Virtuozzo 6

Command Line Reference

December 01, 2016

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

Contents

CHAPTER 1

Introduction

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

This chapter provides general information about Virtuozzo and this guide.

- Virtuozzo basics (p. 8)
- goals and target audience of the guide (p. 9)
- guide organization (p. 9)
- resources to consult to get more information on Virtuozzo (p. 10)
- ways to submit feedback to the documentation team

In This Chapter

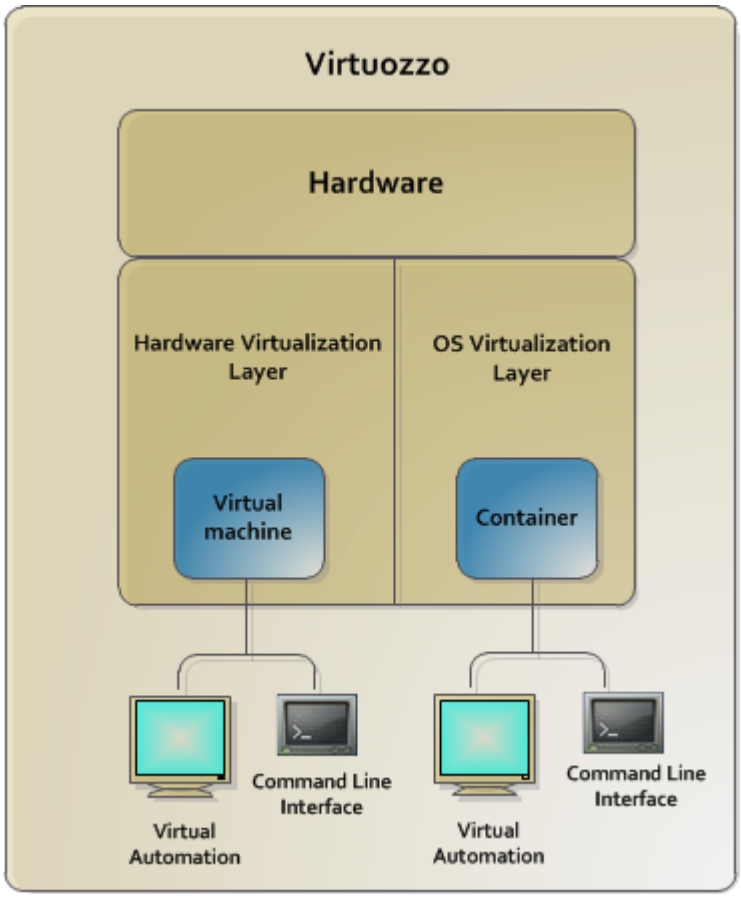
About Virtuozzo 6

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

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

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

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



About This Guide

This guide is a complete reference on all Virtuozzo configuration files and command-line utilities. It familiarizes you with the way to configure Virtuozzo 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

This guide is organized in the following way:

Chapter 1, Introduction (p. 8), gives an overview of Virtuozzo and this guide.

Chapter 2, Managing Virtuozzo 6 (p. 11), provides instructions on Virtuozzo configuration files, scripts, and command-line utilities.

Chapter 3, Managing Containers (p. 73), describes Virtuozzo command-line utilities that can be used for managing your Containers.

Chapter 4, Managing Virtual Machines (p. 146), focuses on Virtuozzo utilities that can used for managing your virtual machines.

Getting Help

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

- *Virtuozzo 6 Installation Guide*. This guide provides detailed information on installing Virtuozzo on your server, including the pre-requisites and the stages you shall pass.
- *Installing via PXE*. This guide provides information on installing Virtuozzo 6 over a network using a PXE (Preboot Execution Environment) server.
- *Getting Started With Virtuozzo 6***.** This guide provides basic information on how to install Virtuozzo on your server, create new Containers and virtual machines, and perform main operations on them. As distinct from the *Virtuozzo 6 Installation Guide*, it does not contain detailed description of all the operations needed to install and set Virtuozzo to work (e.g., installing Virtuozzo in the text mode).
- *Virtuozzo 6 User's Guide*. This guide provides comprehensive information on Virtuozzo covering the necessary theoretical conceptions as well as all practical aspects of working with the product. However, it does not deal with the process of installing and configuring your system.
- *Virtuozzo Storage Administrator's Guide*. This guide is intended for system administrators interested in deploying Virtuozzo Storage in their networks.
- *Virtuozzo 6 Templates Management Guide*. This guide is meant to provide complete information on templates, a Virtuozzo technology allowing you to efficiently deploy standard Linux applications inside your Containers and to greatly save the physical server resources (physical memory, disk space, etc.).

CHAPTER 2

Managing Virtuozzo 6

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

In This Chapter

Virtuozzo Configuration Files

The table below lists the configuration files available in Virtuozzo 6. Most files are located in the `/etc` directory on a host. Some configuration files are stored on other servers (e.g., on a Backup Node). If a configuration file is stored in a place other than the host, its exact location is specified.

Name	Description
/etc/vz/vz.conf	Global configuration file. This file keeps system-wide settings, such as the default location of templates and global network settings.
/etc/vz/conf/<*CT_ID*>**.conf**	Private configuration file of a Container with the ID <*CT_ID*>. This file keeps Container-specific settings: resource management parameters, the location of its private area, IP address, and so on.
/etc/vz/conf/ve-<*name*>**.conf-sample**	Sample files containing a number of default Container configurations. Some pre-created samples file are shipped with Virtuozzo (e.g., `basic` and `confixx`), but you can also create your own samples to meet your demands.
/etc/vz/conf/dists/<*distribution_name*>**.conf**	Linux distribution configuration files. These files define what scripts should be run when you perform specific operations with Containers (e.g., when you set a new IP address for a Container). The scripts differ from Virtuozzo action scripts and depend on the Linux version a particular Container is running.
/etc/vz/pfcache.conf	Configuration file used by the `pfcache` utility to manage memory and IOPS deduplication.
/etc/vz/oom-groups.conf	OOM killer configuration file with task badness adjustments.

/etc/vz/conf/networks_classes	Configuration file defining the network classes for traffic shaping and bandwidth management.
/etc/vzvpn/vzvpn.conf	Configuration file defining the parameters for establishing a private secure channel to the Virtuozzo support server.
/etc/sysctl.conf	Kernel parameters. Virtuozzo adjusts a number of kernel `sysctl` parameters and modifies the default `/etc/sysctl.conf` file.
/etc/vzredirect.d/*.conf	Configuration files defining offline management modes administrators use to access their Containers.
/etc/vzlmond.conf	Configuration file the `vzlmond` daemon uses to collect information on how much resources the server consumes at the moment.
/etc/vzstat.conf	Configuration file listing the warning and error levels for resource control parameters. If a parameter hits the warning or error value set for it, the `pstat` utility will display this parameter in yellow or red.
/etc/vzbackup.conf	Configuration file located on the Backup Node and determining the global backup settings for Containers. Note that the settings in this file do not apply to virtual machines.
/etc/vzpkgpoxy/vzpkgproxy.conf	Configuration file used by the `vzpkgproxy` utility to create 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	Configuration file used by the `vzpkg` utility to manage OS and application EZ templates.
/etc/vzstatrep.conf	Configuration file stored on the Monitor Node and used by the `vzstatrep` utility when generating statistic reports and graphics on the host resource consumption and sending these reports to the server administrator.
/etc/shaman/shaman.conf	Configuration file the `shaman` utility uses to manage and monitor High Availability clusters.

Global Virtuozzo Configuration File

Virtuozzo keeps its system wide configuration parameters in the `/etc/vz/vz.conf` configuration file. This file is in shell format. Keep in mind that Virtuozzo 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 quotas, 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 Virtuozzo.

Name	Description	Default value
VIRTUOZZO	This can be either “yes” or “no”. Virtuozzo System V startup script checks this parameter. If set to “no”, then Virtuozzo 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 Virtuozzo utilities communicating with the outer world through HTTP will use this server for managing all your HTTP messages.	`-`
ACTIONLOGDIR	This is the directory where `prlctl` keeps a log of its actions in the format suitable for Virtuozzo 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. Virtuozzo 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 `pmigrate` 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 `pmigrate` 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 percent, 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.	
VEFSTYPE	File system to use when caching OS templates: • **ext4**: Use to create Containers with the new layout. • **vz4**: Use to create legacy Containers.	`ext4`
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 in `/etc/fstab` file on the server, and having the `noauto` flag set for them in this file.	`/vz`
IPV6	Defines whether the IPv6 support is enabled on the host.	`yes`
GOLDEN_IMAGE	Enables (`yes`) or disables (`no`) embedding application templates into OS EZ template cache prior to creating Containers based on this cache.	`yes`
PFCACHE	Path to the memory and IOPS deduplication cache with common Container files.	`/vz/pfcache`
PFCACHE_IMAGE	Path to the private area of the memory and IOPS deduplication cache.	`/vz/pfcache.hdd`
PFCACHE_IMAGE_SIZE	Image size (in 1-KB blocks) of the memory and IOPS deduplication cache.	`10485760`
PFCACHE_INCLUDES	Directories for which memory and IOPS deduplication is enabled by default.	`bin lib lib64 opt sbin usr`
VZ_TOOLS_MEMLIMIT	Limits the total memory consumption of all `vz` utilities, e.g., `vzbackup`, `vzmigrate`, etc. Set in bytes.	

Logging parameters

Name	Description	Default value
LOGGING	This parameter defines whether `prlctl` should log its actions.	`yes`
LOGFILE	File where `prlctl` logs its actions.	`/var/log/vzctl.log`
LOG_LEVEL	There are three levels of logging defined in the current version of Virtuozzo.	`0`

Possible values of the LOG_LEVEL parameter

Log level	Information to be logged
0	Actions of `prlctl` on Containers like `start`, `stop`, `create`, `destroy`, `mount`, `umount`.
1	Level 1 logs events, calls to `prlctl` 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

Name	Description	Default value
DISK_QUOTA	Enables or disables disk quotas for Containers. If set to “no” then disk space accounting will be disabled.	`yes`

SLM parameters

Note: In Virtuozzo Containers 6, the SLM system was superseded by the new VSwap memory management scheme, and the SLM parameters are left for compatibility reasons only.

Name	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 Virtuozzo 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

Name	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 Kbps. If you want to shape traffic on more than one interface, set this parameter to “dev1:rate1 dev2:rate2”. For example, for two 100 Mbps 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 Kbps. Containers can consume bandwidth up to this limit in addition to the limit specified by the `RATE` parameter. Default value corresponds to 4 Mbps 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 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 Kbps. Note that 8 Kbps, 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`.	`eth0:1:8`
VZ_TOOLS_BCID	Enables limits for `pbackup`, `prestore`, and `pmigrate` operations. Make sure that the value of this parameter does not coincide with the ID assigned to some of your Containers. For example, if you leave the default value 2, ensure that no Container with ID 2 exists on the Node.	
VZ_TOOLS_IOLIMIT	Sets the disk I/O limit for the backup, restore, and migration operations, in bytes per second. Not set by default.	

