_NAME*>. However, you can make the command update a particular EZ template by specifying its name as <*object*>. You can also use the `-p` option to make the command update the packages installed inside the Container by using `vzpkg install` instead of EZ templates.

vzpkg remove

This command is used to remove an application EZ template or a software package from a Container. It has the following syntax:

```
vzpkg remove [options] <CT_ID>|<CT_NAME> <object> [...]
```

This command will remove an object (`object`) which can be either an application EZ template or a standard software package (if the `-p` option is specified) installed with the `vzpkg install` command from the Container with the ID of *`CT_ID`* or with the name of <*`CT_NAME`*>. You may specify a number of objects for removing.

The options available to this command are:

Name	Description
`-p, --package`	Removes the specified package(s) from the Container.
`-w, --with-depends`	Removes also the packages having dependencies with the object specified.
`-f, --force`	Forces the EZ template/package deletion.
`-n, --check-only`	Simulates the same operations as `vzpkg remove` completes without specifying this option (handles interdependencies of the packages to be removed from the server, etc.); however, the packages themselves are not deleted from the specified Container(s).
`-d, --debug <num>`	Sets the debugging level to one of the specified values (from 0 to 10). 10 is the highest debug level and 0 sets the debug level to its minimal value.
`-q, --quiet`	Disables logging to the screen and to the log file.

By default, the specified object is treated by `vzpkg remove` as an application EZ template. However, you can use the `-p` option to explicitly specify that the object should be considered as a software package.

Note: A Container has to be running in order to remove an application EZ template/package from it.

vzpkg link

If an application (or its update) was directly installed inside a Container, whereas a compatible application EZ template is also installed on the server, the Container can be linked to this template, so the real files inside the Container are replaced with symlinks to these very files on the server. The `vzpkg link` command has the following syntax:

```
vzpkg link [options] <CT_ID>|<CT_NAME>
```

The options that can be used with this command are the following:

Option	Description
`-s, --slow`	Check all packages inside the Container including the ones installed by using the technology of Parallels Server Bare Metal templates and replace the real application files, if any, with symlinks to the files on the server.
`-C, --cache`	Makes the `vzpkg link` command look for the packages in the local `vzpkg` cache only. If there is a package not available locally, the command will fail. You can omit this parameter if the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file; in this case `vzpkg link` will also check the local `vzpkg` cache only.
`-r, --remote`	If the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the`/etc/vztt/vztt.conf` file, you should use this option to make `vzpkg link` look for the packages in the remote repositories set for handling the corresponding EZ templates.
`-d, --debug <num>`	Sets the debugging level to one of the specified values (from 0 to 10). 10 is the highest debug level and 0 sets the debug level to its minimal value.
`-q, --quiet`	Disables logging to the screen and to the log file.

vzpkg create cache

This command is used to create tarballs (caches) for OS EZ templates. You should execute this command before you can start using a newly installed OS EZ template for creating Containers. It has the following syntax:

```
vzpkg create cache [options] [<OS_template> [...]]
```

You can use the following options with this command:

Name	Description
`-C, --cache`	Makes the `vzpkg create cache` command check for the packages included in the EZ OS template in the local `vzpkg` cache only and use them for the cache creation. You can omit this parameter if the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file. In this case `vzpkg create cache` will also check the local `vzpkg` cache only.
`-r, --remote`	If the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file, you should use this option to make `vzpkg create cache` check for the packages included in the EZ OS template in the remote repositories set for its handling.
`-d, --debug <num>`	Sets the debugging level to one of the specified values (from 0 to 10). 10 is the highest debug level and 0 sets the debug level to its minimal value.
`-q, --quiet`	Disables logging to the screen and to the log file.
`-f, --force`	Forces the process of the cache creation.

`vzpkg create cache` checks the template area on the server (by default, the `/vz/template` directory is used) and if it finds an OS EZ template for which no tar archive exists, it creates a gzipped tarball for the corresponding OS EZ template and places it to the `/vz/template/cache` directory. When a Container is being created, `pctl` just unpacks the tar archive.

By default, `vzpkg create cache` checks the tar archive existence for all OS EZ templates installed on the server and creates some, if necessary. However, you can explicitly indicate what OS EZ template should be cached by specifying its name as <*OS_template*>. If the cache of the OS template specified already exists on the server, the command will fail and you will be presented with the corresponding error message.

vzpkg update cache

This command is used to update tarballs (caches) of the OS EZ templates installed on the server. It has the following syntax:

```
vzpkg update cache [options] [<OS_template> [...]]
```

You can pass the following options to this command:

Name	Description
`-C, --cache`	Makes the `vzpkg update cache` command check for the packages updates in the local `vzpkg` cache only and use them for the cache creation. You can omit this parameter if the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file. In this case `vzpkg update cache` will also check the local `vzpkg` cache only.
`-r, --remote`	If the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file, you should use this option to make `vzpkg update cache` check for the packages updates in the remote repositories set for handling the given EZ OS template.

`vzpkg update cache` checks the cache directory in the template area (by default, the template area is located in the `/vz/template` directory on the server) and updates all existing tarballs in this directory. However, you can explicitly indicate what OS EZ template tarball is to be updated by specifying its name as <*`OS_template`*>. Upon the `vzpkg update cache` execution, the old tarball is renamed by receiving the `-old` suffix (e.g. `redhat-el5-x86.tar.gz-old`).

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

vzpkg remove cache

This command removes the cache for the OS EZ templates specified. It has the following syntax:

```
vzpkg remove cache [options] [<OS_template> [...]]
```

You can use the following options with this command:

Name	Description
`-d, --debug <num>`	Sets the debugging level to one of the specified values (from 0 to 10). 10 is the highest debug level and 0 sets the debug level to its minimal value.
`-q, --quiet`	Disables logging to the screen and to the log file.

By default, `vzpkg remove cache` deletes all caches located in the `/vz/template/cache` directory on the server. However, you can explicitly indicate what OS EZ template tar archive is to be removed by specifying its name as <*`OS_template`*>.

Note: The OS EZ template caches having the `-old` suffix are not removed from the `/vz/template/cache` directory. You should use the `-rm` command to delete these caches from the server.

vzpkg localinstall

The `vzpkg localinstall` command is used to install a software package inside a Container from the corresponding file on the server. It has the following syntax:

```
vzpkg localinstall [options] <CT_ID>|<CT_NAME> rpm_file_path [...]
```

Options available to this command are:

Name	Description
`-C, --cache`	When handling the package interdependencies, makes the `vzpkg localinstall` command look for the needed packages in the local `vzpkg` cache only. If there is a package not available locally, the command will fail. You can omit this parameter if the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file.
`-r, --remote`	If the elapsed time from the last `vzpkg` local cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file, you should use this option to make `vzpkg localinstall` look for the packages in the remote repository.
`-n, --check-only`	Simulates the same operations as `vzpkg localinstall` completes without specifying this option (e.g. handles the package interdependencies); however, the package itself is not installed in the specified Container.
`-d, --debug <num>`	Sets the debugging level to one of the specified values (from 0 to 10). 10 is the highest debug level and 0 sets the debug level to its minimal value.
`-q, --quiet`	Disables logging to the screen and to the log file.

When executed, the command installs the package, the full path to which is specified as *`rpm_file_path`*, inside the Container with the ID of *`<CT_ID>`* or with the name of *`<CT_NAME>`*. You may specify a number of packages at once to be installed inside the Container.

During its execution, `vzpkg localinstall` automatically handles the interdependencies among the packages to be installed inside a Container and ensures that all dependencies are satisfied. If the package dependencies cannot be resolved, the installation process will fail and the corresponding message will be displayed.

vzpkg localupdate

The `vzpkg localupdate` command is used to update the software packages installed inside your Container(s) by means of the `vzpkg install` or `vzpkg localinstall` commands. It has the following syntax:

```
vzpkg localupdate [options] <CT_ID>|<CT_NAME> rpm_file_path [...]
```

Options available to this command are:

Name	Description
`-C, --cache`	When handling the package interdependencies, makes the `vzpkg localupdate` command look for the needed packages in the local `vzpkg` cache only. If there is a package not available locally, the command will fail. You can omit this parameter if the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file.
`-r, --remote`	If the elapsed time from the last `vzpkg` local cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file, you should use this option to make `vzpkg localupdate` look for the packages in the remote repository.
`-n, --check-only`	Simulates the same operations as `vzpkg localupdate` completes without specifying this option (e.g. handles the package interdependencies); however, the package itself is not installed in the specified Container.
`-d, --debug <num>`	Sets the debugging level to one of the specified values (from 0 to 10). 10 is the highest debug level and 0 sets the debug level to its minimal value.
`-q, --quiet`	Disables logging to the screen and to the log file.

When executed, `vzpkg localupdate` compares the file on the server the full path to which is specified as *`rpm_file_path`* with the corresponding package inside the Container with the ID of <*CT_ID*> or the name of *<CT_NAME>* and updates it, if necessary. You may specify a number of packages at once to be updated inside your Container.

vzpkg upgrade

The `vzpkg upgrade` command is used to upgrade an OS EZ template the Container is based on to a newer version. It has the following syntax:

```
vzpkg upgrade [options] <CT_ID>|<CT_NAME>
```

You can use the following options with this command:

Name	Description
`-C, --cache`	Makes the `vzpkg upgrade` command check for the packages included in the OS EZ template in the local `vzpkg` cache only. If any package is not available locally, the command will fail. You can omit this parameter if the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file; in this case `vzpkg upgrade` will also check the local `vzpkg` cache only.
`-r, --remote`	If the elapsed time from the last local `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file, you should use this option to make `vzpkg upgrade` check for the packages in the remote repositories set for handling the given EZ OS template.
`-n, --check-only`	Simulates the same operations as `vzpkg upgrade` completes without specifying this option (downloads the packages to the server, handles their interdependencies, etc.); however, the packages themselves inside the Container are not upgraded.
`-f, --force`	Forces the process of upgrading the OS EZ template.
`-d, --debug <num>`	Sets the debugging level to one of the specified values (from 0 to 10). 10 is the highest debug level and 0 sets the debug level to its minimal value.
`-q, --quiet`	Disables logging to the screen and to the log file.

vzpkg fetch

This command is used to download packages included in the corresponding OS EZ template or their updates from the remote repository to the `vzpkg` local cache on the server and to prepare them for the installation. It has the following syntax:

```
vzpkg fetch [options] <OS_template>
```

You can pass the following options to `vzpkg fetch`:

Option	Description
`-O, --os`	Download packages/updates for the specified EZ OS template.
`-A, --app`	Download packages/updates for EZ application templates used with the EZ specified OS template.
`-C, --cache`	Makes the `vzpkg fetch` command look for the metadata in the `vzpkg` local cache only. You can omit this parameter if the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file.
`-r, --remote`	If the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file, you should use this option to make `vzpkg fetch` look for the OS EZ template metadata in the remote repositories set for handling the corresponding EZ template.
`-f, --force`	Forces the process of downloading packages and/or their updates to the server.
`-d, --debug <num>`	Sets the debugging level to one of the specified values (from 0 to 10). 10 is the highest debug level and 0 sets the debug level to its minimal value.
`-q, --quiet`	Disables logging to the screen and to the log file.

You can make `vzpkg fetch` run as a `cron` job (e.g. nightly) checking for available packages or packages updates for your EZ templates and keeping them in the local cache. Having all the necessary packages in the `vzpkg` local cache can greatly speed up the execution of the `vzpkg install`, `vzpkg update`, or `vzpkg create cache` commands since the packages are available locally and there is no need to check for them in the corresponding remote repositories.

vzpkg clean

This command is used to remove the software packages, their headers, and metadata downloaded to the server from the repository during the `vzpkg` execution (e.g. while caching an OS EZ template or adding an application EZ template to a Container for the first time). It has the following syntax:

```
vzpkg clean [options] [<OS_template> [...]]
```

You can use the following options with `vzpkg clean`:

Name	Description
`-k, --clean-packages`	Removes the packages, headers, and metadata of the specified EZ OS template from the local `vzpkg` cache. This is also the default behaviour of `vzpkg clean`.
`-t, --clean-template`	Checks the template area for the specified EZ OS template (the template area has the default path of `/vz/template`) and removes all packages that are currently not used by any Container on the server and not included in the EZ OS template cache.
`-a, --clean-all`	Removes both: ▪ the packages, headers, and metadata of the specified EZ OS template from the `vzpkg` local cache and ▪ the packages that are currently not used by any Container on the server and not included in the EZ OS template cache.
`-f, --force`	Forces the `vzpkg clean` execution.
`-n, --check-only`	Simulates the same operations as `vzpkg clean` completes without specifying this option; however, the packages and headers are not removed from the server.
`-d, --debug <num>`	Sets the debugging level to one of the specified values (from 0 to 10). 10 is the highest debug level and 0 sets the debug level to its minimal value.
`-q, --quiet`	Disables logging to the screen and to the log file.

vzpkg update metadata

This command is used to update the OS EZ template local metadata on the server. It has the following syntax:

```
vzpkg update metadata [options] [OS_template ...]
```

The following options can be used with `vzpkg update metadata`:

Name	Description
`-C, --cache`	Makes the `vzpkg update metadata` command look for available metadata updates in the local `vzpkg` cache only. You can omit this parameter if the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file.
`-r, --remote`	If the elapsed time from the last `vzpkg` cache update does not exceed the value of the `METADATA_EXPIRE` parameter specified in the `/etc/vztt/vztt.conf` file, you should use this option to make `vzpkg update metadata` look for the updated metadata in the remote repositories set for handling the corresponding OS EZ template.
`-d, --debug <num>`	Sets the debugging level to one of the specified values (from 0 to 10). 10 is the highest debug level and 0 sets the debug level to its minimal value.
`-q, --quiet`	Disables logging to the screen and to the log file.

When executed without any options, the command updates the metadata of all OS EZ templates installed on the server. If you specify one or more OS EZ templates, the command will update the metadata of the indicated OS templates only. You can run this command a cron job at regular intervals to be sure that your OS EZ templates metadata are always up-to-date.

vzmktmpl

This utility is used to create new EZ templates. It has the following syntax:

```
vzmktmpl [options] metafile
```

The following options can be passed to `vzmktmpl`:

Name	Description
`--pre-cache` *`file`*	The path to the script which will be executed by the `vzpkg cache` command before installing the packages included in the EZ template on the server. This script is executed in the server context and relevant for OS EZ templates only.
`--post-cache` *`file`*	The path to the script which will be executed by the `vzpkg cache` command after installing the packages included in the EZ template on the server. This script is executed in the server context and relevant for OS EZ templates only.
`--pre-install` *`file`*	The path to the script which will be executed by the `vzpkg install` command before adding the application EZ template to the Container. This script is executed in the Container context and relevant for application EZ templates only.
`--post-install` *`file`*	The path to the script which will be executed by the `vzpkg install` command after adding the application EZ template to the Container. This script is executed in the Container context and relevant for application EZ templates only.
`--pre-upgrade` *`file`*	The path to the script which will be executed by the `vzpkg upgrade` command before upgrading the OS EZ template inside the Container. This script is executed in the Container context.
`--post-upgrade` *`file`*	The path to the script which will be execute by the `vzpkg upgrade` command after upgrading the OS EZ template inside the Container. This script is executed in the Container context.
`--pre-update` *`file`*	The path to the script which will be executed by the `vzpkg update` command before updating the packages included in the application EZ template inside the Container. This script is executed in the Container context.
`--post-update` *`file`*	The path to the script which will be executed by the `vzpkg update` command after updating the packages included in the application EZ template inside the Container. This script is executed in the Container context.
`--pre-remove` *`file`*	The path to the script which will be executed by the `vzpkg remove` command before removing the application EZ template from the Container. This script is executed in the Container context and relevant for application EZ templates only.
`--post-remove` *`file`*	The path to the script which will be executed by the `vzpkg remove` command after removing the application EZ template from the Container. This script is executed in the Container context and relevant for application EZ templates only.
`--environment` *`file`*	The path to the file storing a list of environment variables. The variables should be set in the form of *`key=value`*. The variables specified in this file are used when running the `vzpkg create cache` and `vzpkg update cache` commands and exported to

	the Container environment during the EZ template scripts execution.
`-d, --doc` *`file`*	The path to the file containing the information on the EZ template. You can specify several files and separate them by commas.
`-s, --spec-only`	Creates the package specification file only.
`-r, --srpm`	Creates the package source file only.
`-h, --help`	Displays the utility usage and exits.

The utility requires only the metafile to create an EZ template and save it as a software package (see the next subsection for information on EZ template metafiles). However, you can set a number of scripts to be executed on different stages of the EZ template lifecycle (e.g. upon its installing on the server or after its removing from the Container) or use other options listed in the table above.

Note: We recommend that you use the sample scripts located in the `/usr/share/vztt/samples` directory on your server as the basis for creating your own scripts.

vzpkg.metafile

This file is used by the `vzmktmpl` utility as the basis for the EZ template creation. The parameters in this file are presented on separate lines in the following format:

```
<parameter_name>
<parameter_value>
```

The table below describes these parameters:

Name	Description
`%osname`	Mandatory. The name of the Linux distribution for which you are creating the OS EZ template or under which the application EZ template being created is to be run.
`%osver`	Mandatory. The version of the Linux distribution specified as the value of the `%osname` parameter.
`%osarch`	Mandatory. The microprocessor architecture where the EZ template is to be run. You can set the value of this 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).
`%appname`	Mandatory, for application EZ templates only. The name of the application EZ template.
`%setname`	Optional. The name of the non-base OS EZ template, if any. This parameter should be specified only while creating non-base OS EZ templates.
`%upgradable_version`	Optional. A list of Linux distribution versions which can be upgraded to the version of the Linux distribution for which you are creating the EZ template. For OS EZ templates only.
`%packages`	Mandatory. A list of software packages to be included in the EZ template. The names of the packages listed as the value of this parameter should correspond to the names of real packages that are stored in the repository used for managing your EZ templates and can be specified in one of the following ways: For RPM-based Linux distributions: ▪ as a package name only (e.g. `wget`) ▪ as a package name with the indication of the system architecture on which the package is to be run (e.g. `wget.i386`, `wget.noarch`) ▪ as a package name with its versions (e.g. `wget-1.9.1`) ▪ as a package name with its versions and release number (e.g. `wget-1.9.1-12`) ▪ as a package name with its version, release number, and system

	architecture (e.g. `wget-1.9.1-12.i386`) ▪ as a package name with its version, release number, system architecture, and epoch number (e.g. `10:wget-1.9.1-12.i386`) For Debian-based Linux distributions: ▪ as a package name only (e.g. `wget`) ▪ as a package name with its version (e.g. `wget-1.9.1-12`)
`%packages_0`	Mandatory, for Debian-based OS EZ templates only. A list of packages to be used for creating a minimal Debian/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.
`%packages_1`	Mandatory, for Debian-based OS EZ templates only. A list of 'base' packages for the Debian/Ubuntu distribution. These packages are needed to install the packages listed as the value of the `%packages` parameter.
`%package_manager`	Mandatory. The short name of the package manager to be used for handling the EZ template. 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 `PKGMAN` parameter: *32-bit Linux distributions*: ▪ `rpm44x86`: Red Hat Enterprise Linux 5, Fedora Core 4, 5, and 6, Fedora 7 and 8 ▪ `rpm43x86`: Red Hat Enterprise Linux 3 and 4 (with the 2.6 kernel and NPTL support) and CentOS 4, 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 and Fedora Core 4, 5, and 6, Fedora 7 and 8 ▪ `rpm43x64`: Red Hat Enterprise Linux 3 and 4 (with the 2.6 kernel and NPTL support) and CentOS 4, 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, 5 ▪ `rpm41s9i64`: SUSE Linux Enterprise Server 9

	▪ `rpm41i64`: SUSE Linux Enterprise Server 10 ▪ `dpkgi64`: Debian and Ubuntu This parameter should be obligatorily specified for all base OS EZ templates and can be omitted for application EZ templates and non-base OS EZ templates.
`%repositories`	Mandatory, for RPM-based Linux distributions only. A list of repositories where the packages comprising the EZ template are stored.
`%mirrorlist`	Mandatory. One or several URLs to the file containing a list of repositories from where the packages comprising the EZ template are to be downloaded. This parameter can be omitted if you are creating a metafile for an application EZ template or a non-base OS EZ template.
`%distribution`	Optional. The type of the Linux distribution. Examples of Linux distribution types are `centos`, `debian`, `fedora-core`, `gentoo`, `mandrake`, `redhat`, `rhel-3, rhel-4, rhel-5`, `fedora-core-4`, `fedora-core-5`, `slackware`, `slackware-10.0`, `suse`, `suse-9.3`, etc.
`%description`	Optional. Detailed information on the EZ template package file.
`%version`	Optional. The version of the EZ template package file.
`%release`	Optional. The release of the EZ template package file.
`%license`	Optional. The information about the owner of the EZ template package file.
`%changelog`	Optional. The information about the changes made to the EZ template package file.

vzpkgproxy

The `vzpkgproxy` utility is used to set up a caching proxy server meant for handling your OS and application EZ templates. It has the following syntax:

```
vzpkgproxy
```

The `vzpkgproxy` package where the `vzpkgproxy` utility is included can be installed by using the `rpm -i` command on any computer meeting the following requirements:

- The Apache `httpd` server, version 2.0.52 and higher, should be installed on the workstation.
- The `createrepo` package, version 0.4.2 and higher, should be installed on the workstation.

During its installation, the utility performs all the tasks necessary to install, configure, and put into operation your caching proxy server.

vzrhnproxy

The `vzrhnproxy` utility is used to set up a Red Hat Network (RHN) Proxy Server meant for handling the packages included in the RHEL 4 and 5 OS EZ templates. It has the following syntax:

```
vzrhnproxy register OS_arch OS_name server_hostname
            server1_IP_Address [server2_IP_Address ...]
vzrhnproxy list
vzrhnproxy update [profile_name ...]
vzrhnproxy help
```

The `vzrhnproxy` utility can be installed with the `rpm -i` command on any 'RHEL 4'-based server (e.g. RHEL 4 and RHEL 5, Fedora 7 and 8, or CentOS 4 and 5). To start using `vzrhnproxy` for creating an RHN Proxy Server, you should specify valid credentials in the `/etc/vz/rhnproxy/rhn.conf` file to log in to RHN and run the `vzrhnproxy register` command on the server where you wish to create the RHN Proxy Server. This command will:

1 Connect to Red Hat Network with the credentials specified in the previous step.

2 Register in RHN and create a system profile for the server with:

- the hostname of *`HN_hostname`*
- the OS name of *`OS_name`*. In the current version of Parallels Server Bare Metal, this name can be set to one of the following:

  * `4AS` for Red Hat Enterprise Linux 4 Advanced Server

  * `4ES` for Red Hat Enterprise Linux 4 Edge Server

  * `4WS` for Red Hat Enterprise Linux 4 Workstation

  * `4Desktop` for Red Hat Enterprise Linux 4 Desktop

  * `5Server` for Red Hat Enterprise Linux 5 Server

  * `5Client` for Red Hat Enterprise Linux 5 Desktop
- the system architecture of *`OS_arch`* which can be one of the following: `i386` for 32-bit versions of RHEL 4 and 5, `x86_64` for x86-64-bit versions of RHEL 4 and 5, or `ia64` for IA64-bit versions of RHEL 4 and 5

3 Download the headers of the packages comprising the corresponding RHEL distribution to the RHN Proxy Server.

4 Create a pseudo-repository containing the repodata generated on the basis of the downloaded headers.

5 Grant the server with the IP address of *`server1_IP_Address`* access to the RHN Proxy Server. You can specify several IP addresses to allow several servers to use this Proxy Server.

The `vzrhnproxy list` command displays a list of all system profiles you have registered with Red Hat Network.

The `vzrhnproxy update` command updates the repodata in the pseudo-repository on the Proxy Server for the specified system profile (*`profile_name`*); you can specify more than one system profile whose repodata are to be updated.

Supplementary Tools

vzfsutil

This command is used for checking the VZFS consistency, correcting and optimizing the Container private area, upgrade the Container private area from VZFS 1 to VZFS 2. It has the following syntax:

```
vzfsutil [general_options] -t template_path private_path
         <action_options>
```

A Container private area consists of number of VZFS symlinks to templates and when the Container is running, these VZFS symlinks are visible as regular files inside the Container. When the user inside the Container changes one of the VZFS symlink files, a file with a special name and with the changed content is created in the special copy-on-write directory (`$VE_PRIVATE/cow`) of the Container private area. However, when the user creates a file inside the Container, the file is created in the root directory (`$VE_PRIVATE/root`) of the private area (`$VE_PRIVATE`).

This utility optimizes the private area by moving copy-on-write files to their actual location in the root directory of the private area. It also checks the correctness of VZFS symlinks and fixes found inconsistencies.