Template parameters

Name	Description	Default value
TEMPLATE	This is the directory where to find templates. It is not recommended to redefine this option since all Virtuozzo templates use the default directory.	`/vz/template`

Container default parameters

Name	Description	Default value
VE_ROOT	Path to the Container root directory where the private area is mounted to.	`/vz/root/CT_ID`
VE_PRIVATE	Path to the Container private area. Must 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 `prlctl 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 `prlctl create`.	`centos-6-x86_64`
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

Name	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 running Containers) amount of allocated memory to the capacity of the system.	`60`

Supplementary parameters

Name	Description	Default value
VZWDOG	Defines whether the `vzwdog` module is loaded on Virtuozzo startup. This module is responsible for catching messages from the kernel. It is needed if you configure the serial Monitor Server for Virtuozzo.	`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 `prlctl 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 (p. 18)
- resource management parameters (p. 20)
- networking (p. 23)
- backup (p. 24)

Note: In Virtuozzo, 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

The table below list the miscellaneous parameters you can set in the configuration file of a Container:

Name	Description
VERSION	Specifies the Virtuozzo version the configuration file applies to. "2" relates to Virtuozzo version 4 and later.
ONBOOT	Specifies whether the Container should be started automatically on system startup. Virtuozzo 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 run from inside. If omitted or set to `yes`, restarting is allowed.
CAPABILITY	Specifies capabilities inside the Container. You can set the following capabilities: `AUDIT_WRITE`, `CHOWN`, `DAC_OVERRIDE`, `DAC_READ_SEARCH`, `FOWNER`, `FS_MASK`, `FSETID`, `IPC_LOCK`, `IPC_OWNER`, `KILL`, `LEASE`, `LINUX_IMMUTABLE`, `MKNOD`, `NET_ADMIN`, `NET_BIND_SERVICE`, `NET_BROADCAST`, `NET_RAW`, `SETGID`, `SETFCAP`, `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, VE_ADMIN.
OSTEMPLATE	The name of the OS template that was used for creating the Container. You do not have to change this parameter; `prlctl` will set it for you upon calling the `prlctl 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	In a configuration file of an existing Container, this parameter lists application templates installed with the `prlctl create`, `vzpkgadd`, or `vzpkg install` commands. In this case you should not modify it, because it is used by template management utilities to track installation history. This parameter is omitted if no templates have been installed 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 Virtuozzo 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 `prlctl 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 Virtuozzo utilities during their execution.

Resource Management Parameters

All resource management parameters can be subdivided into the CPU, disk, system, and VSwap categories for your convenience. Any parameter can be set with the `prlctl set` command and the corresponding option name (in the lower case, e.g., `--kmemsize` for `KMEMSIZE`, etc.). See **Managing Containers** (p. 73) 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. If the barrier and limit fields are in use, ranges for both thresholds are given.

CPU Parameters

Parameter	Description	Typical value
CPUUNITS	CPU weight. This is a positive integer number that defines how much CPU time the Container can get as compared to the other virtual machines and Containers running on the server. The larger the number, the more CPU time the Container can receive. Possible values range from 8 to 500000. If this parameter is not set, the default value of 1000 is used.	`250…1000`
CPULIMIT | **CPULIMIT_MHZ**	CPU limit, in per cent (`CPULIMIT`) or megahertz (`CPULIMIT_MHZ`), the Container is not allowed to exceed. The parameter is not set for newly created Containers; so they can consume all free CPU power of the server. When setting this parameter in per cent, keep in mind that one CPU core makes up 100%. So if the server has 4 CPU cores, the total CPU power will equal 400%.	
CPUS	Number of CPU cores defining the CPU limit for a Container. The limit is calculated by multiplying the power of one CPU core by the number of the specified CPU cores. This option also defines the number of CPUs shown to users from inside a Container. This parameter is not set for newly created Containers; so they can consume all free CPU power of the server.	
CPUMASK	The CPU affinity mask defining which CPUs on the Node can be used to handle the processes running in the Container. The CPU mask can be specified as both separate CPU index numbers (1,2,3) and CPU ranges (2-4,5-7).	
NODEMASK	The NUMA node mask defining a NUMA node to bind the Container to. Once you set the mask, the processes running in the Container will be executed only on the CPUs that belong to the specified NUMA node.	

BURST_CPU_AVG_USAGE	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 Virtuozzo configuration file.	`disabled`
BURST_CPULIMIT	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 Virtuozzo configuration file.	

Disk Parameters

Parameter	Description	Typical value
DISKSPACE	Total size of disk space that can be consumed by the Container, in 1 KB blocks.	`204800…10485760-204800…11534340`
QUOTAUGIDLIMIT	This parameter enables (if set to a value other than `0`) or disables (if set to `0`) per-user and per-group quotas for further management with the standard Linux `quota` utility. Enabling per-user and per-group quotas for a Container requires restarting the Container.	`0…N`
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`
IOPSLIMIT	The maximum number of disk input and output operations per second a Container is allowed to perform. By default, any newly created Container does not have the IOPS limit set and can perform so many disk I/O operations per second as necessary.	
IOLIMIT	The bandwidth a Container is allowed to use for its disk input and output (I/O) operations. By default, the limit is set in megabytes per second. However, you can use the following suffixes to use other measurement units: • `G` — sets the limit in gigabytes per second. • `K` — sets the limit in kilobytes per second. • `B` — sets the limit in bytes per second. In the current version of Virtuozzo, the maximum I/O bandwidth limit you can set for a Container is 2 GB per second. The default I/O bandwidth limit for all newly created Containers is set to 0, which means that no limits are applied to any Containers.	

System Parameters

Parameter	Description	Typical value
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.	`184320…3932160-196608…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`

VSwap Parameters

Parameter	Description	Typical value
PHYSPAGES	Amount of RAM that can be used by the processes of a Container, in 4-KB pages.	
SWAP	Amount of swap space that can be used by the Container for swapping out memory once the RAM is exceeded, in 4-KB pages.	
VM_OVERCOMMIT	Memory overcommit factor that defines the memory allocation limit for a Container. The limit is calculated as `(PHYSPAGES + SWAP) * factor`.	`Not limited`

Networking Parameters

Network-related parameters allow you to set bandwidth management parameters, hostname and IP addresses that a Container can use, and other parameters.

Name	Description
HOSTNAME	If this parameter is specified, then `prlctl` 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.
NETFILTER	Indicates which `iptables` modules are allowed for the Container. If some of the allowed modules are not loaded on the destination Hardware Node after migration or restoration from backup, they will be automatically loaded on the migrated or restored Container start. The following modes are available: • **disabled**: none. • **stateless**: (default) all modules except `conntrack` and NAT-related. • **stateful**: all modules except NAT-related. • **full**: all modules.
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 Kbps, 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 Parameters

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.

Name	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

Linux Distribution Configuration Files

Some Virtuozzo utilities (e.g., `prlctl`) 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, Virtuozzo 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 Virtuozzo by default (`centos.conf`, `fedora-core.conf`, `gentoo.conf`, etc.). To view all configuration files available on your Virtuozzo, 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., `centos-5.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 Virtuozzo, 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:

```
prlctl start <CT_ID>
prlctl set <CT_ID> --ipadd <IP_address>
prlctl 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:

```
prlctl set <CT_ID> --ipdel <IP_address>
```

```
prlctl 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:

```
prlctl 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:

```
prlctl 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:

```
prlctl 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 per-user/group quota. The script is launched inside the Container on executing the following command:

```
prlctl 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:

```
prlctl 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>
```

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 prlctl or vzp2v 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, prlctl 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.

Memory and IOPS Deduplication Configuration File

Contained in the `/etc/vz/pfcache.conf` file, memory and IOPS deduplication parameters allow you to tailor cache behavior and performance to your needs.

Parameters

Name	Description	Default Value
COUNT	The minimum number of file copies required for the file to become cacheable. Copies may exist in the same Container or different Containers.	`2`
MINSIZE	Minimal cacheable file size, bytes. Files smaller than this value will not be cached.	`0`
MAXSIZE	Maximal cacheable file size, bytes. Files larger than this value will not be cached.	`2147483648`
TIMEOUT	Time between caching attempts, seconds.	`5`
PFCACHE_IOLIMIT	Memory and IOPS deduplication cache IO bandwidth limit, bps. Unlimited by default.	
PFCACHE_IOPSLIMIT	Memory and IOPS deduplication cache IOPS limit. Unlimited by default.	
LOGLEVEL	Logging verbosity. Messages are logged in the system log file `/var/log/messages`.	`1`
PAGEMIN	The total number of memory pages used in Containers. • **0** — Cache even files without memory pages. • **1** — Cache only files in use. • **N** — Cache only when the total number of memory pages in Containers reaches N.	`1`
PURGEAHEAD	Extra cache space to free up in addition to the requested space. In per cent of the requested space. Used with the `pfcache purge --size` command.	`20%`

Network Classes Definition File

In Virtuozzo, 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 Virtuozzo 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. For example:

```
# HW node VPS's networks
0 10.10.10.0/24
0 10.10.15.0/24
# all IP("local" traffic)
1 0.0.0.0/0
# class 2 - "foreign" traffic
#2 10.0.0.0/8
#2 11.0.0.0/8
# inside "foreign" network there
# is a hole with "local" traffic
#1 10.10.16.0/24
```

vzvpn Configuration File

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

Format

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

Parameters

Name	Description
REMOTE_HOST	Mandatory. The hostname or the IP address of the Virtuozzo support server.
REMOTE_PORT	Mandatory. The port number of the Virtuozzo 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 Virtuozzo 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 Virtuozzo support server will be closed.
PING	Mandatory. The time, in seconds, at the end of which the port of the Virtuozzo 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 Virtuozzo 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 Virtuozzo 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 Virtuozzo 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 certain of your actions (for example, you have received the corresponding information from Virtuozzo).

Kernel Parameters

There is a number of kernel limits that should be set for the Virtuozzo software to work correctly. Virtuozzo 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 Virtuozzo:

```
# Controls IP packet forwarding
net.ipv4.ip_forward = 1
# Controls source route verification
net.ipv4.conf.default.rp_filter = 1
# Do not accept source routing
net.ipv4.conf.default.accept_source_route = 0
# Controls the System Request debugging functionality of the kernel
kernel.sysrq = 1
# Controls whether core dumps will append the PID to the core filename.
# Useful for debugging multi-threaded applications.
kernel.core_uses_pid = 1
# Controls the use of TCP syncookies
net.ipv4.tcp_syncookies = 1
# Disable netfilter on bridges.
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
net.bridge.bridge-nf-call-arptables = 0
# Controls the default maxmimum size of a mesage queue
kernel.msgmnb = 65536
# Controls the maximum size of a message, in bytes
kernel.msgmax = 65536
# Controls the maximum shared segment size, in bytes
kernel.shmmax = 68719476736
# Controls the maximum number of shared memory segments, in pages
kernel.shmall = 4294967296
net.ipv6.conf.all.proxy_ndp=1
net.ipv4.conf.default.proxy_arp = 0
net.ipv4.conf.all.rp_filter = 0
fs.super-max = 2560
fs.file-max = 262144
```

```
kernel.fairsched-nodes-max = 1538
net.ipv4.neigh.default.gc_thresh2 = 2048
net.ipv4.neigh.default.gc_thresh3 = 4096
net.ipv4.conf.default.send_redirects = 0
net.ipv4.conf.all.send_redirects = 0
net.ipv6.neigh.default.gc_thresh2 = 2048
net.ipv6.neigh.default.gc_thresh3 = 4096
net.nf_conntrack_max = 500000
fs.aio-max-nr = 1048576
```

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 Virtuozzo 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 Virtuozzo 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.

Format

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

Parameters

Name	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 be 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). For more details on the `vzstatrep` utility, see **vzstatrep** (p. 61).

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

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

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 stored.	`/var/log/vzstat`

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

pstat Configuration File

This file (`/etc/vzstat.conf`) lists a number of CPU, memory, and disk-related parameters used by the `pstat` utility. The values assigned to these parameters denote either the warning or the error level for the `pstat` 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.

Parameters

Name	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 percent.	10	Lower	Warning
CPU_SYS	CPU system time, in percent.	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 percent.	8	Lower	Warning
MEM_ZONE_ACT_INACT_FREE_ERR	Size of available memory (free + active + inactive pages), in percent.	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 percent.	75	Lower	Warning
SWAP_FREE_ERR	Free swap space, in percent.	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 the disk, in percent.	20	Lower	Warning
DISK_FREE_INODES_ERR	Free inodes on the disk, in percent.	5	Lower	Error
DISK_FREE_SPACE_WARN	Free disk space, in percent.	20	Lower	Warning
DISK_FREE_SPACE_ERR	Free disk space, in percent.	5	Lower	Error
CT_FAILCNT_DELTA	Number of failed UBC resource allocations for a particular Container between `pstat` screen updates (any resource type counts).	1	Higher	Error

vzrmond Configuration File

The configuration for the `vzrmond` daemon is stored in the `/etc/vzrmond.conf` file. The daemon runs on the Monitor Server, provides remote monitoring capabilities for servers registered with it, and sends alerts to the specified e-mail addresses. Alerts can also be sent from external applications (e.g., via ICQ or by SMS). The configuration file lists a number of parameters, some of which should have user-provided values (from `HOSTS` to `CUSTOM_LIST` in the table below). These values must be enclosed in double quotes and separated by white spaces. The remaining parameters are set to default values which may also be changed. These values are not enclosed in quotes.

Note: To install the daemon, install the `vzrmon` package from the official repository.

Parameters

Name	Description	Default value
HOSTS	The list of hosts to monitor delimited by white spaces. Both hostnames and IP addresses are allowed.	
EMAIL_ADDRESSES	E-mail address(es) to send alerts to. Must be separated by white spaces.	

EMAIL_NOTIFICATIONS	Notification types which can 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	A custom program to send alerts from (e.g., via ICQ or by SMS).	
CUSTOM_LIST	Command-line options for the program specified in `CUSTOM_ACTION`. Must be separated by white spaces.	
POLL_PERIOD	Registered servers polling internal, in seconds.	15
CHK_MAX_FAILS	The maximum number of failed attempts to reach a server, after which the “Server is dead” alert will be 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 maximum 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 maximum 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 start monitoring a server, you should provide valid values for the HOSTS and EMAIL parameters. An external program to send alerts from must be installed on the Monitor Server and its name and command-line options must be set in the CUSTOM_ACTION and CUSTOM_LIST parameters. Alerts will be sent as messages standard 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.

Restart the vzrmond daemon after editing its 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.

Format

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

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 Virtuozzo 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.

Note: The settings in the `/etc/vzbackup.conf` file are valid only for Containers backed up with the `pbackup` utility. These settings are ignored when you back up individual Containers using the `prlctl backup` command or virtual machines using both the `pbackup` utility and the `prlctl backup` command.

Global Parameters

Name	Description	Default value
BACKUP_DIR	The directory where backups are stored. **Note:** To set the backup directory in the	`/var/parallels/backups`

	current version of Virtuozzo, use the `prlsrvctl set` command (p. 51).	
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 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 `pbackup` utility.	`none`
BACKUP_COMPATIBILITY_MODE	If set to "yes" and not commented out, the `pbackup`/`prestore` utilities use `vzbackup`/`vzrestore` for operations on Containers. Otherwise, `prlctl backup` and	`disabled`

	`prlctl restore` commands are used. This parameter is commented out by default.	
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-server Parameters

Name	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 host, 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.

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.

Format

<parameter_name>=<parameter_value>

Parameters

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.

Format

<parameter_name>=<parameter_value>

Parameters

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 ?).

pcompact.conf

The `/etc/vz/pcompact.conf` file is used by the `pcompact` utility to compact virtual disks in Containers.

Format

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

Parameters

Name	Description
THRESHOLD=<*number*>	Compact the virtual disk if unused space on it exceeds **THRESHOLD** percent of the ploop size.
DELTA=<*number*>	Reduce disk space to be compacted by **DELTA** percent of the ploop size.
DEFRAG=<**yes**|**no**>	Perform or skip file system defragmentation.

shaman.conf

The `/etc/shaman/shaman.conf` file is used by the `shaman` utility to manage and monitor High Availability clusters and cluster resources.

Format

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

Options

Name	Description
CLUSTER_NAME	Set the name of the cluster for `shaman` to operate on.

LOG_LEVEL	Set the verbosity level for printed messages: • 0. Print error messages only. • 1. Print warning messages in addition to error messages. • 2. Print information messages in addition to error and warning messages. This is the default verbosity level. • 4. Print debug diagnostic messages in addition to messages from level 0-2.
LOCK_TIMEOUT	Set the timeout for `shaman-monitor` operations (e.g., electing a new master or deciding that a slave node is down). This parameter helps avoid situations when `shaman-monitor` performs a cluster-related operation if someone pulls out the network cable for just a couple of seconds (not longer than the half of the `LOCK_TIMEOUT` value). The default value is 60 seconds. The specified value is added to the value of other timeout-related parameters (see below).
LEASE_CHECK_TIMEOUT_FOR_MASTER	Set the timeout for electing a new master node when the original master node or the `shaman-monitor` daemon fails, or high availability support gets disabled. The default value is 10 seconds.
LEASE_CHECK_TIMEOUT_FOR_SLAVE	Set the timeout after which the master node will consider a slave node as broken if this node and the `shaman-monitor` daemon get down, or high availability support gets disabled. The default value is 10 seconds.
LEASE_LOST_ACTION	Defines the action to perform if `shaman-monitor` loses connection to cluster. This may happen when a node goes online after having been disconnected from network for more than **LOCK_TIMEOUT** seconds. In this case, the watchdog timer has not expired yet, but the cluster is already unavailable, because the master node has prohibited access to the cluster until the node is rebooted. Available values are `crash`, `halt`, `reboot`, and `none` (do nothing).
POOL_CHECK_TIMEOUT	Set the interval for `shaman-monitor` to check for the resources scheduled for relocation. The default value is 30 seconds.
RELOCATION_SKIP_THRESHOLD	Sets the threshold for the number of simultaneously crashed nodes. If the number of simultaneously crashed nodes becomes greater than or equal to the threshold, the master stops relocating resources from the crashed nodes. When the number of simultaneously crashed nodes drops below the threshold, the master automatically resumes relocating resources from the crashed nodes. The threshold can be useful when multiple nodes are being rebooted at the same time. Without it, the master would start relocating resources from all the rebooting nodes. The threshold is set to 3 by default and must be 2 or greater. For clusters with only 3 nodes, the threshold is automatically set to 2.

WATCHDOG_TIMEOUT	Set the interval for the watchdog timer. The watchdog timer is responsible for performing the action defined in **WATCHDOG_ACTION** (see below) if `shaman-monitor` crashes or hangs up. The `shaman-monitor` daemon activates the watchdog timer on its start-up and periodically resets it to the specified value. If something goes wrong with shaman-monitor so that it fails to reset the timer, the watchdog timer counts down until it reaches zero and performs the defined action. Setting the interval to zero disables the watchdog timer. Minimal watchdog timer interval that could be set is 10 seconds. The default value is 120 seconds.
WATCHDOG_ACTION	Defines a sequence of actions to perform after the watchdog timer expires (happens when `shaman-monitor` crashes or hangs up) . When the watchdog timer expires, the first specified action is attempted. If unsuccessful, the next specified action is attempted and so on. If the last specified action is unsuccessful, then the action specified in the `/sys/kernel/watchdog_action` file is performed. At least one action must be specified. Multiple actions must be separated with commas. Available actions are listed in the `/sys/kernel/watchdog_available_actions` file. The default sequence is `netfilter, reboot`.

Virtuozzo Scripts

This section provides information on scripts you can use in Virtuozzo 6.

Overview

The table below lists the scripts you can use to customize the behavior of Containers and virtual machines and define High Availability actions.

Scripts

Name	Description
/vz/private/<*CT_ID*>**/scripts/**<*script_name*>	Container action scripts. These scripts allow you to run user-defined actions on particular events. The currently defined actions are `start`, `stop`, `mount`, `unmount`.
/etc/vz/conf/dists/scripts/<*script_name*>	Scripts to be executed on performing certain Container-related operations (e.g., when 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	Virtuozzo start/stop script. This script is responsible for proper Virtuozzo startup and shutdown procedures, including modules loading and Container start/stop procedures.

/etc/vmprivate/<*vm_name*>**/scripts/**<*script_name*>	Virtual machine actions scripts. These scripts allow you to run user-defined actions when starting and stopping virtual machines.
/usr/share/shaman	Scripts used by the shaman utility to manage and monitor High Availability clusters and cluster resources.

Container 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. Virtuozzo allows you to automate procedures like the above by using *action scripts* below.

Scripts

Name	Description
global mount	This script runs immediately after `prlctl` 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 script is called in the server context.
start	After `prlctl` 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 `prlctl` is about to dismount the Container private area. It also runs in the servercontext.

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

Caution: When executing `prlctl start`, both `mount` and `start` scripts run. However, if the `start` script fails then neither `stop` nor `umount` scripts will run. As a result, `prlctl` 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 `prlctl`.

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. `prlctl`) 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. 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 prlctl 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 information from `/proc` may be not functional.

Virtual Machine Action Scripts

In Virtuozzo 6, you can create custom scripts and configure them to run when you start or stop virtual machines. The following custom scripts are currently supported:

- *prestart*: this script is executed before a virtual machine is started.
- *poststart*: this script is executed after a virtual machine is started.
- *prestop*: this script is executed before a virtual machine is stopped.

- *poststop*: this script is executed after a virtual machine is stopped.

Action scripts specific for virtual machines must be placed to the `/scripts` subdirectory in the virtual machine's home directory. For example, assuming that you created the `MyVM` virtual machine in the default directory, the paths to the scripts must be as follows:

- `/var/parallels/MyVM/scripts/prestart`
- `/var/parallels/MyVM/scripts/poststart`
- `/var/parallels/MyVM/scripts/prestop`
- `/var/parallels/MyVM/scripts/poststop`

The information to scripts is passed using the `UUID` and `VM_HOME` variables, where `UUID` is the ID of the virtual machine you start or stop and `VM_HOME` is the full path to the virtual machine's directory.

Note: Action scripts you create for a virtual machine cannot contain `prlctl` commands intended for performing operations on this virtual machine (e.g., assigning an IP address to the virtual machine).