The required parameters for this utility invocation are:

`-t` *`template_path`*	Path to the directory where templates are installed. As a rule, it is `/vz/template`.
`<`*`action_options`*`>`	One or more actions described below.
`private_path`	Path to the Container private area.

General options.

`-v`	Verbose mode. Causes the utility to print debugging messages about its progress. You can give up to two `–v` switches to increase verbosity.
`-q`	Quiet mode. Causes all warning and diagnostic messages to be suppressed. Only fatal errors are displayed.
`-i`	Interactive mode. When this option is given, the utility asks for confirmation before correcting any inconsistency. This option works only with check actions.
`-p, --progress`	Display (at certain intervals) the information on the operations currently performed by `vzfsutil`.
`-V`	Display the utility version.
`--upgrade`	Upgrades the Container private area from VZFS v1 to VZFS v2. You can also use `--ctid` with this option to perform an additional check of the Container private area.
`--ctid=`*`CT_ID`*	Performs an additional check for the Container with the specified ID.

Action options.

Action options can be checking and optimizing. Checking actions have the format of `--<option>[=<action>]` where `<action>` is one of the following:

`i, ignore`	Do not repair found inconsistency, only report it. This is the default action if no action is explicitly specified.
`m, move`	Move inconsistent VZFS symlinks into the `$VE_PRIVATE/lost_files` directory.
`r, remove`	Remove inconsistent VZFS symlinks.

Options can be one or several of the following:

`--call,--cA`	Correct all found errors. This options turn on and set action on all type of corrections. However, you can specify different action on a particular type of correction by adding specific option. For example, you may want to report only all inconsistencies except the orphaned file in copy-on-write area, which you want to remove. In that case you have to issue the command with `––call=i ––ccow=r` action options.
`--ccow,--cC`	Correct orphaned files in the copy-on-write directory `$VE_PRIVATE/cow`. The file is considered to be orphaned when there is no VZFS symlink in the Container private area referring to the file.
`--cmark,--cM`	Correct broken copy-on-write mark on the VZFS symlink. VZFS uses sticky bit on the VZFS symlink when file is being copied as a mark that copy-on-write directory has a new file for the symlink. The mark is considered to be broken when copy-on-write area contains no file for the VZFS symlink. If this option is used with the `remove` action then VZFS symlink is not removed but the mark (sticky bit) is turned off on the VZFS symlink.
`--clink,--cL`	Correct broken VZFS symlink. VZFS symlink is considered to be broken when it points to non-existent or non-regular file in the template directory.

Optimization actions have the format of `--<option>[=<action>]` where `<action>` can be one of the following:

`i, ignore`	Do not do actual optimization, only report what can be optimized. This is the default action if no action is explicitly specified.
`d, do`	Proceed with the optimization.

Options can be one or several of the following:

`--oall,--oA`	Do all types of optimizations.
`--oreplace,--oR`	Replaces VZFS symlinks in the Container private area with the files found in the copy-on-write directory.
`--oempty,--oE`	Compares the content of the file in the copy-on-write directory with the file from the template directory and if they are the same, removes the file from the copy-on-write directory. It also turns off the copy-on-write mark (sticky bit) from the corresponding VZFS symlink.

The example below illustrates the usage of `vzfsutil` for getting rid of files in the copy-on-write directory of Container 101.

```
# ls /vz/private/101/cow
02c38bc341d382c11ecb8140b2811ea3  81294809f6694940b2b8bd321ceda09e
1052304dc986a695b12d34eaf1324976  823500b4147c2d3a30e4478faee48550
2dfb1f92fc775fc85d7898391d72080d  96c61677e816f3433e3f0eeb6e539380
410fb765985f96d07b5e90b875cb325f  a5f1c84a031fcf70743896779758a181
57ec3d1ee07a848f883ca6ba4511e9bc  b46db38473ff5b6082c1803f0cc1f59c
# vzfsutil -t /vz/template --oall=d /vz/private/101
```

```
Optimization: 'cow' '57ec3d1ee07a848f883ca6ba4511e9bc' \
different from 'template' 'redhat-el5-x86/initscripts-6.43-1/sbin/ifup'
Optimized: replaced 'magic' 'sbin/ifup' with 'cow' \
'57ec3d1ee07a848f883ca6ba4511e9bc'
Optimization: non-changed 'cow' '410fb765985f96d7b5e90b875cb325f' \
for 'etc/printcap'
Optimized: removed non-changed 'cow' '410fb765985f96d7b5e90b875cb325f'
 [further output suppressed]
# ls -l /vz/private/101/cow
total 0
```

Caution: Potentially, with incorrect usage (for example, if you specify the `remove` action with non-interactive mode and incorrect template area), this utility may destroy VZFS private area. Use this utility with care. It is highly recommended to run it in the “report-only” mode before making any changes.

vzcache

The `vzcache` utility scans the specified Containers for common files and caches these files in the server template area (`/vz/template/vc` by default), replacing the real files inside the Containers with symlinks to the template area. In the case of a significant number of identical files, using this utility results in a notable disk space gain. `vzcache` has the following syntax:

```
vzcache [options] CT_ID-list
```

The following command-line options can be used with the `vzcache` utility:

Option	Description
`-h`, `--help`	Print the usage information.
`--version`	Display the utility version.
`-v`, `--verbose`	Verbose mode. Causes `vzcache` to print debugging messages about its progress. Multiple `-v` options increase verbosity; the maximal number is 2.
`-q`, `--quiet`	Quiet mode. Print error messages only.
`-r`, `--cachearea`	Process the Container template area only. During the `vzcache` execution, a separate template area (`/vz/template/vc/`*`CT_UUID`* by default) is created for each specified Container. *`CT_UUID`* denotes the Container unique identifier and can be determined by viewing the *`UUID`* parameter in the Container configuration file. You are recommended to use this option when running `vzcache` for migrated or restored Containers.
`-C`, `--clean`	Clears the Container template area (`/vz/template/vc/`*`CT_UUID`* by default) from those cached files that are not present any more inside the corresponding Container. Only one Container can be cleaned up at a time.
`-s`, `--size-limit` *`N`*	Do not process files smaller than *`N`* bytes. By default, only empty files are not processed.

vzps and vztop

These two utilities can be run on the server just as the standard Linux `ps` and `top` utilities. For information on the `ps` and `top` utilities, consult Linux Administrator's Guide or the corresponding man pages. The `vzps` and `vztop` utilities provide certain additional functionality related to monitoring separate Containers running on the server.

The `vzps` utility has the following functionality added:

- The `-E` *`CT_ID`* command-line switch can be used to show only the processes running inside the Container with the specified ID.

The `vztop` utility has the following functionality added:

- The `-E` *`CT_ID`* command-line switch can be used to show only the processes running inside the Container with the ID specified. If `-1` is specified as *`CT_ID`*, the processes of all running Containers are displayed.
- The `e` interactive command (the key pressed while `top` is running) can be used to show/hide the `CTID` column, which displays the Container where a particular process is running (`0` stands for the server itself).
- The `E` interactive command can be used to select another Container the processes of which are to be shown. If `-1` is specified, the processes of all running Containers are displayed.

vzsetxinetd

To switch the service running in a particular Container between an independent and `xinetd`-dependent modes, the `vzsetxinetd` utility is used. It has the following syntax:

```
vzsetxinetd -h
vzsetxinetd -s CT_ID SERVICE ...
vzsetxinetd [-u] [-f] CT_ID SERVICE on|off ...
```

where *`CT_ID`* is the ID of the corresponding Container, and *`SERVICE`* is the corresponding service: `sendmail`, `sshd`, `proftpd`, or `courier-imap`. There may be any number of the *`SERVICE`* or *`SERVICE`* `on|off` strings in a single invocation of the `vzsetxinetd` utility. With "`on`", the service will be based on `xinetd`; if "`off`" is specified, it will be standalone.

Notes:

1. The `courier-imapd`, `courier-imapds`, `courier-pop3d`, and `courier-pop3ds` services are provided by the `courier-imap` service, thus `vzsetxinetd` can manage these services via the `courier-imap` service.

2. The Parallels HSPcomplete application cannot be used for managing Containers having one or more services configured with the `vzsetxinetd` utility.

Here follows the explanation of available options:

`-h`, `--help`	Prints help on the usage of the utility.
`-s`, `--state`	Prints the information on whether the corresponding service is `xinetd`-dependent or standalone.
`-u`, `--userfiles`	This option tells the utility to save the previously saved user files related to the service.
`-f`, `--force`	This option tells the utility to set the service to the specified mode even in case the service is currently in this mode already.

vzdqcheck

This utility counts inodes and disk space used using the same algorithm as Parallels Server Bare Metal quota. It has the following syntax:

```
vzdqcheck [options] <path>
```

The command traverses directory tree given as the `path` argument and calculates space occupied by all files and number of inodes. The command does not follow mount points and correctly handles VZFS symlinks unlike the standard `du` command.

Options available to the `vzdqcheck` command are:

`-h`	Usage info.
`-V`	`vzquota` version info.
`-v`	Verbose mode.
`-q`	Quiet mode.

vzdqdump and vzdqload

The `vzdqdump` and `vzdqload` utilities are used for dumping the Container user/group quota limits and grace times from the kernel or the quota file or for loading them to a quota file, respectively. `vzdqdump` displays the corresponding values on the console screen, and `vzdqload` gets the information from the standard input.

The syntax of the commands is the following:

```
vzdqdump [general_options] quota_id [-f] [-c quota_file] –G|-U|-T
vzdqload [general_options] quota_id [-c quota_file] –G|-U|-T
```

The general options are described in the table below:

`-h`	Usage info.
`-V`	`vzquota` version info.
`-v`	Verbose mode.
`-q`	Quiet mode.

The *`quota_id`* parameter corresponds to the ID of the Container for which you wish to dump/load the quotas. Other options are the following:

`-f`	Dump the user/group quota information from the kernel rather than from the quota file
`-c quota_file`	Specifies a quota file to process other than the default quota file (`var/vzquota/quota.<`*`CT_ID`*`>`)
`-G, --grace`	Dump/load user/group grace times
`-U, --limits`	Dump/load user/group disk limits
`-T, --exptimes`	Dump/load user/group expiration times

Quotas must be turned off when the `vzdqload` utility is working. Mind that only 2nd-level disk quotas are handled by the utilities.

vznetstat

This utility outputs traffic usage statistics for virtual machines and Containers. It has the following syntax:

```
vznetstat [-v <ID>] [-c <class>] [-a] [-r]
```

The utility displays input and output traffic for virtual machines and Containers for each defined network class. The network classes are described in the `/etc/vz/conf/networks_classes` file. If no options are specified the network statistics for all running virtual machines and Containers is printed.

The utility accepts the following options:

`-v <ID>`	Display statistics for virtual machines and Containers with the ID of `<ID>`. Multiple `–v` options can be given to a single `vznetstat` invocation.
`-c <class>`	Show the network statistics for the `<class>` class only.
`-a`	Display statistics for all classes.
`-r K|M|G`	Display the network statistics, which is shown in bytes by default, in the following measurement units: ▪ `K`: display the network statistics in kilobytes ▪ `M`: display the network statistics in megabytes ▪ `G`: display the network statistics in gigabytes
`--help`	Display the utility usage information.

vzcpucheck

This utility displays the current server utilization in terms of allocated CPU units as well as total server CPU units capacity. It has the following syntax:

```
vzcpucheck [-v]
```

Without arguments, the utility prints the sum of CPU units of all running Containers and the total server capacity. If the `–v` option is given, the utility prints per Container CPU units information.

vzmemcheck

This utility shows the server memory parameters: low memory utilization, low memory commitment, RAM utilization, memory+swap utilization, memory+swap commitment, allocated memory utilization, allocated memory commitment, allocated memory limit. It has the following syntax:

```
vzmemcheck [-v] [-A]
```

The following options can be specified in the command-line:

`-v`	Display information for each Container.
`-A`	Display absolute values (in megabytes).

It is possible to use any of the available options, both of them, or to do without any options.

vzcalc

This utility is used to calculate Container resource usage. It has the following syntax:

```
vzcalc [-v] <CT_ID>
```

This utility displays what part of server resources Container `<CT_ID>` is using. An optional `–v` switch produces verbose output including number of processes, low memory, allocated memory and memory and swap statistics.

For stopped Containers the utility displays promised and maximum values the Container can consume. For running Containers, it also outputs the current values.