High Availability Scripts

Virtuozzo comes with a number of scripts used by the `shaman` utility to manage and monitor cluster resources. There are two types of scripts:

- **Common scripts**. The common scripts are located in the `/usr/share/shaman` directory and used by the `shaman` utility to call resource-specific scripts.
- **Resource-specific scripts**. For each common script, there are one or more resource-specific scripts. Resource-specific scripts are peculiar to each cluster resource and located in separate subdirectories. For virtual machines and Containers, these directories are `/usr/share/shaman/vm-` and `/usr/share/shaman/ct-`, respectively. Resource-specific scripts are used to perform various actions on cluster resources.

For the full list of scripts and their descriptions, see the `shaman-scripts` man page.

Virtuozzo Utilities

This section provides information on utilities that can be used to manage Virtuozzo parameters.

prlsrvctl

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

Syntax

prlsrvctl [*command* [*options*] [**-l**, **--login** [*user*[**:***passwd*]**@**]*server*[**:***port*]]]

Options

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 host and execute a command there. 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:*port*	The remote server IP address or hostname and port number, If port number is omitted, the default port will be used.

Note: To display help, enter `prlsrvctl` on the command-line without any options.

prlsrvctl info

Displays the host and Virtuozzo configuration information.

Syntax

prlsrvctl info

Remarks

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

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

prlsrvctl install-license

Installs the Virtuozzo license on the host.

Syntax

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

Options

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

prlsrvctl net

The `prlsrvctl net` command is used to create and configure virtual networks.

Subcommands

Name	Description
net add (p. 48)	Creates a new virtual network
net set (p. 49)	Configures the parameters of an existing virtual network.
net del (p. 50)	Removes an existing virtual network.
net list (p. 50)	List the available virtual networks.

net add

Creates a new virtual network.

Syntax

```
prlsrvctl net add <vnetwork_ID> [-i,--ifname <if>] [-m,--mac <mac_address>]
                    [-t,--type <bridged|host-only>] [-d,--description <desc>]
                    [--ip <IP_address>[/<mask>]] [--dhcp-server <on|off>]
                    [--dhcp-ip <IP_address>] [--ip-scope-start <IP_address>]
                    [--ip-scope-end <IP_address>]
                    [--ip6 <IP_address>[/<mask>]] [--dhcp6-server <on|off>]
                    [--dhcp-ip6 <IP_address>] [--ip6-scope-start <IP_address>]
                    [--ip6-scope-end <IP_address>]
```

Options

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 host to which this virtual network should be bound.
-m,--mac <*mac_address*>	The MAC address of a virtual network adapter on the host to which this virtual network should be bound.
-t,--type <**bridged**|**host-only**>	The type of the virtual network to create. Possible values are: • `bridged`. A virtual machine and Container connected to this type of virtual network appears as an independent computer on the network. • `host_only` (default). A virtual machine and Container connected to this type of virtual network can access only the host and the virtual machines and Containers connected to the same virtual network.
-d,--description <*desc*>	A user-defined description of the virtual network. Descriptions with white spaces must be enclosed in quotation marks.
--ip <*IP_address*>[**/**<*mask*>] **--ip6** <*IP_address*>[**/**<*mask*>]	Set an IPv4/IPv6 address and subnet mask for the Virtuozzo virtual adapter.
--dhcp-server <**on**|**off**> **--dhcp6-server** <**on**|**off**>	Enable or disable the Virtuozzo virtual DHCPv4/DHCPv6 server.
--dhcp-ip <*IP_address*> **--dhcp-ip6** <*IP_address*>	Set an IPv4/IPv6 address for the Virtuozzo virtual DHCPv4/DHCPv6 server.
--ip-scope-start <*IP_address*> **--ip-scope-end** <*IP_address*> **--ip6-scope-start** <*IP_address*> **--ip6-scope-end** <*IP_address*>	Set the starting and ending IPv4/IPv6 addresses for the DHCPv4/DHCPv6 pool. The virtual machines and Containers connected to the network you are creating will automatically receive their IPv4/IPv6 addresses from the respective DHCPv4/DHCPv6 pool.

net set

Configures the settings of an existing virtual network.

Syntax

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

Options

Name	Description
<vnetwork_ID>	The name of the virtual network to modify.
-i,--ifname *<if>*	The name of a physical network adapter on the host to which this virtual network should be bound.
-m,--mac *<mac_address>*	The MAC address of a virtual network adapter on the host to which this virtual network should be bound.
-t,--type <bridged|host-only>	The type of the virtual network to create. Possible values are: • `bridged`. A virtual machine and Container connected to this type of virtual network appears as an independent computer on the network. • `host_only`. A virtual machine and Container connected to this type of virtual network can access only the host and the virtual machines and Containers connected to the same virtual network.
-d,--description *<desc>*	A user-defined description of the virtual network. Descriptions with white spaces must be enclosed in quotation marks.
-n, --name *<new_name>*	A new name for the virtual network. Use this parameter if you would like to rename the virtual network.

net del

Deletes an existing virtual network.

Syntax

prlsrvctl net del *<vnetwork_ID>*

Options

Name	Description
<vnetwork_ID>	The name of the virtual network to delete.

net list

Lists the existing virtual networks.

Syntax

prlsrvctl net list

Options

None.

prlsrvctl problem-report

Generates and displays problem reports.

Syntax

```
prlsrvctl problem-report
```

Options

None.

Remarks

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

prlsrvctl set

Configures Virtuozzo 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-tmpdir <tmp_dir>]
                [--backup-path <path>] --idle-connection-timeout <timeout>
                [--verbose-log <on|off>] [--cluster-mode <on|off>]
                [--vm-cpulimit-type <full|guest>]
```

Options

Name	Description
--mem-limit <auto>|<*size*>	Sets the upper limit of the memory size that can be reserved for use by virtual machines. 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 <low | normal | high>	The lowest allowable security level that can be used to connect to the host. 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 <on | off>	Enables/disables the participation in the Customer Experience Program (CEP). The following options are available: • `on` — enables CEP. • `off` — disables CEP.
--mng-settings <allow | deny>	Allows to grant or deny permission to new users to modify Virtuozzo preferences. By default, only administrators of the host OS can modify Virtuozzo preferences. When a new user profile is created (this happens when a user logs in to the host 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 <host | vm>**	Allows to set the assignment mode for the specified VTd device. The following options are available: • `host` — assign the device to the host. • `vm` — assign the device to virtual machines.
--backup-storage `[`*user*`[:`*passwd*`]@]`*server*`[:`*port*`]]`	The default backup server where to store virtual machine backups.
--backup-path *<path>*	The name and path of the default directory on the backup server where to store virtual machine backups.
--verbose-log <on | off>	Turns the verbose output for the command on or off.
--cluster-mode <on | off>	Turns the cluster mode on or off.
--idle-connection-timeout *<timeout>*	Sets a timeout interval in seconds after which, if no data has been received from the storage server or backup client, the process of backup/restore is terminated.
--backup-tmpdir *<tmp_dir>*	Specifies a temporary directory where special snapshots created during virtual machine backup will be stored. This may be necessary so as not to run out of storage space on physical servers where most of the storage space is allocated to virtual machines and very little is left for the server itself.
--vm-cpulimit-type <full | guest>	Specifies the type of virtual machine threads to be affected by the CPU limit: • `full` (default) — both hardware emulation and guest OS threads are limited; • `guest` — only guest OS threads are limited.

	With the `guest` option, the guest OS is guaranteed to have all the resources implied by the VM configuration. At the same time, the VM's hardware emulation threads spend additional resources of the host. For example, for a VM with two 2.8 GHz vCPUs, switching to `guest` means that VM's guest applications will have all the resources of two 2.8 GHz vCPUs at their disposal. **Notes:** 1. Some types of guest applications, like voice-over-IP software, significantly increase expenses on hardware emulation threads. 2. After changing this parameter, restart running virtual machines for the changes to take effect.

prlsrvctl shutdown

Shuts down the Virtuozzo component responsible for managing virtual machines and Containers. No operations on virtual machines and Containers are possible.

Syntax

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

Options

Name	Description
-f, --force	Specifies whether the shutdown operation should be forced. If one or more virtual machines and Containers 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 Virtuozzo component.

prlsrvctl statistics

Obtains Virtuozzo statistics.

Syntax

prlsrvctl statistics [**-a**, **--all**] [**--loop**] [**--filter** *<name>*]

Options

Name	Description
--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.
-a, --all	*This parameter is not currently used.*
--filter <*name*>	*This parameter is not currently used.*

prlsrvctl usb

The `prlsrvctl usb` command is used to permanently assign a USB device to a specific virtual machine. A permanently assigned USB device will be connected to the virtual machine automatically on server restart. This functionality works only with virtual machines (not Containers).

Subcommands

Name	Description
usb list (p. 54)	Lists USB devices connected to the server together with the information about their virtual machine assignments for the current user.
usb set (p. 55)	Permanently assigns a USB device to the specified virtual machine.
usb del (p. 55)	Removes a previously created USB device assignment.

usb list

Lists the USB devices connected to the physical server.

Syntax

prlsrvctl usb list

Options

None.

Returns

A list of USB devices in tabular format with the following columns:

Name — the USB device name.

ID — a string that uniquely identifies the USB devices on the physical server. The ID never changes even if the device is disconnected from the server and then reconnected again. Please note that if a device ID is listed in quotes, they are a part of the ID and must be included in other calls that use it as an input parameter.

VM UUID — a universally unique ID of the virtual machine to which this USB device is permanently assigned. If a USB device is not assigned to any virtual machine, this column will be empty.

usb set

Permanently assigns a USB device to the specified virtual machine. A permanently assigned USB device will be connected to the virtual machine automatically on server restart. The USB device assignment is performed for the current user only. Other users may create their own USB device assignments. This functionality works only with virtual machines (not Containers).

Syntax

prlsrvctl usb set <*usb_dev_ID*> <*vm_ID*|*vm_name*>

Options

Name	Description
<*usb_dev_ID*>	The USB device ID. To obtain the list of USB devices connected to the server use the **usb list** (p. 54) command.
<*vm_ID*|*vm_name*>	The universally unique ID or name of the virtual machine to which to assign the USB device.

usb del

Deletes a USB device assignment previously created with the **usb set** (p. 55) command. The USB device assignment is performed on the user level, so if you remove an assignment, it will only be removed for the current user. Other users may have their own USB devices assignments, which will not be affected.

Syntax

prlsrvctl usb del <*usb_dev_ID*>

Options

Name	Description
<*usb_dev_ID*>	The USB device ID. To see the current USB device assignments for the current user use the **usb list** (p. 54) command.

prlsrvctl user list

Displays the list of Virtuozzo users. Only those users are displayed who has created at least one virtual machine and Container.

Syntax

```
prlsrvctl user list [-o,--output <name|mng_settings|def_vm_home>]
```

Options

Name	Description
-o,--output <name|mng_settings|def_vm_home>	Fields to include in the output. The following fields are available: • `name` — User name. • `mng_settings` — Indicates whether the user is allowed to modify Virtuozzo preferences. • `def_vm_home` — The user default virtual machine folder. The fields must be specified in lowercase.

prlsrvctl user set

Configures the profile of a Virtuozzo user.

Syntax

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

Options

Name	Description
<*UUID*|*name*>	The user UUID (universally unique ID) or name.
--def-vm-home <*path*>	The default virtual machine and Container directory name and path.
--mng-settings <allow|deny>	Specifies whether the user should be allowed to modify Virtuozzo preferences.

privnet

The `prlsrvctl privnet` command is used to manage private networks on physical servers.

Subcommands

Name	Description
add (p. 57)	Creates a new private network.
set (p. 57)	Configures the parameters of an existing private network.
del (p. 58)	Removes an existing private network.
list (p. 58)	Lists the available private networks.

add

Creates a new private network.

Syntax

```
prlsrvctl privnet add <private_network_ID> [-a,--ipadd <addr>[/mask]] [--global <yes|no>]
```

Options

Name	Description
<private_network_ID>	The private network ID.
-a,--ipadd *<addr>*[*/mask*]	Add a subnet to the private network. The network can have multiple subnets.
--global <**yes**|**no**>	Make the private network weak, allowing access to and from external networks.

set

Configures an existing private network.

Syntax

```
prlsrvctl privnet set <private_network_ID> [-a,--ipadd <addr>[/mask]]
                      [-d,--ipdel <addr>[/mask]] [--global <yes|no>]
```

Options

Name	Description
<private_network_ID>	The private network ID.
-a,--ipadd *<addr>*[*/mask*]	Add a subnet to the private network. The network can have multiple subnets.
-d,--ipdel *<addr>*[*/mask*]	Delete the specified subnet from the private network.

--global <yes | no>	Make the specified private network weak, allowing access to and from external networks.

del

Deletes a private network from the physical server.

Syntax

prlsrvctl privnet del *<private_network_ID>*

Options

Name	Description
<private_network_ID>	The ID of the private network to delete.

list

Lists the private networks that exist on the physical server.

Syntax

prlsrvctl privnet list

Options

None.

vzlicload

Installs and removes Virtuozzo licenses.

Syntax

vzlicload [*options*]

Options

Name	Description
-p, --product-key *<key>*	Installs the specified Virtuozzo 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 (for details, see **vzlicview** (p. 59)).
-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 Virtuozzo 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.

Syntax

vzlicupdate [*options*]

Options

Name	Description
-a, --activate <*activation_code*>	Activates the Virtuozzo 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 Key Authentication (KA) server responsible for updating Virtuozzo licenses, activating Virtuozzo installations, and transferring licenses from the Source Server to the Destination Server. If not specified, the `ka.virtuozzo.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.

Syntax

vzlicview [*options*]

Options

Name	Description
-p, --product-key <*key*>	Displays the license information contained in the specified Virtuozzo product key.
-f, --license-file <*file*>	Displays the license information from the specified Virtuozzo 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 Virtuozzo licenses:

Name	Description
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).

Syntax

vznetcfg <*command*>

Options

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_VN_name*> 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*>	Creates a new VLAN adapter, associates it with the VLAN ID of <*index*> (where <*index*> 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 Virtuozzo 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.

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).

Syntax

vzstatrep [*options*]

Options

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 must use this one.
--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. For details on the `/etc/vzstatrep.conf` file, see **vzstatrep Configuration File** (p. 36).

pstat

Virtuozzo real-time monitoring utility. Displays the status and load of the system, including disk, network, CPU, and memory parameters (swap included). The status is updated in a preset time interval. `pstat` also provides a sortable list of running virtual machines and Containers together with their resources consumption statistics. Utility's display output can be customized.

Syntax

pstat [**-l**] [**-d** <*X*>] [**-p** <*CT_ID*>] [**-b**|**-v**] [**-t**] [**-a**]

Options

Name	Description
-l	Print information once and exit immediately.
-d <*X*>	Specify 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 the specified *CT_IDs*. This flag can be given up to twenty times as follows: `-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.
-a	Display the current disk input and output (I/O) statistics for virtual machines and Containers.
-i	Display the IO accounting information for virtual machines and Containers.
-m	Display disc statistics for all file system types. By default, the statistics is shown for ext2, ext3, ext4, and reiserfs.
-o	Filter the output by the specified parameters. You can specify multiple parameters and separate them by commas. The list of available parameters is given below.
-O	Filter the output by the specified parameters. Unlike the `-o` option, the produced output already contains some default columns (for example, Container ID and IP address). The list of available parameters is given below.
-n	Display network statistics.
-s	Filter the output by the specified keys. The list of available parameters is given below.
-c	Display I/O statistics in the specified units of measurement: `B` (bytes), `K` (kilobytes), `M` (megabytes), or `G` (gigabytes).

You can use the following parameters with the `-o`, `-O`, and `-s` options to filter the information related to virtual machines and Containers:

Name	Column	Description
id	`CTID`	virtual machine or Container ID
st	`ST`	virtual machine or Container status.
vm	`%VM`	Virtual memory usage, in per cent of the total memory on the Node. It's displayed in the form of "actual usage/barrier". This parameter corresponds to the `PHYSPAGES` VSwap parameter.

km	%KM	Kernel memory usage, in per cent of the normal zone size. It's displayed in the form of "actual usage/barrier". This parameter corresponds to the `kmemsize` UBC parameter.
sw	%SW	Swap space usage, in per cent of the total swap space on the Node. It's displayed in the form of "actual usage/barrier". This parameter corresponds to the `SWAPPAGES` VSwap parameter.
swapin	SWAPIN	I/O operations with the swap space, in operations per second.
proc	PROC	Process information. It's displayed in the form of "running processes/total processes/barrier".
cpu	CPU	CPU usage. It's displayed in the form of "actual CPU usage/guaranteed CPU usage". If more than one processor is installed on the server, all processors are considered as 100%.
cpu_iowait	CPU IOWAIT	Percentage of time that Container processes lost waiting for I/O.
cpu_iowait_acc	ACC IOWAIT	Percentage of time that Container processes lost waiting for I/O, accumulated since utility start.
sock	SOCK	Sockets usage calculated as the sum of UBC numtcpsock and numothersock parameters. It's displayed in the form of "open sockets/barrier".
fcnt	FCNT	Fail counters—that is, the number of UBC fail counters for all resources.
mlat	MLAT	Maximum process scheduling latency, in milliseconds. In this case, the latency means the maximum time a process in a virtual machine or Container is waiting for the CPU.
io	IO	I/O rate and limit, in bytes, kilobytes, megabytes, or gigabytes per second.
iops	IOPS	I/O operations rate and limit, in operations per second.
iow	IOW	Transfer rate at which data is written to the virtual machine or Container.
ior	IOR	Transfer rate at which data is read from the virtual machine or Container.
iowt	IOWT	Total amount of data that was written to the virtual machine or Container.
iort	IORT	Total amount of data that was read from the virtual machine or Container.
ios	IOS	Synchronization rate.
iod	IOD	Flushing rate of dirty pages (pages that have been changed but are not yet written to the virtual machine or Container disk).
ioused	IOUSED%	Percentage of time the disks are used by the virtual machine or Container.
iowait	IOWAIT%	Percentage of time when at least one I/O transaction in the virtual machine or Container is waiting for being served.

rx	RX	Incoming traffic rate, in megabytes per second.
tx	TX	Outgoing traffic, in megabytes per second.
rxt	RXT	Total amount of incoming traffic, in megabytes.
txt	TXT	Total amount of outgoing traffic, in megabytes.
rxp	RXP	Incoming traffic, in packets per second.
txp	TXPT	Outgoing traffic, in packets per second.
rxpt	RXPT	Total amount of incoming packets.
txpt	TXPT	Total amount of outgoing packets.

Displayed Information

`pstat` can display the following information:

Type	Description	Example	Toggled 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	Total number of virtual machines and 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 virtual machines and Containers and by the server (shown as `CT0`), the CPU time spent in the user mode, in the system mode, and being idle, and the maximal/average scheduling latency in ms. The scheduling latency is the time spent by the processes in the system awaiting for scheduling.	`CPU [ OK ]: CTs 43%, CT0 12%, user 41%, sys 13%, idle 45%, lat(ms) 3/2`	**c**
Mem	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	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	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	Swap cache information includes the number of addition, deletion, and find operations respecting the swap cache.	`Swap cache: add 0, del 0, find 0/0`	**w, W**
Network information	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	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	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**

Disk I/O statistics	Shows disk input and output statistics for virtual machines and Containers. The following statistics is displayed:
	• `IOUSED%`: the percentage of time the disks are used by the virtual machine or Container.
	• `IOWAIT%`: the percentage of time when at least one I/O transaction in the virtual machine or Container is waiting for being served.
	• `IO`: the current speed of disk I/O in the virtual machine or Container and the I/O limit set for this virtual machine or Container, if any. The value can be displayed in bytes, kilobytes, megabytes, or gigabytes per second, depending on the units you used to set the I/O limit.
	• `IOPS`: the current speed of disk I/O in the virtual machine or Container and the I/O limit set for this virtual machine or Container, if any. The value is displayed in operations per second.
	• `CPU_IOWAIT`: percentage of time that Container processes lost waiting for I/O.
	• `CPU_IOWAIT_ACC`: percentage of time that Container processes lost waiting for I/O, accumulated since utility start.
	• `SWAPIN`: I/O operations with the swap space. The value is measured in operations per second.
		`IOUSED%`
		`5.00`
		`IOWAIT%`
		`0.00`
		`IO`
		`2/100MB/s`
		`IOPS`
		`315/--/s`
		`CPU IOWAIT`
		`25.0%`
		`ACC IOWAIT`
		`25.0%`
		`SWAPIN`
		`0.0/s`
			A
			A
			A
			A

Interactive Controls

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

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 uninterruptible 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 virtual machine and Container IP addresses/hostnames.	
i	Toggle display of idle virtual machines and 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.	
A	Toggle display of disk input and output statistics for virtual machines and Containers.	
I	Toggle display of input and output accounting for virtual machines and Containers.	
X	Toggle display of network statistics for virtual machines and Containers.	
d, D	Toggle/expand display of disk usage and activity information.	
0	Toggle display of statistics for the server itself.	
o	Sort key. You can use one of the following sort option keys:	
	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
	t	Sort by virtual machine or Container status. virtual machines and Containers which probably are unusable or unstable (increasing UBC failure counters or very high scheduling latency) will be shown first.
	s	Sort by the number of open sockets
	m	Sort by memory latency
	v	Sort by virtual memory usage
	k	Sort by kernel memory usage

vzreboot

`vzreboot` is used to update Virtuozzo without rebooting the physical server.

Syntax

vzreboot [*kernel_version*]

When run without options, the command checks the GRUB configuration file to determine the default kernel, loads this kernel into memory, and then boots the new kernel. You can also specify the kernel version manually (e.g., `vzreboot 2.6.32-042stab054.2`) to load and boot to the indicated kernel instead of the one set as default in the GRUB configuration file.

High Availability Utilities

This section describes the utilities you can use to manage High Availability (HA) support for physical servers and virtual machines and Containers hosted on them.

shaman General Syntax

The `shaman` utility is used to manage and monitor clustering resources (virtual machines and Containers) on serves with Virtuozzo. You can use this utility to enable or disable High Availability (HA) support for servers, monitor global cluster events, and view the overall cluster statistics.

Syntax

shaman -c <*cluster_name*> [*options...*] [*command* [*command_options*]]

General Options (can be used with all commands)

Name	Description
-c <*cluster_name*>	Name of the cluster for which to execute the command.
-l, --log <*path*>	Full path to the log file.
-v, --verbose	Increase the log verbosity level.
-q, --quiet	Disable printing of messages to the console.
-V, --version	Display the `shaman` version.
-h, --help	Display the list of allowed options for the `shaman` utility.

Commands

Name	Description
join	Enable HA support for the server.

leave	Disable HA support for the server.
stat	Displays information about the current cluster status and clustering resources.
top	Displays a dynamic real-time view of the cluster and clustering resources.

shaman join

The `shaman join` command is used to enable High Availability (HA) support for physical servers.

Syntax

shaman -c <*cluster_name*> **join** [**-r, --ignore-res**]

Options

Name	Description
-c <*cluster_name*>	Name of the cluster that contains the server for which you want to enable HA support. If you omit the option. the command is executed for the cluster specified in the `/etc/shaman/shaman.conf` configuration file.
join	Tells the `shaman` utility to enable HA support for the server. Enabling HA support means that `shaman` will control all virtual machines and Containers that are already hosted on the server and that you will create on it in the future.
-r, --ignore-res	Enable HA support for the server but leave it disabled for existing virtual machines and Containers. In this case, `shaman` will control only the virtual machines and Containers that you will create on the server after enabling HA support.

shaman leave

The `shaman leave` command is used to disable High Availability (HA) support for physical servers.

Syntax

shaman -c <*cluster_name*> **leave** [**-a, --addr** <*IP_address*>]

Options

Name	Description
-c <*cluster_name*>	Name of the cluster that contains the server for which you want to disable HA support.
leave	Tells the `shaman` utility to disable HA support for the server.

-a, --addr <*IP_address*>	Disables HA support for the server with the IP address of *ip_address*. If you omit this option, the **shaman** utility will turn off HA support for the local node.

shaman stat

The **shaman stat** command is used to display information about the current cluster status and clustering resources. To display a dynamic real-time view of the cluster and clustering resources, use the **shaman top** command.

Syntax

shaman -c <*cluster_name*> **stat** [**-X**, **--xml**] [**-g**, **--group**] [**-n**, **--node-ip** <*IP_address*>] [**-N**, **--node-id** <*ID*>]

Options

Name	Description
-c <*cluster_name*>	Name of the cluster for which to execute the command.
-X, --xml	Display cluster statistics in XML format.
-g, --group	Group clustering resources according to their state into three groups: • **Active**. Healthy virtual machines and Containers hosted in the cluster. • **Broken**. Virtual machines and Containers that could not be relocated from a failed server to a healthy one. • **Pool**. Virtual machines and Containers waiting for relocation from a failed server to a healthy one.
-n, --node-ip <*IP_address*>	Display statistics only for the server with the specified IP address.
-N, --node-id <*ID*>	Display statistics only for the server with the specified ID.

shaman top

The **shaman top** command is used to display a dynamic real-time view of the cluster and clustering resources.

Syntax

shaman -c <*cluster_name*> **top** [**-g**, **--group**] [**-n**, **--node-ip** <*IP_address*>] [**-N**, **--node-id** <*ID*>]

Options

Name	Description
-c <*cluster_name*>	Name of the cluster for which to execute the command.
-g, --group	Group clustering resources according to their state into three groups: • **Active**. Healthy virtual machines and Containers hosted in the cluster. • **Broken**. Virtual machines and Containers that could not be relocated from a failed server to a healthy one. • **Pool**. Virtual machines and Containers waiting for relocation from a failed server to a healthy one.
-n, --node-ip <*IP_address*>	Display statistics only for the server with the specified IP address.
-N, --node-id <*ID*>	Display statistics only for the server with the specified ID.

Interactive commands

Name	Description
g	Group or ungroup cluster resources.
v	Show or hide additional information.
ENTER, SPACE	Update the screen with statistics.
q	Quit the command.
h	Display the help screen.

Virtuozzo Updates

Virtuozzo allows quick and easy updates with the `yum` utility standard for RPM-compatible Linux operating systems. For more information on `yum`, see **Updating Virtuozzo** in the *Virtuozzo 6 User's Guide* and the `yum` manual page.

CHAPTER 3

Managing Containers

Virtuozzo Containers can be managed using the `prlctl` command-line utility. The utility is installed on the host during product installation.

In This Chapter

Matrix of Virtuozzo Command-Line Utilities

The table below contains the full list of Virtuozzo command-line utilities and command you can use for managing Containers.

Note: Starting from Virtuozzo 6, numeric Container IDs and alphanumeric, case-sensitive Container names are interchangeable. Any command that requires a Container ID can be supplied that Container's name instead.

General Utilities

Name	Description
prlctl	Utility to control Containers.
prlctl list	Utility to view a list of Containers existing on the server with additional information.

Container Migration Utilities

Name	Description
prlctl clone	Command for the local cloning of the Containers.
pmigrate	Utility for migrating physical servers to Containers and for moving Containers between hosts.

Container Backup Utilities

Name	Description
pbackup	Utility to back up Containers.
prestore	Utility to restore backed up Containers.
prlctl backup	Command to back up individual Containers.
prlctl restore	Command to restore individual Containers.

Template Management Utilities

Name	Description
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.
vzcreaterepo	Utility to set up repositories for RHEL-like distributions, including Red Hat Enterprise Linux 5.

Supplementary Utilities

Name	Description
vzps, vztop	Utilities working as the standard `ps` and `top` utilities, with Container-related functionality added.
pnetstat	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.
pstat	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.
pfcache	Memory and IOPS deduplication management utility.
vzctl convert	Utility to convert Containers based on VZFS to the Container-in-an-image-file layout.

pcompact	Utility to compact Containers by removing unused blocks from their virtual disks.

prlctl

`prlctl` is the primary tool for Container management. To use it, you have to log in to the server as the root user.

Syntax