The high values of resource usage means that either the server is overcommitted or Container configuration is invalid.

vzcheckovr

This utility is used to check the current system overcommitment and safety of the total resource control settings. It runs as follows:

```
vzcheckovr [-v]
```

where `–v` is the option for verbose output. This utility computes the commitment levels of a number of resource management parameters (Low Memory, Memory + Swap, Allocated Memory) and compares them with the values chosen by the Parallels Server Bare Metal administrator in the `/etc/vz/vz.conf` global configuration file. The utility will produce a warning if these configured values are exceeded. Similarly to `vzmemcheck`, `vzcheckovr` takes into account only those Containers that are currently running and ignores all the others existing on the server.

vzstat

This utility is for real-time monitoring in Parallels Server Bare Metal. It displays the status and load of the system pertaining to its disk, network, CPU, and memory (including swap) parameters, updating this status with the preset time interval. It also provides a list of running Containers together with their resources consumption statistics, and can sort this list by a number of parameters. The utility has an interactive interface for setting the mode of displaying the information.

The syntax of the `vzstat` utility is the following:

```
vzstat [-l] [-d X] [-p CT_ID] [-b|-v] [-t]
```

Here is the description of the command-line parameters:

`-l`	Print information once and exit immediately.
`-d` *`delay`*	Specifies the delay between screen updates. Can be changed on the fly by the `t` interactive command. Default is 1 sec.
`-p` *`CT_ID`*	Monitor only Containers with specified *`CT_ID`*. This flag can be given up to twenty times, in form `-p` *`CT_ID`*`1 -p` *`CT_ID`*`2 ....` This option is not available interactively.
`-b`	"Brief" mode. Minimal details level. Shows only one summary line about each monitoring subsystem. By default, "standard" details level is in use. Valid levels are "brief", "standard" and "verbose". Can be set on the fly by the `b` interactive command. See also the `-v` command-line option and `s` and `v` interactive commands.
`-v`	"Verbose" mode. Provides maximum details about all monitored subsystems. Can be set on the fly by the `v` interactive command. See also the `-b` command-line option and `b` and `s` interactive commands.
`-t`	Text mode, provides information once. It is printed in terse form, suitable for parsing by other programs. All output data are not aligned and numbers are not in a human readable format. In the text mode, there are no colors in the output and only the top 10 Containers sorted by their CPU usage are shown.

`vzstat` is able to display the following information:

Type of Information	Description	Example	Toggling by
Uptime	This line displays the time for which the system has been up, and three "load averages" for the system. The load averages are the average number of processes ready to run during the last 1, 5 and 15 minutes. This line is just like the output of uptime(1).	`1:22am, up 1:31, 2 users, load average: 0.00, 0.06, 0.33`	`l`
Containers and processes	The total number of Containers and processes running at the time of the last update. The output is also broken down into the number of tasks which are running, sleeping, uninterruptable, zombie, or stopped.	`CTNum 102, procs 467: running 12, sleeping 455, unint 0, zombie 0, stopped 0`	`p`
CPU states	Shows the percentage of CPU time used by all Containers (except for Container 0) and by Container 0, the CPU time spent in the user mode, in the system mode, and being idle, and	`CPU [ OK ]: CTs 43%, CT0 12%, user 41%, sys 13%, idle 45%,`	`c`

	the maximal/average scheduling latency in ms. The scheduling latency is the time spent by the processes in the system awaiting for scheduling.	`lat(ms) 3/2`	
Mem	The statistics on the memory usage, including the total available memory, free memory, and maximal/average memory allocation latency. The free memory is displayed both for the low and high memory. The low memory is the sum of the DMA and Normal zones memory and the high memory is the High zone memory. Memory allocation latency is the time required to allocate memory inside the kernel in ms. An excessive allocation latency can be a sign of server's overload.	`Mem [ OK ]: total 755MB, free 671MB/0MB (low/high), lat(ms) 10/7.`	`m, M`
Memory zones information	Information on the memory zones state. This information includes: the total size of the memory zone in MB, the size of active and inactive lists, of the free memory and zone limits.	`ZONE1 (Normal): size 752MB, act 29MB, inact 31MB, free 658MB (0/1/2)`	`m, M`
Memory zones fragmentation	Information on the memory zones fragmentation. This information describes how much system memory is fragmented and which is the biggest block size possible to allocate atomically. The first number before * is a number of blocks and the second is a block size in pages.	`fragm 2*1 3*2 15*4 22*8 25*16 12*32 4*64 0*128 1*256 326*512".`	`m, M`
Memory allocation latency	Memory allocation latency is an average time spent in the kernel memory allocator for different memory type requests. Any memory type is coded as XY, where X is A for GFP_ATOMIC, K for GFP_KERNEL and U for GFP_USER, and Y denotes the allocation request order, i.e. Y=0 for order=0 and 1 for order=1.	`Mem lat (ms): A0 0, K0 0, U0 1, K1 3, U1 2`	`m, M`
Slab cache information	Slab cache information includes: the total slab cache size/real cache size divided into the inode cache size, dentry cache size, buffer heads cache size and page beancounters cache size. The real cache size is the size to which the cache can be shrunk, i.e. it is always less than the total cache size.	`Slab pages: 13MB/13MB (ino 8MB, de 1MB, bh 1MB, pb 0MB)`	`m, M`
Swap	The statistics on the used swap space, including the total swap space, the available swap space and the speed of swap-in/swap-out activity in MB/s.	`Swap [ OK ]: tot 1004MB, free 1004MB, in 0.000MB/s, out 0.000MB/s`	`w, W`
Swap latency	The swap operations latency. This includes the swap-in count, the swap-in maximal/average latency in ms, the swap-out count, the swap-out maximal/average latency in ms, and the maximal/average CPU time spent for the swap-out.	`Swap lat: si 0, 0/0 ms, so 0, 0/0 ms, 0/0 cpu ms`	`w, W`
Swap cache	The swap cache information includes the number of addition, deletion, and find	`Swap cache: add 0, del 0, find`	`w, W`

	operations respecting the swap cache.	`0/0`	
Network information	The network statistics summary includes the total incoming traffic speed in MB/s and incoming packets/s, and outgoing traffic speed in MB/s and outgoing packets/s.	`Net [ OK ]: tot in 1.020MB/s 267pkt/s, out 0.001MB/s 1pkt/s`	`n, N`
Network interface information	Provides the network statistics summary for a particular Ethernet interface, including its total incoming traffic speed in MB/s and incoming packets/s, and outgoing traffic speed in MB/s and outgoing packets/s.	`eth0: in 0.000MB/s 3pkt/s, out 0.001MB/s 1pkt/s`	`n, N`
Disks statistics	The disks statistics summary including the writing and reading activity in MB/s.	`Disks [ OK ]: in 0.000MB/s, out 0.000MB/s`	`d, D`
Mounted disks statistics	The information on the mounted disks such as their mount point, free space, and free inodes left on the device.	`root(/) free: 3489MB(46%), 511077ino(52%)`	`d, D`

Quite a number of single-key interactive commands can be used while `vzstat` is running to alter the way the utility displays information. The commands are not available if `vzstat` runs with the `-t` or `-l` command-line option. These interactive commands are the following:

Key	Action
`h, ?`	Print a help screen.
`space`	Update display immediately.
`q`	Quit.
`t`	Change the delay between screen updates. You will be prompted to enter new delay time, in seconds. Entering 0 causes continuous updates. See also the `-d` command-line parameter.
`b`	Set the "brief" details level. See also the `-b` command-line parameter.
`s`	Set the "normal" details level.
`v`	Set the "verbose" details level. See also the `-v` command-line parameter.
`a`	"Averaged" mode. Monitoring parameters will be averaged through a minute. This includes: 1. Number of uninterruptable processes; 2. Scheduling max latency; 3. Memory allocation max latency; 4. Size of free/active/inactive memory; 5. Swap-in latency; 6. UBC fail-counters absolute values.
e	Toggle display of Container IP addresses/hostnames.
`i`	Toggle display of idle Containers.
`l`	Toggle display of load average.
`p`	Toggle display of processes statistics.
`c`	Toggle display of CPU usage statistics.
`w`	Toggle display of swap information.
`m, M`	Toggle/expand display of memory information. Each subsystem, including memory, network and disk has a number of verbosity levels. In the minimal level no information is displayed. Corresponding interactive lowercase key decreases verbosity level, the same key in uppercase increases it.
`n, N`	Toggle/expand display of network statistics.
`d, D`	Toggle/expand display of disk usage and activity information.

`o`	Sort key. One of the sort option keys should follow it:

`n`	Sort by Container ID
`c`	Sort by CPU usage
`f`	Sort by UBC failure counters
`r`	Sort by the number of running processes
`p`	Sort by the total number of processes
`s`	Sort by Container status. Containers which probably are unusable or unstable (increasing UBC failure counters or very high scheduling latency) will be shown first.

vzpid

This utility prints the ID of the Container where the process is running. It has the following syntax:

```
vzpid <pid> [<pid>…]
```

Multiple process IDs can be specified as arguments.

vzsplit

This utility is used to generate a sample Container configuration file with a set of system resource control parameters. The syntax of this command is as follows:

```
vzsplit [-n num] [-f sample_name] [-s swap_size]
```

This utility is used for dividing the server into equal parts. It generates a full set of Containers system resource control parameters based on the total physical memory of the server it runs on and the number of Containers the server shall be able to run even if the given number of Containers consume all allowed resources.

Without any option the utility prompts for the desired number of Containers and outputs the resulting resource control parameters to the screen.

The utility accepts the following options:

`-n` *`num`*	Desired number of Containers to be simultaneously run on the server.
`-f` *`sample_name`*	Name of the sample configuration to create.
`-s` *`swap_size`*	Size of the swap file on the server. It is recommended to specify the swap size to be taken into account when the utility generates sample configurations.

The resulting sample configuration will be created in the `/etc/vz/conf` directory. The file name will be `ve-<`*`sample_name`*`>.conf-sample`. Now you can pass `<`*`sample_name`*`>` as an argument to the `--config` option of the `pctl create` command. If a sample with this name already exists, the utility will output an error message and will not overwrite the existing configuration.

Note: If you create a Container configuration sample by splitting Parallels Server Bare Metal resources via Parallels Server Bare Metal tools (e.g. Parallels Management Console) rather than using `vzsplit`, this sample is put to the `/var/vzagent/etc/samples` directory after its creation. This sample can then be used for creating new Containers by Parallels Server Bare Metal tools only.

vzcfgscale

This utility is used to “scale” Container configuration. It multiplies Container resource control parameters by the number passed as an argument. The syntax of this utility is as follows:

```
vzcfgscale [options] <CT_config_file>
```

Container configuration file shall be always the last parameter and the utility uses it to produce scaled Container configuration. The utility accepts the following options:

`-o <file>`	Output scaled configuration into the `<file>`. By default, utility prints its output to screen. Note that the file specified cannot be the same as `<CT_config_file>`, otherwise you will loose the configuration file content.
`-a <factor>`	Multiply all Container parameters by a `<factor>`.
`-c <factor>`	Multiply CPU parameters by a `<factor>`.
`-d <factor>`	Multiply disk related parameters by a `<factor>`.
`-u <factor>`	Multiply system resource control parameters by a `<factor>`.
`-n <factor>`	Multiply bandwidth parameters by a `<factor>`.
`-r`	Do not output Container-specific parameters like `VE_ROOT`, `VE_PRIVATE`, and so on. This option is useful for producing configuration samples to be used as an argument for the `--config` option of the `pctl create` command.

At least one multiplying argument is required. It is possible to specify more than one multiplying argument to use different factors for different group of parameters. If both `–a` and a specific group option is used, then the specific option factor takes precedence of the value specified by the `–a` option.

vzcfgvalidate

This utility is used to check resource management parameters consistency in the Container configuration file. It has the following syntax:

```
vzcfgvalidate <CT_config_file>
```

The utility has a number of constraints according to which it tests the configuration file. If a constraint is not satisfied utility prints a message with its severity status. Three severity statuses are thus defined in Parallels Server Bare Metal:

Recommendation	This is a suggestion, which is not critical for Container or server operations. The configuration is valid in general; however, if the system has enough memory, it is better to increase the settings as advised.
Warning	A constraint is not satisfied and the configuration is invalid. Applications in a Container with such invalid configuration may have suboptimal performance or fail in a not graceful way.
Error	An important constraint is not satisfied and the configuration is invalid. Applications in a Container with such invalid configuration have increased chances to fail unexpectedly, to be terminated or to hang.

It is suggested to use this utility when applications in a Containers behave in unexpected way and there seem to be no resource shortage for the Container.

vzhwcalc

`vzhwcalc` is used to scan information on the resources consumption on a server (this can be a physical server or a server) and create a special file on its basis. When launched without any options, it makes a snapshot of the resources consumption and writes down this information to a special file - a server configuration file. The collected information can then be used to create a Container on its basis where the physical server will be migrated. You may also use `vzhwcalc` to collect data on your server resources in one place and be aware of their current consumption.