```
prlctl <subcommand> <CT_ID>
prlctl --version
prlctl --help
```

Subcommands

Name	Description
create	Creates a new Container.
delete, destroy	Deletes a Container.
mount	Mounts the Container private area and executes the Container mount script.
umount	Unmounts the Container private area and executes the unmount script.
start	Starts a Container.
stop	Stops a Container.
restart	Restarts a Container.
status	Displays the Container status.
set	Sets Container parameters: resource control settings, hostname, IP addresses, and so on.
enter	Logs in to a Container without knowing its root password.
exec	Runs arbitrary commands in a Container without logging in to it.
suspend	Saves the state of a running Container in a dump file.
resume	Restores a Container from its dump file.
snapshot, snapshot-list, snapshot-switch, snapshot-delete	Creates and manages Container snapshots.
console	Creates a command prompt channel to a Container.

Options

Name	Description
--version	Displays the `prlctl` package version currently installed on the server.
--help	Displays the usage information about `prlctl`.

prlctl console

Creates a command prompt channel to a Container. Allows to log in to and execute commands in running Containers; provides Container startup/shutdown information that may be used for troubleshooting purposes. Logging in to Containers requires a virtual terminal (e.g., mingetty) to be installed in the Container.

Note: To exit from the console, press Esc and then ".".

Syntax

prlctl console *<CT_ID>*

Options

Name	Description
<CT_ID>	Container ID.

prlctl create

This command is used to create new Containers.

Syntax

prlctl create *<CT_ID>* **--vmtype ct** [*options*]

With this command, you can create regular Containers. A unique Container ID is required for this command.

Note:

1. Container IDs from 1 to 100 are reserved for internal Virtuozzo needs. Do not use IDs from 1 to 100 when creating Containers.

2. If a Container is created with an alphanumeric name, the next available numeric ID will be used as its private area directory name. If a Container is created with a numeric ID, the same number will be used as that Container's private area directory name.

Options

Name	Description
<CT_ID>	ID to assign to the new Container.
--vmtype ct	Tells the `prlctl create` command to make a Container. If the option is omitted, a virtual machine is created instead.
--ostemplate *<name>*	OS EZ template to use for creating the Container. If omitted, this value is taken

	from the `DEF_OSTEMPLATE` parameter in the global Virtuozzo configuration file.
--config <*name*>	Container sample configuration file to use for creating the Container. Sample configuration files are located in `/etc/vz/conf` and have names in the format `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 Virtuozzo configuration file, you will have to set resource control parameters for the Container using the `prlctl set` command.
--uuid <*uuid*>	A custom UUID to assign to the Container.
--private <*path*>	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 `$VEID` string which will be replaced by numeric Container ID value. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--root <*path*>	Path to the mount point of the Container root directory. This option is used to override default path to Container root directory from the `/etc/vz/vz.conf` configuration file (`VE_ROOT` variable). The argument can contain `$VEID` string which will be replaced by numeric Container ID value. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--ipadd <*addr*>[*/mask*]	IP address and subnet mask to assign to the Container. If you omit this option, you can set an IP address for the Container later using the `prlctl set` command. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--hostname <*name*>	Hostname to assign to the Container. If you omit this option, you can assign a hostname to the Container later using the `prlctl set` command. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--description <*desc*>	Custom Container description. Descriptions with white spaces must be enclosed in quotation marks. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--skip_app_templates	Do not install the application templates specified in the Container sample configuration file. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.

--velayout <*velayout_ID*>	Defines the Container layout. *velayout_ID* can be one of the following: • 5: create a Container with the new Container-in-a-file layout. The new layout has been redesigned to store all files of a Container in a single image, similar to a virtual machine's hard disk. This is the default and recommended layout in Virtuozzo 6. • 4: create a Container based on VZFS, the obsolete layout used in earlier versions of Parallels Server Bare Metal. If you omit this option, the layout will be defined by the VEFSTYPE parameter in the global configuration file /etc/vz/vz.conf. For the Container-in-an-image layout, this parameter should be set to ext4.

prlctl delete, destroy

These commands are used to delete Containers from the server.

Syntax

prlctl delete <*CT_ID*>
prlctl destroy <*CT_ID*>

Options

Name	Description
<*CT_ID*>	Container ID.

When executed, prlctl delete/prlctl 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.

Note: A Container must be stopped before its private area can be unmounted.

prlctl exec, enter

These commands are used to run arbitrary commands in a Container being authenticated as root on the server.

Syntax

prlctl exec <*CT_ID|CT_name*> <*command*>
prlctl enter <*CT_ID|CT_name*>

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.

Options

Name	Description
<CT_ID|CT_name>	Container ID or name.

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

`prlctl enter` is similar to `prlctl exec /bin/bash`. The difference between the two is that `prlctl 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.

prlctl mount, umount

The `prlctl mount` 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 `prlctl start` command mounts the Container private area automatically.

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

Syntax

prlctl mount *<CT_ID>*
prlctl umount *<CT_ID>*

Options

Name	Description
<CT_ID>	Container ID.

prlctl move

Moves Container's private area to a new location on the same server. The Container can be stopped, suspended or running.

Syntax

prlctl move *<CT_ID|CT_name>* **--dst** *<path>*

Options

Name	Description
<*CT_ID*|*CT_name*>	Container ID or name.
--dst <*path*>	Path to Container's new private area.

prlctl problem-report

Generates a problem report for the specified Container and either sends it to technical support or displays it on the screen.

Syntax