The `vzhwcalc` utility has the following syntax:

```
vzhwcalc [options]
```

The following options can be passed to the `vzhwcalc` utility:

Option Name	Description
`-o, --out`	The name of the configuration file that will be created by the utility and contain information on the server main resources.
`-t, --scan-time`	The time during which the utility is to be run on the server. The time should be given in the `dhms` format (e.g. `1d2h30m40s`).
`-p, --scan-period`	The interval with which the server will be scanned by the utility.
`--mem-scale`	The enlargement factor by which the calculated memory on the server will be increased in the configuration file.
`--disk-scale`	The enlargement factor by which the calculated disk space on the server will be increased in the configuration file.
`-d, --dist-detect`	The path to the file on the server where the `distdetect-common.sh` script is located. You can specify several scripts and separate them by commas.
`-h, --help`	Print usage information.
`-v, --version`	Print the version of the utility.

The configuration file created by the `vzhwcalc` utility is placed to the same directory on the server from where you have run this utility and has the default name of `ve.conf` (i.e. in case the `-o` option was omitted during the utility execution).

vzmtemplate

The `vzmtemplate` utility is used to migrate the installed OS and application standard templates and OS EZ templates from the Source Server to the Destination Server. It has the following syntax:

```
vzmtemplate [-bhz] [--ssh=<ssh_options>]
           <[user_name@]Destination_server_IP_Address> template_name ...
```

The following options can be used with `vzmtemplate`:

Option	Description
`-z, --eztempl`	Migrates the specified OS EZ template(s) installed on the Source Server. Without this option specified, `vzmtemplate` moves standard OS and application templates.
`--ssh=`*`ssh_options`*	Additional `ssh` options to be used while connecting to the Destination Server.
`-b, --batch`	This option should be passed to `vzmtemplate` in scripts if you are going to use these scripts for running the `vzmtemplate` utility and do not wish them to analyze the `vzmtemplate` command output.
`-h, --help`	Displays the utility usage and exits.

To migrate a template, you should execute the `vzmtemplate` command on the Source Server and pass the corresponding options to it. During its execution, the utility will try to connect to the Destination Server with the IP address of *`Destination_server_IP_Address`* and move the specified template(s) to this server. By default, `vzmtemplate` logs in to the Destination Server as `root` and asks you for the password of this user. However, you can make the utility use other credentials to log in to the Destination Server by appending the corresponding user name with the `@` symbol to the Server IP address (e.g. `user1@192.168.0.123`). Keep in mind that the specified user should have the `root` privileges; otherwise, the command will fail.

CHAPTER 4

Managing Virtual Machines

This chapter describes the utilities that can be used for managing your virtual machines.

In This Chapter

pctl

Parallels virtual machines can be managed using the `pctl` command-line utility. The utility is installed on the Parallels server during the product installation.

General Syntax

The `pctl` utility is used to perform administration tasks on virtual machines. The utility supports a full range of tasks from creating and administering virtual machines to installing Parallels Tools, getting statistics, and generating problem reports.

Syntax

```
pctl command ID|name [options] [-v, --verbose number]
```

Parameters

Name	Description
`command`	The name of the command to execute (see the table below for the complete list of commands).
`ID`	The ID of the virtual machine on which to perform the operation. To obtain the list of the available virtual machines, use the `pctl list` command (p. 180).
`name`	The name of the virtual machine on which to perform the operation. To obtain the list of the available virtual machines, use the `pctl list` command (p. 180).
`options`	Command options. See individual commands for available options.
`-v, --verbose` *`number`*	Show verbose output. The greater the *number*, the more verbose output will be produced.

Remarks

To display help, enter `pctl` without any parameters.

Links

Legend (p. 11)

pctl create

Creates a new virtual machine. A virtual machine can be created from scratch or from a virtual machine template. When created from scratch, the target operating system type or version must be specified. To create a virtual machine from a template, the template name must be passed to the command.

Syntax

```
pctl create name {--ostype name|--distribution name} [--location path]
pctl create name --ostemplate name [--location path]
```

Parameters

Name	Description
name	User-defined new virtual machine name. If the name consists of two or more words separated by spaces, it must be enclosed in quotes.
`-o, --ostype` *name*	The name of the family of the operating system that will be installed in the virtual machine. Select from one of the following: ▪ `windows` ▪ `linux` ▪ `feebsd` ▪ `other` (specify this option if the operating system you are planning to install is not listed above).
`-d, --distribution` *name*	The operating system version that will be installed in the virtual machine. For the full list of OS versions, refer to the `pctl` man pages.
`--ostemplate` *name*	The name of the virtual machine template from which to create the new virtual machine. Use the `pctl list --template` command to obtain the list of the available templates.
`--location` *path*	Name and path of the directory where to store the new virtual machine files. If this parameter is omitted, the files will be crated in the default virtual machine directory.

Remarks

When creating a virtual machine from scratch, you may specify the operating system family or version. If an operating system version is specified using the `--distribution` parameter, the virtual machine will be configured for that operating system. If an operating system family is specified using the `--ostype` parameter, the virtual machine will be configured for the default version of this OS family. The default versions are determined internally by Parallels and are kept in sync with other Parallels management tools such as Parallels Management Console. The best way to find out the default versions used in your Parallels installation is by creating a sample virtual machine.

Links

General Syntax (p. 59), Legend (p. 11)

pctl start, stop, reset

Start, stop, and reset a virtual machine.

Syntax

```
pctl start ID|name
pctl stop ID|name [--kill]
pctl reset ID|name
```

Parameters

Name	Description
ID	The ID of the virtual machine to start, stop, or reset.
name	The name of the virtual machine to start, stop, or reset.
`--kill`	Perform a 'hard' virtual machine shutdown. If this option is omitted, an attempt to perform a graceful shutdown will be made.

Remarks

The `stop` command can perform a 'hard' or a graceful virtual machine shutdown. If the `--kill` parameter is included, the 'hard' shutdown will be performed. If the parameter is omitted, the outcome of the graceful shutdown attempt will depend on the following:

- If the Parallels Tools package is installed in a virtual machine, the graceful shutdown will be performed using its facilities.
- If the Parallels Tools package is not installed, the command will try to perform a graceful shutdown using ACPI. Depending on the ACPI support availability in the guest operating system, this may work or not.

The `reset` command stops and then starts a virtual machine. The command first performs a 'hard' virtual machine shutdown and then starts the virtual machine from the stopped state.

The `start` command can be used to start a stopped virtual machine or to resume a paused virtual machine (p. 181).

Links

General Syntax (p. 59), Legend (p. 11)

pctl delete

Deletes a virtual machine from the Parallels server. The command removes a virtual machine from the Parallels Server Bare Metal registry and permanently deletes all its files from the server. Once completed, this operation cannot be reversed.

Syntax

```
pctl delete ID|name
```

Parameters

Name	Description
ID	The ID of the virtual machine to delete.
name	The name of the virtual machine to delete.

Links

General Syntax (p. 59), Legend (p. 11)

pctl clone

Creates an exact copy of the specified virtual machine.

Syntax

```
pctl clone ID|name --name new_name [--template] [--location path]
```

Parameters

Name	Description
ID	The ID of the virtual machine to clone
name	The name of the virtual machine to clone.
`--name` *new_name*	The name to be assigned to the new virtual machine.
`--template`	Create a virtual machine template instead of a real virtual machine. Templates are used as a basis for creating new virtual machines.
`--location` *path*	Name and path of the new virtual machine directory. If this parameter is omitted, the new virtual machine will be created in the default directory.

Links

General Syntax (p. 59), Legend (p. 11)

pctl list

Obtains a list of virtual machines on the Parallels server. The command allows you to obtain a summary list containing only the virtual machine ID, name, and status or to obtain a detailed information about a specific or all virtual machines.

Syntax

```
pctl list [--all] [--template] [--no-header]
            [-o, --output name[,name...]] [-s, --sort name|-name]

pctl list --info [ID|name]
```

Parameters

Name	Description
`-a, --all`	List all, running, stopped, suspended, and paused virtual machines. If this and the rest of the parameters are omitted, only the running virtual machines will be displayed.
`-t, --template`	List the available virtual machine templates. The real virtual machines will not be included in the output.
`--no-header`	Do not display column headers.
`ID`	The ID of the virtual machine for which to display the detailed information. If none specified, the information will be displayed for all registered virtual machines.
`-o, --output` *`name`*	Display one (or any combination) of the following fields: ▪ `uuid` -- Virtual machine ID. ▪ `name` -- Virtual machine name. ▪ `status` --Virtual machine status (running, stopped, etc.). The above fields can be combined in a single command using comma separator (e.g. `uuid, name`). The excluded fields will not be displayed. The field names must be typed in lower case.
`-s, --sort` *`name`*	Sort the virtual machine list by the specified parameter in ascending order.
`-i, --info`	Display detailed information about a virtual machine.
`ID`	The ID of the virtual machine for which to display the detailed information. If not specified, the information will be displayed for all registered virtual machines.
`name`	The name of the virtual machine for which to display the detailed information. If not specified, the information will be displayed for all registered virtual machines.

Links

General Syntax (p. 59), Legend (p. 11)

pctl pause, suspend, resume

Pause, suspend, and resume a virtual machine.

Syntax

```
pctl pause ID|name
pctl suspend ID|name
pctl resume ID|name
```

Parameters

Name	Description
ID	The ID of the virtual machine to pause, suspend, or resume.
name	The name of the virtual machine to pause, suspend, or resume.

Remarks

The `pause` command pauses a virtual machine. To continue the virtual machine operation, use the `pctl start` command (p. 178).

The `suspend` command suspends the virtual machine operation. When a running virtual machine is suspended, the state of the virtual machine processes is saved to a file on the host. After that, the machine is stopped. To resume the machine, use the `resume` command.

Links

General Syntax (p. 59), Legend (p. 11)

pctl register, unregister

The `register` command is used to register a virtual machine with Parallels Server Bare Metal.

The `unregister` command removes a virtual machine from the Parallels Server Bare Metal registry.

Syntax

```
pctl register path
pctl unregister ID|name
```

Parameters

Name	Description
path	An absolute path to the virtual machine directory.
ID|name	The ID or the name of the virtual machine to remove from the Parallels Server Bare Metal registry.

Remarks

Use the `register` command when you have a virtual machine on the server that does not show up in the list of the virtual machines registered with the Parallels Server Bare Metal. This can be a machine that was previously removed from the registry or a machine that was copied from another location.

The `unregister` command removes a virtual machine from the Parallels Server Bare Metal registry, but does not delete the virtual machine files from the server. You can re-register such a machine with Parallels Server Bare Metal later using the `register` command.

Links

General Syntax (p. 59), Legend (p. 11)

pctl installtools

Installs Parallels Tools in the specified virtual machine.

Syntax

```
pctl installtools ID|name
```

Parameters

Name	Description
ID	The ID of the target virtual machine.
name	The name of the target virtual machine.

Notes

To use this command, the target virtual machine must be running.

Links

General Syntax (p. 59), Legend (p. 11)

pctl snapshot

Takes a snapshot of a running virtual machine.

Syntax

```
pctl snapshot ID|name [-n,--name name] [-d,--description desc]
```

Parameters

Name	Description
ID	The virtual machine ID.
name	The virtual machine name.
-n, --name *name*	User-defined snapshot name.
-d, --description desc	User-defined snapshot description.

Links

General Syntax (p. 59), Legend (p. 11)

pctl snapshot-delete

Deletes a virtual machine snapshot.

Syntax

```
pctl snapshot-delete ID|name -i,--id snapshot_id
```

Parameters

Name	Description
ID	The virtual machine ID.
name	The virtual machine name.
`-i, --id` *snapshot_id*	The ID of the snapshot to delete.

Notes

If the specified snapshot has child snapshots that were derived from it, they will NOT be deleted.

Links

General Syntax (p. 59), Legend (p. 11)

pctl snapshot-list

Displays a list of snapshots of the specified virtual machine.

Syntax

```
pctl snapshot-list ID|name [-t,--tree] [-i,--id snapshot_id]
```

Parameters

Name	Description
ID	The virtual machine ID.
name	The virtual machine name.
`-t, --tree`	Displays the snapshot list as a tree. The default display format is tabular with Parent Snapshot ID and Snapshot ID as columns.
`-i, --id` *snapshot_id*	The ID of the snapshot to use as a root. If this parameter is omitted, the entire snapshot tree will be displayed.