```
prlctl problem-report <CT_ID> -d,--dump
prlctl problem-report <CT_ID> -s,--send [--proxy [user[:passwd]@]<proxyhost>[:port]] [--no-proxy]
```

Options

Name	Description
<*CT_ID*>	ID of the Container for which to generate the problem report.
-d, --dump	Collect technical data about the specified Container and display it on the screen. You can also pipe the output to a file and then send it to technical support to analyze your problem.
-s, --send	Send the generated problem report to technical support.
--proxy [*user*[:*passwd*]@]<*proxyhost*>[:*port*]	Use the specified information to send the generated report through a proxy server, if you use one to connect to the Internet.
--no-proxy	Do not use a proxy server to send the generated report. This is the default behavior, so you can omit this parameter.

prlctl register, unregister

The `register` command is used to register a Container with Virtuozzo.

The `unregister` command removes a Container from the Virtuozzo registry.

Syntax

```
prlctl register <path> [--preserve-uuid <yes|no>]
prlctl unregister <CT_ID>
```

Options

Name	Description
<path>	Full path to the Container directory.
<CT_ID>	ID of the Container to remove from the Virtuozzo registry.
--preserve-uuid <yes|no>	Specifies what to do with the Container UUID (universally unique identifier). If you specify `yes`, the UUID is preserved. If you specify `no`, the UUID is regenerated. **Note:** By default, UUIDs are regenerated.

Remarks

- Use the `register` command when you have a Container on the server that does not show up in the list of the Containers registered with the Virtuozzo. This can be a Container that was previously removed from the registry or that was copied from another location.
- The `unregister` command removes a Container from the Virtuozzo registry, but does not delete the Container files from the server. You can re-register the Container later using the `register` command.

prlctl reinstall

Recreates a Container from scratch according to its configuration file. Copies old private area content to the `/vz/root/<CT_ID>/old` directory.

Syntax

```
prlctl reinstall <CT_ID> [--skipbackup] [--resetpwdb] [--scripts <script> [...]]
prlctl reinstall <CT_ID> [--listscripts] [--desc]
```

Options

Name	Description
<CT_ID>	Container ID.
--resetpwdb	Removes Container's user database and creates a clean database as for any new installation.
--skipbackup	Does not save the old private area contents to the `/old` directory.
--scripts *<script>* `[...]`	Specifies the scripts to be executed during reinstallation. These scripts are used to customize application templates in the new Container and bring them to the same state as in the old Container. By default, all available scripts are executed.
--listscripts	Lists the scripts to be executed during Container reinstallation.
--desc	Displays the description of the scripts to be executed during Container reinstallation. Used together with the `--listscripts` option.

Notes:

1. 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.

2. Currently, the `reinstall` command may not be supported by the `prlctl` utility. Use `vzctl` instead.

prlctl set

This command is used for setting Container parameters.

Syntax

prlctl set *<CT_ID> <option> <value>*

where *<CT_ID>* is Container ID.

The command options specified in this file can be subdivided into the following categories:

- miscellaneous
- networking
- resource management
- hard disk drive management

Note: The current version of Virtuozzo does not support using some options with the `prlctl set` command (e.g., `--ext_ipadd` or `--disabled`). If you run `prlctl set` with such an option, the command will notify you that the option is not supported. In this case, you can execute the `vzctl set` command instead.

General Options

The table below lists the general options you can use with `prlctl set`.

Name	Description
--onboot <yes|no>	Set to “yes” to have Virtuozzo 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 Panel 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.
--crypted	Used with `--userpasswd`. Indicates that the specified password is already a hash.
--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. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--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. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--netdev_add <*name*>	Moves the specified network device from the server to the Container. For example: `--netdev_add eth0`. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--netdev_del <*name*>	Moves the specified network device from the given Container to the server. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--capability <*name*:**on** | **off**>	Specifies capabilities inside the Container. Setting the following capabilities is allowed: `AUDIT_WRITE`, `CHOWN`, `DAC_OVERRIDE`, `DAC_READ_SEARCH`, `FOWNER`, `FS_MASK`, `FSETID`, `IPC_LOCK`, `IPC_OWNER`, `KILL`, `LEASE`, `LINUX_IMMUTABLE`, `MKNOD`, `NET_ADMIN`,

	`NET_BIND_SERVICE`, `NET_BROADCAST`, `NET_RAW`, `SETGID`, `SETFCAP`, `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`, `VE_ADMIN`. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--features <*name*>**:on | off>**	Enables/disables the support for the following functionality inside the Container: • `nfs`: mounting NFS shares • `ipip`: creating IPIP tunnels • `sit`: using the Simple Internet Transition (SIT) mechanisms • `ppp`: using the PPP protocol • `ipgre`: creating IP-GRE tunnels • `bridge`: using bridges to connect virtual Ethernet devices • `nfsd`: running an NFS-kernel-space server **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--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. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--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. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--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. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--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 `prlctl set`. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--name <*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. Names must contain the `A-Z`, `a-z`, `0-9`, `\`, `-`, `_` symbols only. Names with white spaces must be enclosed in quotation marks.
--description <*desc*>	Custom Container description. Descriptions must contain the A -z, 0-9 symbols only. Descriptions with white spaces must be enclosed in quotation marks.
--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. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--bindmount_del <*dst*|**all**>	Removes the mount point created by using the `--bindmount_add` option from the Container. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--vnc-mode <auto|manual|off>	Enables or disables access to the Container via the VNC protocol.
--vnc-port <*port*>	Sets the VNC port number for the Container. Used with `--vnc-mode manual`.
--vnc-passwd <*passwd*> | **--vnc-nopasswd**	Sets the VNC password for the Container or specifies that no password is needed for VNC connections. Either of these options is mandatory for any VNC setup.
--autocompact <on|off>	Enables or disables compaction for all disks in the Container. **Note:** For details on how to enable or disable compaction for a specific disk in the Container, see

	Hard Disk Drive Management Options (p. 94).

Resource Management Options

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

Name	Description
--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).
-p, --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 Virtuozzo, the limit shall be set to the same value as the barrier. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--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 Virtuozzo, the limit shall be set to the same value as the barrier. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--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 Virtuozzo, the limit shall be set to the same value as the barrier. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
-e, --numiptent <*bar*:*lim*>	Number of IP packet filtering entries. **Note:** Currently, this option may not be supported by

	the `prlctl` utility. Use `vzctl` instead.
--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 Virtuozzo, the limit shall be set to the same value as the barrier. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
-k, --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. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--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. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
-b, --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. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--othersockbuf <*bar*:*lim*>	Total size in bytes of UNIX-domain socket buffers, UDP and other datagram protocol send buffers. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--dgramrcvbuf <*bar*:*lim*>	Total size in bytes of receive buffers of UDP and other datagram protocols. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--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 Virtuozzo, the limit shall be set to the same value as the barrier. **Note:** Currently, this option may not be supported by

	the `prlctl` utility. Use `vzctl` instead.
-l, --lockedpages <*bar*:*lim*>	Memory not allowed to be swapped out (locked with the `mlock()` system call), in 4-KB pages. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--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. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--physpages <*bar*:*lim*>	The total size of RAM used by processes, in 4-KB pages. 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. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--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. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
-n, --numfile <*bar*:*lim*>	Number of files opened by all Container processes. In this version of Virtuozzo, the limit shall be set to the same value as the barrier. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
-f, --numflock <*bar*:*lim*>	Number of file locks created by all Container processes. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
-t, --numpty <*bar*:*lim*>	Number of pseudo-terminals. For example, `ssh` session, `screen`, `xterm` application consumes pseudo-terminal resource. In this version of Virtuozzo, the limit shall be set to the same value as the barrier. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.

-i, --numsiginfo <*bar*:*lim*>	Number of `siginfo` structures (essentially, this parameter limits size of signal delivery queue). In this version of Virtuozzo, the limit shall be set to the same value as the barrier. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
-x, --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. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--cpuunits <*units*>	CPU weight. This is a positive integer number that defines how much CPU time the Container can get as compared to the other virtual machines and Containers running on the server. The larger the number, the more CPU time the Container can receive. Possible values range from 8 to 500000. If this parameter is not set, the default value of 1000 is used.
--cpulimit <*percent*|*megahertz*>	CPU limit, in per cent or megahertz (MHz), the Container is not allowed to exceed. This parameter is not set for newly created Containers; so they can consume all free CPU power of the server. By default, the limit is set in percent. To set the limit in MHz, specify "m" after the value. When setting this parameter in per cent, keep in mind that one CPU core makes up 100%. So if the server has 4 CPU cores, the total CPU power will equal 400%.
--cpus <*num*>	Number of CPU cores defining the CPU limit for a Container. The limit is calculated by multiplying the power of one CPU core by the number of the specified CPU cores. This option also defines the number of CPUs shown to Container users. This parameter is not set for newly created Containers; so they can consume all free CPU power of the server.
--cpumask <*num*>	CPU affinity mask. This mask defines the CPUs on the server that can be used to handle the processes running in the Container. The CPU mask can be specified as both separate CPU index numbers (1,2,3) and CPU ranges (2-4,5-7).
--nodemask <*num*|**all**>	The NUMA node mask defining a NUMA node to bind the Container to. Once you set the mask, the processes running in the Container will be executed only on the CPUs that belong to the specified NUMA node. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.

--diskspace <*bar*:*lim*>	Total disk space consumed by the Container, in 1-KB blocks. In the current version of Virtuozzo, both values must be equal.
--quotaugidlimit <0 | *N*>	Enables (if set to a value other than `0`) or disables (if set to `0`) per-user/group quotas for further management with the standard Linux `quota` utility. 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`.
--iolimit <*num*>	The bandwidth a Container is allowed to use for its disk input and output (I/O) operations. By default, the limit is set in megabytes per second. However, you can use the following suffixes to use other measurement units: • `G`: sets the limit in gigabytes per second. • `K`: sets the limit in kilobytes per second. • `B`: sets the limit in bytes per second. In the current version of Virtuozzo, the maximum I/O bandwidth limit you can set for a Container is 2 GB per second. The default I/O bandwidth limit for all newly created Containers is set to 0, which means that no limits are applied to any Containers.
--iopslimit <*num*>	The maximum number of disk input and output operations per second a Container is allowed to perform. By default, any newly created Container does not have the IOPS limit set and can perform so many disk I/O operations per second as necessary.
--rate <*class*:*Kbits*>	If traffic shaping is turned on, then this parameter specifies bandwidth guarantee for the Container. The format is *`class`*:*`Kbits`* where *`class`* is the network class (group of IP addresses) and *`Kbits`* is the traffic bandwidth.
--ratebound <yes | no>	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.
--reset_ub	Resets the current values of all system parameters of the server to the ones set in the `0.conf` file.

	Note: Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--memsize	The amount of RAM that can be used by the processes of a Container, in megabytes. You can use the following suffixes to set the RAM in other measurement units: • `G` in gigabytes • `K` in kilobytes • `B` in bytes
--swappages	The amount of swap space that can be used by the Container for swapping out memory once the RAM is exceeded, in 4-KB pages. You can use the following suffixes to set swap in other measurement units: • `G` in gigabytes • `M` megabytes • `K` in kilobytes • `B` in bytes
--swap	The amount of swap space that can be used by the Container for swapping out memory once the RAM is exceeded, in bytes. You can use the following suffixes to set swap in other measurement units: • `G` in gigabytes • `M` in megabytes • `K` in kilobytes
--vm_overcommit	Memory overcommit factor that defines the memory allocation limit for a Container. The limit is calculated as `(PHYSPAGES + SWAP) * factor` **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.

Network Options

Network-related options 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, and other parameters.

Name	Description
--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.
--ipadd <*addr/net_mask*>	Assigns the IP address and network mask to the Container. **Note:** You can assign network masks to Containers operating in the `venet0` networking mode only if the `USE_VENET_MASK` parameter in the Virtuozzo Containers configuration file is set to `yes`.
--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.
--ext_ipadd <*addr*>	Assigns the external IP address to the Container. External IP addresses are considered valid IP addresses by the `venet0` adapter, though they are not set as alias addresses inside Containers and are not announced via Address Resolution Protocol (ARP). You can assign the same external IP address to several Containers, irrespective of whether they reside on the same or different Hardware Nodes. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--ext_ipdel <*addr* | **all**>	Removes the external IP address from the Container. To delete all external IP addresses assigned to the Container, specify `--ext_ipdel all`. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--nameserver <*addr*>	The DNS server IP address for the Container. 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.
--netfilter <**disabled** | **stateless** | **stateful** | **full**>	Indicates which `iptables` modules are allowed for the Container. If some of the allowed modules are not loaded on the destination Hardware Node after migration or restoration from backup, they will be automatically loaded on the migrated or restored Container start. The following modes are available: • **disabled**: none. • **stateless**: (default) all modules except `conntrack` and NAT-related. • **stateful**: all modules except NAT-related.

	• **full**: all modules. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--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 Virtuozzo 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 <*addr*>	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 <*addr*>	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. **Note:** Currently, this option may not be supported by the `prlctl` utility. Use `vzctl` instead.
--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.
--apply-iponly <yes|no>	If set to `yes`, the hostname, nameserver, and search domain settings from the Container configuration file are ignored.

Hard Disk Drive Management Options

This group of options is used to manage virtual hard disks in a Container.

Syntax