Links

General Syntax (p. 59), Legend (p. 11)

pctl snapshot-switch

Reverts the specified virtual machine to the specified snapshot.

Syntax

```
pctl snapshot-switch ID|name -i,--id snapshot_id
```

Parameters

Name	Description
ID	The virtual machine ID.
name	The virtual machine name.
-i, --id *snapshot_id*	The ID of the snapshot to revert to.

Links

General Syntax (p. 59), Legend (p. 11)

pctl migrate

Migrates a virtual machine from one host to another.

Syntax

```
pctl migrate <[source_server/]ID> <destination_server[/ID]> [--dst <path>]
```

Options:

Name	Description
ID	The source virtual machine ID or name.
source_server	The IP address or hostname of the Source Server where the virtual machine is hosted before migration.
destination_server	The IP address or hostname of the Destination Server where the virtual machine is migrated from the Source Server.
--dst *path*	Name and path of the directory on the Destination Server where the virtual machine files should be stored.

Links

General Syntax (p. 59), Legend (p. 11)

pctl capture

Captures the screen of a virtual machine desktop and saves it to a file on the client machine. The data is saved in the Portable Network Graphics (PNG) format.

Syntax

```
pctl capture ID|name --file name
```

Parameters

Name	Description
ID	The virtual machine ID.
name	The virtual machine name.
`--file` *name*	Name and path of the file to which the image should be saved. You should include the file extension (.png) or the file will be saved without one.

Links

General Syntax (p. 59), Legend (p. 11)

pctl backup

Backs up a virtual machine.

Syntax

```
pctl backup vm_id|vm_name
                [-s,--storage user[[:passwd]@server[:port]]
                [--description desc]
```

Parameters

Name	Description
vm_id | *vm_name*	The UUID or the name of the virtual machine to back up.
`-s,--storage`	This option is used to specify the backup server connection and login parameters. If this option is omitted, the backup will be saved on the default backup server. The default backup server can be configured using the `prlsrvctl set` command.
user	The name of the user on a remote backup server.
passwd	The user password. If omitted, the user will be prompted to enter a password.
server	Server hostname or IP address.
port	Port number. If omitted, the default port number will be used.
`--description` *desc*	Backup description.
`-i`	Create a full backup of the virtual machine. A full backup contains all virtual machine data.
`-f`	Create an incremental backup of the virtual machine. An incremental backup contains only the files changed since the previous full or incremental backup. This is the default backup type.

Links

General Syntax (p. 59), Legend (p. 11)

pctl backup-list

Lists the available backups for the specified virtual machine.

Syntax

```
pctl backup-list [vm_id|vm_name] [-f,--full]
                        [-s,--storage user[[:passwd]@server[:port]]
```

Parameters

Name	Description
vm_id | *vm_name*	The UUID or the name of the virtual machine for which to list the available backups.
`-f, --full`	Display full backup information.
`-s,--storage`	Backup server connection and login parameters. If this option is omitted, the backups will be searched for on the default backup server. The default backup server can be configured using the `prlsrvctl set` command.
`--localvms`	List only the backups of the virtual machines that were residing on the local server .
user	The name of the backup server user.
passwd	The user password.
server	Backup server hostname or IP address.
port	Port number. If omitted, the default port is used.

Links

General Syntax (p. 59), Legend (p. 11)

pctl backup-delete

Deletes a virtual machine backup.

Syntax

```
pctl backup-delete {{vm_id|vm_name} | -t,--tag backup_id}
                       [-s,--storage user[[:passwd]@server[:port]]
```

Parameters

Name	Description
vm_id | *vm_name*	The UUID or the name of the virtual machine. If this option is specified, the command will delete the latest virtual machine backup. To delete a specific backup, omit this option and specify the backup ID using the `--tag` option (described below).
`-t,--tag` *backup_id*	The ID of the backup to delete.
`-s,--storage`	The backup server connection and login parameters. If this option is omitted, the backups will be searched for on the default backup server. The default backup server can be configured using the `prlsrvctl set` command.
user	The name of the backup server user.
passwd	The user password.
server	Backup server hostname or IP address.
port	Port number. If this option is omitted, the default port will be used.

Links

General Syntax (p. 59), Legend (p. 11)

pctl restore

Restores a virtual machine from a backup.

Syntax

```
pctl restore {{vm_id|vm_name} | -t,--tag backup_id}
                 [-s,--storage user[[:passwd]@server[:port]]
```

Parameters

Name	Description
vm_id | *vm_name*	The UUID or the name of the virtual machine. If this option is specified, the command will restore it from the latest available backup. To restore a virtual machine from a specific backup, omit this option and specify the backup ID using the `--tag` option (described below).
`-t,--tag` *backup_id*	The backup ID from which to restore a virtual machine.
`-s,--storage`	The backup server connection and login parameters. If this option is omitted, the backups will be searched for on the default backup server. The default backup server can be configured using the `prlsrvctl set` command.
user	The name of the backup server user.
passwd	The user password.
server	Backup server hostname or IP address.
port	Port number. If omitted, the default port will be used.

Links

General Syntax (p. 59), Legend (p. 11)

pctl exec

Executes a command inside a virtual machine. Parallels Tools must be installed in a virtual machine to use this utility. Commands in Linux guests are invoked with `bash -c`.

Syntax

```
pctl exec vm_id|vm_name command
```

Parameters

Name	Description
vm_id | *vm_name*	The UUID or the name of the virtual machine.
command	A command to execute.

Links

General Syntax (p. 59), Legend (p. 11)

pctl enter

Creates a command prompt channel to a virtual machine. By using this command, you can create a command prompt channel and execute commands in a virtual machine. Parallels Tools must be installed in a virtual machine to use this utility.

Syntax

```
pctl enter exec vm_id|vm_name
```

Parameters

Name	Description
vm_id | *vm_name*	The UUID or the name of the virtual machine.

Links

General Syntax (p. 59), Legend (p. 11)

pctl problem-report

Obtains a problem report for the specified virtual machine and displays it on the screen.

Syntax

```
pctl problem-report ID|name
```

Parameters

Name	Description
ID	The ID of the virtual machine for which to obtain the problem report.
name	The name of the virtual machine for which to obtain the report. If the name consists of separate words, it must be enclosed in quotes.

Remarks

The command collects technical data about a virtual machine and displays the report on the screen (the output can also be piped to a file). The report can then be directed to Parallels technical support for analysis.

Links

General Syntax (p. 59), Legend (p. 11)

pctl server

Obtains the information about the Parallels server and Parallels Server Bare Metal. It also allows you to disable the Parallels Server Bare Metal component responsible for managing virtual machines.

Syntax

```
pctl server shutdown|info
```

Parameters

Name	Description
`info`	Displays the Parallels Server Bare Metal information.
`shutdown`	Disables the Parallels Server Bare Metal component responsible for managing virtual machines. If one or more virtual machines are running, clients are connected, or some tasks are currently in progress, then the operation will be aborted.

See Also

`prlsrvctl info` (p. 60)

`prlsrvctl shutdown` (p. 65)

Links

General Syntax (p. 59), Legend (p. 11)

pctl set

The pctl `set` command is used to modify the configuration of a virtual machine and manage virtual machine devices and shared folders. The following subsections provide technical information on how to use the command to perform these tasks.

Modifying Virtual Machine Configuration

The `pctl set` command can be used to modify some of the virtual machine configuration parameters, including virtual CPU availability, RAM and video memory size, startup and shutdown options, and some others.

Syntax

```
pctl set ID|name [--cpus number] [--memsize number]
           [--videosize number] [--description description]
           [--autostart on|off|auto] [--autostart-delay number]
           [--autostop stop|suspend]
           [--start-as-user administrator|owner|user:passwd]
```

Parameters

Name	Description
ID	Target virtual machine ID.
name	Target virtual machine name.
`--cpus` *number*	Number of virtual CPUs in the virtual machine. If the host has more than one CPU, this option allows you to set the number of virtual CPUs to be available in the virtual machine.
`--memsize` *number*	The amount of memory (RAM) available to the virtual machine, in megabytes.
`--videosize` *number*	The amount of video memory available to the virtual machine graphics card.
`--description` *VM_description*	Short description of the virtual machine.
`--autostart on|off|auto`	Defines the virtual machine start-up options: ▪ `on` -- the virtual machine is started automatically wen the Parallels server starts or the Parallels Server Bare Metal component responsible for managing virtual machines is disabled. ▪ `off` -- the autostart is off. This is the default virtual machine start-up mode. ▪ `auto` -- resume the virtual machine state prior to shutting down the Parallels server or disabling the Parallels Server Bare Metal component responsible for managing virtual machines. If you set this option to `on` or `auto`, you must additionally specify the `--start-as-user` option (see below).
`--autostart-delay` *number*	Sets the time delay used during the virtual machine automatic startup.

`--autostop stop|suspend`	Sets the automatic shutdown mode for the specified virtual machine: ▪ `stop` -- the virtual machine is stopped when you shut down the Parallels server or disable the Parallels component responsible for managing virtual machines. ▪ `suspend` -- the virtual machine is suspended when the Parallels server is shut down or the Parallels component responsible for managing virtual machines is disabled.
`--start-as-user administrator|owner|`*`user`*`:`*`passwd`*	Specifies the account to use to autostart the virtual machine: ▪ `administrator` -- start the virtual machine as the administrator of the host operating system. ▪ `owner` -- start the virtual machine as the virtual machine owner. ▪ *`user:passwd`* -- start the virtual machine as the specified user.

Links

General Syntax (p. 59), Legend (p. 11)

Managing Virtual Devices

The `pctl set` command allows to add virtual devices to a virtual machine and to modify and delete existing virtual devices.

General Syntax

```
pctl set ID|VM_name --device-add dev_type options
pctl set ID|VM_name --device-set name options
pctl set ID|VM_name --device-del name
```

Parameters

Name	Description
ID	The virtual machine ID.
VM_name	The virtual machine name.
`--device-add` *dev_type options*	Adds a virtual device to the specified virtual machine. The *dev_type* parameter specifies the virtual device type (`hdd`, `cdrom`, `fdd`, `net`, etc.). The *options* parameters specifies device-type specific options.
`--device-set` *name options*	Modifies the configuration of an existing virtual device in the specified virtual machine. The *name* parameter specifies the virtual device name. The *options* parameters specifies device-type specific options.
`--device-del` *name*	Deletes a virtual device from the virtual machine. The *name* parameter specifies the name of the virtual device to delete.

Remarks

All device-related parameters can be subdivided into the following categories:

- Hard disk drives (p. 198)
- Optical disk drives (p. 200)
- Network cards (p. 203)
- Floppy disk drives (p. 202)
- USB devices (p. 207)
- Serial ports (p. 205)
- Parallel ports (p. 206)
- Sound cards (p. 208)

Each group of parameters is explained in the following subsections in detail.

Notes

All operations on virtual machine devices (adding, modifying, or removing a device) must be performed on a stopped virtual machine. An attempt to perform any of these operations on a running virtual machine will result in error.

Links

Legend (p. 11)

Hard Disk Drive Management Parameters

This group of parameters is used to add and configure virtual hard disks in a virtual machine.

Syntax

```
pctl set ID|VM_name --device-add hdd [--image name]
          [--type expand|plain][--size number][--split]
          [--iface ide|scsi][--position number]
          [--enable|--disable]

pctl set ID|VM_name --device-add hdd --device name
          [--iface ide|scsi][--position number]
          [--enable|--disable]

pctl set ID|VM_name --device-set hddN [--image name]
          [--type expand|plain][--size number][--split]
          [--iface ide|scsi][--position number]
          [--enable|--disable]

pctl set ID|VM_name --device-set hddN --device name
          [--iface ide|scsi][--position number]
          [--enable|--disable]
```

Parameters

Name	Description
`ID`	The virtual machine ID.
`VM_name`	The virtual machine name.
`--device-add`	Adds a virtual hard disk drive to the virtual machine. You can connect up to four IDE devices and up to seven SCSI devices to a virtual machine. This includes hard disks and optical disk drives.
`--device-set`	Modifies the parameters of an existing virtual hard disk.
`hdd`	Specifies the type of the virtual device to add to the virtual machine (in this instance, a virtual hard disk).
`hddN`	The name of the virtual hard disk to modify. Virtual hard disks are named using the `hddN` format where *N* is the drive index number starting from 0 (e.g. `hdd0`, `hdd1`). To obtain the list of disk names, use the `pctl list` command with the `--info` option.
`--image name`	This options is used to create a virtual hard disk using an image file. You have an option of creating a new image file or to use an existing image. ▪ To use an existing image file, specify its name and path using the *name* parameter. ▪ To create a new image file, omit the `--image` parameter. New image files are created in the virtual machine directory and are automatically named using the `harddiskN.hdd` format, where *N* is the disk index number (e.g. `harddisk0.hdd`, `harddisk1.hdd`).
`--device name`	This option is used to create a virtual hard disk based on a boot camp partition (Mac hosts). The *name* parameter must contain the boot

	camp partition name.
`--type expand|plain`	For image file based virtual disk drives, specified the disk type: ▪ `expand` -- expanding disk. The image file is small initially and grows in size as you add data to it. This is the default virtual disk type. ▪ `plain` -- plain disk. The image file has a fixed size from the moment it is created (i.e the space is allocated for the drive fully). Plain disks perform faster than expanding disks.
`--size` *`number`*	The size of the virtual hard disk, in megabytes. The default size is 32,000 MB.
`--split`	Splits the hard disk image file into 2 GB pieces. You should split a virtual disk if it is stored on a file system that cannot support files larger than 2 GB (e.g. FAT16).
`--iface ide|scsi`	Interface type: ▪ `ide` -- IDE drive. ▪ `scsi` - SCSI drive (default).
`--position` *`number`*	The SCSI or IDE device identifier to be used for the virtual disk. The allowed ID ranges are the following: ▪ for IDE devices: `0:0`, `0:1`, `1:0`, `1:1`; ▪ for SCSI device: `0:0`, `1:0`, `2:0`, `3:0`, `4:0`, `5:0`, `6:0`. You can use one of the following formats for specifying IDs: *`ID`*`:`*`bus`*, *`ID-bus`*, *`ID`*. For example, if you specify 3:0 (or 3-0 or 3) as *`number`* for a SCSI drive, the guest OS will see the drive as having ID 3 on SCSI bus 0.
`--enable`	Enables the specified virtual disk drive. All newly added disk drives are enabled by default (provided the `--disable` option is omitted).
`--disable`	Disables the specified virtual disk drive. The disk drive itself is not removed from the virtual machine configuration.

Links

General Syntax (p. 59), Virtual Device Management (p. 196), Legend (p. 11)

Optical Disk Drive Management Parameters

This group of parameters is used to add and configure virtual optical disk drives, such as DVD or CD drives.

Syntax

```
pctl set ID|VM_name --device-add cdrom --image image_name
          [--iface ide|scsi] [--position number]
          [--enable|--disable] [--connect|--disconnect]

pctl set ID|VM_name --device-add cdrom --device device_name
          [--iface ide|scsi] [--position number]
          [--enable|--disable] [--connect|--disconnect]

pctl set ID|VM_name --device-set cdromN
          {--device name|--image name} [--iface ide|scsi]
          [--position number][--enable|--disable]
          [--connect|--disconnect]
```

Parameters

Name	Description
ID	The virtual machine ID.
name	The virtual machine name.
`--device-add`	Adds a DVD/CD drive to the virtual machine. You can connect up to four IDE devices and up to seven SCSI devices to a virtual machine. This includes virtual hard disks and DVD/CD drives.
`--device-set`	Modifies the parameters of an existing virtual optical disk.
`cdrom`	Specifies the virtual device type (in this instance, a CD or DVD drive).
`cdrom`*N*	The name of the DVD/CD drive to modify. The *N* postfix indicates the drive index number. To obtain the list of the available drives, use the `pctl list` command with the `--info` option.
`--device` *name*	The name of the physical optical disk to connect to the virtual machine.
`--image` *name*	The name of an existing disk image file to mount in the virtual machine. Currently, the following image file formats are supported: `.iso`, `.cue`, `.ccd`, and `.dmg`. The image must not be compressed and/or encrypted.
`--iface ide|scsi`	Interface type: ▪ `ide` -- IDE disk. ▪ `scsi` -- SCSI disk (default).
`--position` *number*	The SCSI or IDE device identifier to be used for the DVD/CD drive. The allowed ID ranges are the following: ▪ for IDE devices: `0:0`, `0:1`, `1:0`, `1:1`; ▪ for SCSI device: `0:0`, `1:0`, `2:0`, `3:0`, `4:0`, `5:0`, `6:0`. You can use one of the following formats for specifying IDs: *ID*`:`*bus*, *ID-bus*, *ID*. For example, if you specify 3:0 (or 3-0 or 3) as *number* for a SCSI drive, the guest OS will see the drive as having ID 3 on SCSI bus 0.

`--enable`	Enables the specified DVD/CD drive. All newly added drives are enabled by default (provided the `--disable` option is omitted).
`--disable`	Disables the specified optical disk drive. The disk drive itself is not removed from the virtual machine configuration.
`--connect`	Automatically connect the specified optical disk drive during the virtual machine startup process.
`--disconnect`	Do not automatically connect the specified optical disk drive during the virtual machine startup process.

Links

General Syntax (p. 59), Virtual Device Management (p. 196), Legend (p. 11)

Floppy Disk Drive Management Parameters

This group of parameters is used to add floppy disk drives to a virtual machine and to modify existing virtual floppy disk drives.

Syntax

```
pctl set ID|VM_name --device-add fdd [--device name]
          [--enable|--disable][--connect|--disconnect]

pctl set ID|VM_name --device-set fdd [--device name]
          [--enable|--disable][--connect|--disconnect]
```

Parameters

Name	Description
ID	The virtual machine ID.
VM_name	The virtual machine name.
`fdd`	Specifies the type of the virtual device to add or modify (in this instance, a floppy disk drive).
`--device-add`	Adds a new floppy disk drive to the virtual machine. You can connect only one floppy disk drive to a virtual machine.
`--device-set`	Modifies the parameters of an existing virtual floppy disk drive.
`--device` *name*	The name of the physical floppy disk drive to connect to the virtual machine. If this parameter is omitted, a floppy drive image emulating the floppy disk drive will be created.
`--enable`	Enables the specified floppy disk drive. All newly added floppy drives are enabled by default (provided the `--disable` option was omitted during the drive creation).
`--disable`	Disables the specified floppy disk drive. The drive itself is not removed from the virtual machine configuration.
`--connect`	Connect the specified floppy disk drive automatically during the virtual machine startup process.
`--disconnect`	Use this option if you don't want the specified floppy disk drive automatically connected to the virtual machine on its start.
`--image` *path*	The name and path of an existing floppy disk image file (usually `floppy.fdd`) to mount in the virtual machine.

Links

General Syntax (p. 59), Virtual Device Management (p. 196), Legend (p. 11)

Network Adapter Management Parameters

This group of parameters is used to manage virtual network adapters in a virtual machine.

Syntax

```
pctl set ID|VM_name --device-add net --type shared|host|bridged
           [--mac addr][--enable|--disable][--connect|--disconnect]

pctl set ID|VM_name --device-add net --type bridged --iface name
           [--mac addr][--enable|--disable] [--connect|--disconnect]

pctl set ID|VM_name --device-set netN --type shared|host
           [--mac addr][--enable|--disable][--connect|--disconnect]

pctl set ID|VM_name --device-set netN --type bridged
           --iface name [--mac addr|auto][--enable|--disable]
           [--connect|--disconnect]
```

Parameters

Name	Description
ID	The virtual machine ID.
VM_name	The virtual machine name.
`--device-add`	Adds a new virtual network adapter to the virtual machine.
`--device-set`	Used to configure an existing virtual network adapter.
`net`	Specifies the virtual device type to add (in this instance, a virtual network adapter).
`net`*N*	The name of the virtual network adapter to modify. To obtain the list of the available adapters, use the `pctl list` command with the `--info` option.
`--type shared|host|bridged`	Sets the networking mode for the virtual network adapter: ▪ `shared` -- Shared networking. Select this option if you wish to enable Network Address Translation (NAT) for the adapter. The adapter will share the IP address with the Parallels server when communicating with external networks. ▪ `host` -- Host-only networking. Select this option if you wish the virtual machine to communicate only with the Parallels server and other virtual machines included in the same network. Access to external networks is not allowed. ▪ `bridged` -- Bridged networking. The adapter is bound to the specified physical network adapter. The virtual machine will appear as a standalone computer on the network.
`--iface` *name*	Used with the bridged networking mode (see above). Specifies the name of the physical network adapter to which the virtual adapter should be bound.
`--mac` *addr*	The MAC address to be assigned to the virtual network adapter. If this option is omitted, the MAC address will be generated automatically.
`--mac` *addr*`|auto`	Specifies the MAC address to assign to an existing network adapter. Specify a desired MAC address using the *addr* parameter value or use the `auto` option to re-generate the existing address automatically.

`--enable`	Enables the virtual network card. All newly created network adapters are enabled by default (provided the `--disable` option is omitted).
`--disable`	Disables virtual network adapter. The adapter itself is not removed from the virtual machine configuration. Please note that a disabled virtual network adapter can only be enabled in a stopped virtual machine.
`--connect`	Automatically connect the virtual network adapter during the virtual machine startup process.
`--disconnect`	Do not automatically connect the virtual network adapter during the virtual machine startup process.

Links

General Syntax (p. 59), Virtual Device Management (p. 196), Legend (p. 11)

Serial Port Management Parameters

This group of parameters is used to manage serial ports in a virtual machine.

Syntax

```
pctl set ID|VM_name --device-add serial
          {--device name|--output file|--socket name}
          [--enable|--disable][--connect|--disconnect]

pctl set ID|VM_name --device-set serialN
          {--device name|--output file|--socket name}
          [--enable|--disable][--connect|--disconnect]
```

Parameters

Name	Description
ID	The virtual machine ID.
VM_name	The virtual machine name.
`--device-add`	Adds a new serial port to the virtual machine. You can connect up to four serial ports to a virtual machine.
`--device-set`	Modifies the parameters of an existing serial port.
`serial`	Specifies the type of the virtual device to add (in this instance, a serial port).
`--device` *name*	The name of the physical serial port to which to connect the virtual machine.
`--output` *file*	The name and path of the output file to which to connect the virtual serial port.
`--socket` *name*	The name of the physical socket to which to connect the virtual serial port.
`--enable`	Enables the virtual serial port. All newly added serial ports are enabled by default (provided the `--disable` option is omitted).
`--disable`	Disables the virtual serial port.
`--connect`	Automatically connect the virtual serial port during the virtual machine startup process.
`--disconnect`	Do not automatically connect the virtual serial port during the virtual machine startup process.

Links

General Syntax (p. 59), Virtual Device Management (p. 196), Legend (p. 11)

Parallel Port Management Parameters

This group of parameters is used to manage parallel port in a virtual machine.

Syntax

```
pctl set ID|VM_name --device-add parallel
          {--device name|--output file_name}
          [--enable|--disable][--connect|--disconnect]

pctl set ID|VM_name --device-set parallelN
          {--device name|--output file_name}
          [--enable|--disable][--connect|--disconnect]
```

Parameters

Name	Description
ID	The virtual machine ID.
name	The virtual machine name.
`--device-add`	Adds a new parallel port to the virtual machine. You can connect up to three parallel ports to a virtual machine.
`--device-set`	Modifies the parameters of an existing virtual parallel port.
`parallel`	Specified the type of the virtual device to add (in this instance, a virtual parallel port).
`parallel`*N*	The name of the parallel port to modify. To obtain the list of ports, use the `pctl list` command with the `--info` option.
`--device` *name*	The name of the physical parallel port to which to connect the virtual parallel port.
`--output` *file_name*	The name of the output file to which to connect the virtual parallel port.
`--enable`	Enables the specified parallel port. All newly added parallel ports are enabled by default (provided the `--disable` option was omitted during the port creation).
`--disable`	Disable the specified virtual parallel port. The port itself is not removed from the virtual machine configuration.
`--connect`	Automatically connect the specified virtual parallel port during the virtual machine startup process.
`--disconnect`	Do not automatically connect the specified virtual parallel port during the virtual machine startup process.

Links

General Syntax (p. 59), Virtual Device Management (p. 196), Legend (p. 11)

USB Controller Management Parameters

This group of parameters is used to manage the USB controller in a virtual machine.

Syntax

```
pctl set ID|VM_name --device-add usb [--enable|--disable]
```

Parameters

Name	Description
ID	The virtual machine ID.
VM_name	The virtual machine name.
usb	The type of the virtual device to add to the virtual machine (in this instance, a USB device).
--enable	Enables the USB controller. This is the default option.
--disable	Disables the USB controller.

Links

General Syntax (p. 59), Virtual Device Management (p. 196), Legend (p. 11)

Sound Device Management Parameters

This group of parameters is used to manage sound devices in a virtual machine.

Syntax

```
pctl set ID|VM_name --device-add sound --output name
            [--enable|--disable][--connect|--disconnect]

pctl set ID|VM_name --device-set sound --output name
            [--enable|--disable][--connect|--disconnect]
```

Parameters

Name	Description
ID	The virtual machine ID.
VM_name	The virtual machine name.
`sound`	The type of the virtual device to add to the virtual machine (in this instance, a sound device).
`--output` *name*	The name of a physical output device to which to connect the virtual sound device.
`--input` *name*	The name of the physical input device to which to connect the virtual sound device.
`--enable`	Enables the specified sound device. All newly added sound devices are enabled by default (provided the `--disable` option is omitted).
`--disable`	Disables the specified virtual sound device.
`--connect`	Automatically connect the sound device during the virtual machine startup process.
`--disconnect`	Do not automatically connect the sound device during the virtual machine startup process.

Links

General Syntax (p. 59), Virtual Device Management (p. 196), Legend (p. 11)

Removing Devices from Virtual Machine

The `--device-del` option is used to remove virtual devices from a virtual machine.

Syntax

```
pctl set ID|name --device-del name
```

Parameters

Name	Description
`--device-del` *`name`*	The name of the virtual device to delete from the virtual machine. To obtain the list of virtual devices, use the pctl `list` command with the `--info` option.

Links

General Syntax (p. 59), Virtual Device Management (p. 196), Legend (p. 11)

Managing Shared Folders

The `pctl set` command can be used to add shared folders to a virtual machine and to modify and delete existing shared folders.

Syntax

```
pctl set ID|VM_name --sharedfolder-add name --path path
                                         [--mode ro|rw]
                                         [--description txt]
                                         [--enable|--disable]

pctl set ID|VM_name --sharedfolder-set name [--mode ro|rw]
                                         [--path path]
                                         [--description txt]
                                         [--enable|--disable]

pctl set ID|VM_name --sharedfolder on|off

pctl set ID|VM_name --sharedfolder-del name
```

Parameters

Name	Description
ID	The virtual machine ID.
VM_name	The virtual machine name.
`--sharedfolder-add`	Adds a shared folder to the virtual machine.
`--sharedfolder-set`	Modifies the settings of an existing shared folder.
`--sharedfolder on|off`	Turns folder sharing on or off.
`--sharedfolder-del`	Removes the shared folder specified by *name* from the shared folder list.
name	User-defined shared folder name.
`--path`	Name and path of a folder on the Parallels server to share with the specified virtual machine.
`--mode`	Sharing mode: ▪ `ro` -- read-only. ▪ `rw` -- read and write.
`--description`	User-defined shared folder description.
`--enable`	Enable the shared folder.
`--disable`	Disable the shared folder.

Links

General Syntax (p. 59), Legend (p. 11)

pmigrate

Migrating virtual machines and Containers is performed using the `pmigrate` utility. This utility has the following syntax:

```
pmigrate <source_server> <destination_server> [options]
```

<*source_server*> is the Source Server which can be either the server where the virtual machine and Container to be migrated is residing (if you are migrating a virtual machine and Container) or the physical computer to be migrated (if you are migrating a physical computer). <*destination_server*> is the Destination Server, i.e. the Parallels server where the virtual machine and Container or the physical server is to be migrated. If the Source and/or Destination Server is not specified, the operation is performed on the local server.

<*source_server*> and <*destination_server*> consists of two parts:

- <*type*> denotes the type of computer to migrate and can be one of the following:
  - `h` must be specified when migrating a physical computer.
  - `c` must be specified when migrating a Container.
  - `v` must be specified when migrating a virtual machine.
- <*address*> denotes the location of computer to migrate and can be one of the following:
  - The computer location if you are migrating a physical computer.
  - The computer location and the virtual machine name or Container ID if you are migrating a virtual machine or Container, respectively. The location must be separated from the virtual machine name/Container ID by the slash (`/`).

The location format is as follows:

```
[<user>[:<password>]@]<destination_server_IP_address_or_hostname>[:<destinatio
n_server_port>]
```

The options ([*options*]) you can use with `pmigrate` depend on whether you are migrating a virtual machine or a Container. This section describes the parameters that can be used for moving virtual machines between Parallels servers and migrating Containers and physical computers to virtual machines. For information on parameters relating to migrating Containers, refer to `pmigrate` (p. 116).

Common options:

`--dst=<path>`	Specifies the name and path of the directory on the Destination Server where the virtual machine files will be stored. If this option is omitted, the default directory is used.
`-h, --help`	Displays the information on the utility usage.

Options specific for migrating Containers to virtual machines:

`-s, --size=<size>`	Sets the limit of the resulting virtual machine disk capacity. If you omit this option or specify 0, no limit is set. In this case disk quotas set for virtual machines on the Parallels server will be used. The following size modifiers can be used: G - gigabytes, M - megabytes (by default), K - kilobytes.

`-r, --reg[=<y|n>]`	Specifies whether to register the resulting virtual machine on the Parallels server. By default, the virtual machine is registered.
`-d, --remove[=<y|n>]`	Removes the source Container after migration. By default, the Container is left intact.
`--key=<`*`auth_key`*`>`	Set the authentication key for passwordless access to the Parallels server. To set this key, use the `parallels-c2v-agent --regkey <`*`auth_key`*`>` command on the server where the Container is residing (`<`*`auth_key`*`>` can be any alphanumeric value).

Options specific for migrating physical computers to virtual machines:

`-r, --reg[=<y|n>]`	Specifies whether to register the resulting virtual machine on the Parallels server. By default, the virtual machine is registered.
`--osdata=<`*`path`*`>`	Sets the path to the directory with data files required for the OS reconfiguration and not found on the source computer.
`-a, --all`	Migrate all computer's disks and partitions with all available data. If this option is omitted, only data from the active disk drive is migrated.
`--key=<`*`auth_key`*`>`	Set the authentication key for passwordless access to the physical computer you want to migrate. To set this key on the computer, use the `parallels-transporter-agent --regkey <`*`value`*`>` command (`<`*`auth_key`*`>` can be any alphanumeric value).

pbackup

The `pbackup` utility is run on the so-called Backup Server. It connects via SSH to the servers where some or all virtual machines are to be backed up and puts the tarballs into the directory defined in the `/etc/vzbackup.conf` global backup configuration file (by default, this directory is `/vz/backup`). Later on, the virtual machine backups may be restored from this directory. It has the following syntax:

```
pbackup [backup_options] SERVER1 ... [CT options]
```

You may specify any number of servers names or IP addresses in the command-line. You may also enter these names as the value of the `BACKUP_NODES` parameter in the global backup configuration file to avoid the necessity to specify them in the command-line. In this case, you shall specify the `–a` option instead.

The backup options are the following:

`-a`	Back up all servers specified in the global backup configuration file.
`-c` *`CONFIG`*	Use an alternative backup configuration file.
`-n` *`CREDENTIALS`*	Set the Backup Server, i.e. the server that will be used for storing the resulting virtual machine backups. If this parameter is omitted, the specified virtual machines will be backed up to the server where they are hosted. The Backup Server can be specified in this format: *`user`*`[[:`*`password`*`]@`*`server_IP_address_or_hostname`*`[:`*`port`*`]`
`--ssh-opts` *`OPTIONS`*	Options to be passed to `ssh`. See examples in the global backup configuration file.
`-F, -I`	Crete a full backup. A full backup contain all virtual machine data.
`-i`	Make an incremental backup or, if no full backups are available, a full backup. An incremental backup contains only the file that were changed since the previous full or incremental backup.

The virtual machine options define the list of virtual machines to be backed up:

`-e` *`VM1`*`...`	The virtual machines to back up on the server. Virtual machines can be specified using both their IDs and their names.
`-x` *`VM1`*`...`	The virtual machines that need not be backed up (virtual machines to exclude). Virtual machines can be specified using both their IDs and their names.

prestore

The `prestore` utility is also run on the Backup Server. It uses the virtual machine backups stored on the Backup Server to restore them to their original servers (or to any other location if the `–d` option is specified). The syntax of the utility is the following:

```
prestore [restore_options] server1 ... [CT_options]
```

You can specify any number of servers (their names or IP addresses) whose virtual machines were at one time backed up and now need to be restored.

The restore options are the following:

`-l`	Do not restore any virtual machines. Show the information on the virtual machines available to be restored.
`-n` *`destination_server`*	The Backup Server where to look for existing backups. If this option is omitted, `prestore` searches for the backups on the server from which they were originally backed up. The Destination server can be specified in the following format: `<`*`user`*`[[:`*`password`*`]@`*`server_IP_address_or_hostname`*`[:`*`port`*`]>`
`-e` *`CT1`*`...`	The virtual machines to be restored on the server. Any virtual machine can be specified using both its ID or name.
`-x` *`CT1`*`...`	The virtual machines that need not be restored (virtual machines to exclude). Any virtual machine can be specified using both its ID and name.

prl_disk_tool

The `prl_disk_tool` utility is used to manage disk drives of your virtual machines. This utility has the following syntax:

```
prl_disk_tool <COMMAND> [OPTIONS] --hdd <disk_name>
```

Resizing Virtual Disks

When resizing virtual disks, the utility has the following syntax:

```
prl_disk_tool resize --size <size>[M|G] [--resize_partition] --hdd <disk_name>
[--force]
prl_disk_tool resize -i,--info [--units <K|M|G>] --hdd <disk_name>
```

Parameters

`resize`	Changes the capacity of the virtual disk. The disk to be resized must be formatted as NTFS, FAT 16, FAT 32, ext2, or ext3.
`--size`	The new size for the virtual disk. It can be set either in megabytes (specify `M` after the value) or in gigabytes (specify `G` after the value). By default, the size is set in megabytes.
`--resize_partition`	Resizes the last partition of the specified virtual disk.
`--hdd`	The full path to the virtual disk to be configured.
`--force`	Forces the resizing operation for suspended virtual disks.
`-i, --info`	Do not resize the virtual disk; just show the size the disk will have after resizing.
`--units`	Displays the disk size in kilobytes (`K`), megabytes (`M`, default), and gigabytes (`G`).

Compacting Virtual Disks

When compacting virtual disks, the utility has the following syntax:

```
prl_disk_tool compact [--buildmap] --hdd <disk_name> [--force]
prl_disk_tool compact -i,--info --hdd <disk_name>
```

Parameters

`compact`	Removes all empty blocks from the expanding virtual disk and reduces its size on your physical disk. The disk to be compacted must be formatted as NTFS, FAT 16, FAT 32, ext2, or ext3. You can also try to compact virtual disks with other filesystems using the `--buildmap` option.
`--hdd`	The full path to the virtual disk to be configured.
`--buildmap`	Used to compact virtual disks with unsupported filesystems.
`--force`	Forces the compacting operation for suspended virtual disks.
`-i, --info`	Do not compact the virtual disk; just display the information about the size the disk will have after compacting.

Preparing Virtual Disks for Booting

The `prl_disk_tool` utility can also be used to prepare virtual disks and real Boot Camp partitions for booting into virtual machines. In this case the utility has the following syntax:

```
prl_disk_tool configure --hdd <disk_name> [--para <paravirt_driver>] [--boot
<boot_driver>]
```

Parameters

`configure`	Prepares a disk or a real Boot Camp partition for booting into a virtual machine.
`--hdd`	The full path to the virtual disk to be configured.
`--para`	Specifies the full path to the Parallels virtualization driver.
`--boot`	Specifies the full path to the Parallels boot driver.

Displaying Utility Help

To display the utility help, use this command:

```
prl_disk_tool --help
```

vznetstat

This utility outputs traffic usage statistics for virtual machines and Containers. It has the following syntax:

```
vznetstat [-v <ID>] [-c <class>] [-a] [-r]
```

The utility displays input and output traffic for virtual machines and Containers for each defined network class. The network classes are described in the `/etc/vz/conf/networks_classes` file. If no options are specified the network statistics for all running virtual machines and Containers is printed.

The utility accepts the following options:

`-v <ID>`	Display statistics for virtual machines and Containers with the ID of `<ID>`. Multiple `–v` options can be given to a single `vznetstat` invocation.
`-c <class>`	Show the network statistics for the `<class>` class only.
`-a`	Display statistics for all classes.
`-r K|M|G`	Display the network statistics, which is shown in bytes by default, in the following measurement units: ▪ `K`: display the network statistics in kilobytes ▪ `M`: display the network statistics in megabytes ▪ `G`: display the network statistics in gigabytes
`--help`	Display the utility usage information.

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