```
prlctl set <CT_ID> --device-add hdd [--image <file>] [--size <size>] [--mnt <path>]
          [--iface <sata|ide|scsi>] [--position <pos>] [--subtype <buslogic|lsi-spi|lsi-sas>]
prlctl set <CT_ID> --device-set hdd<N> [--image <file>] [--size <size>] [--mnt <path>]
          [--iface <sata|ide|scsi>] [--position <pos>] [--subtype <buslogic|lsi-spi|lsi-sas>]
          [--autocompact <on|off>]
prlctl set <CT_ID> --backup-add <backup_ID> [--disk <disk_name>]
prlctl set <CT_ID> --device-del hdd<N> [--detach-only|--destroy-image]
prlctl set <CT_ID> --backup-del <backup_ID>|all
```

Options

Name	Description
<*CT_ID*>	Container ID.
--device-add hdd	Adds a virtual hard disk to the Container. If no other options are specified, the command creates a new unmounted disk with the following parameters: • name: `hdd`<*N*> where <*N*> is the next available disk index. • size: 65536 MB • image location: `/vz/private/`<*CT_ID*>`/disk-`<*ID*>`.hdd`. **Note:** In the current version of Virtuozzo, only expanding SATA disks can be added to Containers.
--device-set hdd<*N*>	Modifies the parameters of the virtual hard disk `hdd`<*N*>. **Note:** For the list of disks, use the `prlctl list -i` command.
--image <*file*>	Specifies an image file that will be used to emulate the virtual disk. • If the specified image does not exist, it is created and used to emulate the virtual hard disk. • If the specified image exists, it is used to emulate the virtual hard disk.
--size <*size*>	Specifies the size of the virtual hard disk, in megabytes.

--mnt *<path>*	Specifies the mount point of the virtual hard disk inside the Container. A corresponding entry is also added to Container's `/etc/fstab` file, so the disk is mounted automatically on Container start.
--autocompact <**on**|**off**>	Enables or disables compaction for the specified disk in the Container. **Note:** For details on how to enable or disable compaction for a specific disk in the Container, see **General Options** (p. 82).
--backup-add *<backup_ID>*	Attach the backup with the identifier *<backup_ID>* to the virtual machine as a virtual hard disk. To obtain the backup ID, use the `prlctl backup-list -f` command (p. 107).
--disk *<disk_name>*	Used with `--backup-add`. The name of the disk in the backup to attach. If a disk is not specified, all disks contained in the backup will be attached. To obtain the disk name, use the `prlctl backup-list -f` command (p. 107).
--device-del hdd*<N>*	Deletes a virtual hard disk from the stopped Container.
--detach-only	Removes the virtual disk from the Container configuration but leaves its image file intact.
--destroy-image	Removes the virtual disk from the Container configuration and deletes its image file.
--backup-del *<backup_ID>*|**all**	Detach either the backup with the identifier *<backup_ID>* or detach all backups from the virtual machine.

prlctl snapshot, snapshot-list, snapshot-switch, snapshot-delete

Takes, displays, reverts to, and deletes Container snapshots.

Syntax

```
prlctl snapshot <CT_ID> [-n,--name <name>] [-d,--description <desc>]
prlctl snapshot-list <CT_ID> [-t,--tree] [-i,--id <snapshot_ID>]
prlctl snapshot-switch <CT_ID> -i,--id <snapshot_ID> --skip-resume
prlctl snapshot-delete <CT_ID> -i,--id <snapshot_ID>
```

Options

Name	Description
<CT_ID>	Container ID.
-n, --name *<name>*	User-defined snapshot name. Names with white spaces must be enclosed in quotation marks.
-d, --description *<desc>*	User-defined snapshot description. Descriptions with white spaces must be enclosed in quotation marks.
-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*>	• Use with `prlctl snapshot-list` to specify the ID of the snapshot to use as the root. If this parameter is omitted, the entire snapshot tree will be displayed. • Use with `prlctl snapshot-switch` to specify the ID of the snapshot to revert to. • Use with `prlctl snapshot-delete` to specify the ID of the snapshot to delete.
--skip-resume	Skips automatic Container resume when switching to snapshots of running Containers.

Note: If the snapshot you want to delete has child snapshots derived from it, they will not be deleted.

prlctl start, stop, restart, status

These commands start, stop, restart, and show the current state of Containers, respectively.

Syntax

prlctl start <*CT_ID|CT_name*> [**--wait**]
prlctl stop <*CT_ID|CT_name*> [**--fast**]
prlctl restart <*CT_ID|CT_name*>
prlctl status <*CT_ID|CT_name*>

Options

Name	Description
<*CT_ID*|CT*_name*>	Container ID or name.

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 `prlctl 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, `prlctl` 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

Scripts (in the order of execution):

Name	Description
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 `prlctl` 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 `prlctl` 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 `prlctl` 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.

`prlctl 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, `prlctl` 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 `prlctl stop` is described below in the order of execution:

1. **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 `prlctl` does not stop the Container.
2. **umount**. Optional Container unmount script. If it exists, then it is executed after stopping the Container but before dismounting its private area.
3. **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 Virtuozzo versions.

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

The `prlctl 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.

prlctl statistics

Obtains Virtuozzo statistics for a specified Container.

Syntax

prlctl statistics *<CT_ID|CT_name>* [**--loop**] [**--filter** *<pattern>*]

Options

Name	Description
--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 <*pattern*>	Filters output by a specified pattern that supports wildcards (asterisk).

Guest Disk Usage Performance Counters

Name	Description
guest.fs<*N*>**.name**	Device name in a guest file system.
guest.fs<*N*>**.total**	Total size of a file system, in kilobytes.
guest.fs<*N*>**.free**	The amount of free space in a file system.
guest.fs<*N*>**.disk.**<*N*>	Disk indices.

Note: Guest counter values are obtained from guest virtual environments. Guest VEs cannot be considered a trusted source because counter values could be altered by a malicious user from within the VE.

prlctl suspend, resume

The `prlctl suspend` command is used to save the state of a running Container.

Syntax

prlctl suspend <*CT_ID*>

Options

Name	Description
<*CT_ID*>	Container ID.

During the `prlctl 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 Virtuozzo global file).

The `prlctl resume` command is used to restore the Container from its dump file created with the `prlctl suspend` command.

Syntax

prlctl resume <*CT_ID*>

When executed, `prlctl 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.

prlctl list

Displays a list of Containers on the Hardware Node. Displays information on Containers on the Hardware Node.

Syntax

prlctl list --vmtype ct [**-a**, **--all**] [**-o**, **--output** <*field*>[,...]] [**-s**, **--sort** <*field*|*-field*>]
[**-t**, **--template**] [**-j**, **--json**]
prlctl list -i, **--info --vmtype ct** [*CT_ID*|*CT_name*] [**-f**, **--full**] [**-t**, **--template**] [**-j**, **--json**]

Options

Name	Description
-a, --all	List all running, stopped, suspended, and paused Containers. If this and the rest of the parameters are omitted, only the running Containers will be displayed.
-t, --template	List available Container templates instead of actual Containers.
-o, --output <*field*>[,...]	Display only the specified fields. For the list of fields, see **prlctl list Output Parameters** (p. 99). Type field names in lower case. Separate multiple fields with commas.
-s, --sort <*field*|*-field*>	Sort Containers by the specified field in either ascending or descending order.
-i, --info	Display detailed information about the specified Container.
-f, --full	Display detailed information about network cards in Containers. Used with the `--info` option.
<*CT_ID*|*CT_name*>	The ID or name of the Container for which to display the detailed information. If not specified, the information will be displayed for all registered Containers.
-j, -json	Produce machine-readable output in the JSON format.

prlctl list Output Parameters

Listed below are the parameters that can be specified after the `-o` switch.

Name	Output Column	Description
uuid	`UUID`	Container ID.

hostname	HOSTNAME	Container hostname.
name	NAME	Container name.
description	DESCRIPTION	Container description.
ostemplate	OSTEMPLATE	Specifies the name of the OS template your Container is based on (e.g., `redhat-el5-x86`).
ip	IP_ADDR	Container IP address.
status	STATUS	Container status (e.g., running or stopped).
numproc	NPROC	The number of processes and threads allowed.
mac	MAC	Network device's MAC address.
netif	NETIF	Network devices in the Container.
iolimit	IOLIMIT	The bandwidth a Container is allowed to use for its disk input and output (I/O) operation, in bytes per second.
ha_enable	HA_ENABLE	Indicates whether the Container is joined to the High Availability Cluster.
ha_prio	HA_PRIO	Container priority in the High Availability Cluster (0 is the lowest). Higher-priority virtual environments are restarted first in case of failures.

Migration Utilities

This section describes the utilities you can use to migrate Containers between physical servers or move/clone Containers within the same server.

pmigrate

The `pmigrate` utility is used to perform different kinds of Container migration.

Syntax

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

<*source_server*> can be either the physical server hosting the virtual machine or Container to move or the physical computer to migrate. <*destination_server*> is the server where the resulting Container is to be hosted. If the source and/or destination server is not specified, the operation is performed on the local server.

<*source_server*> and <*destination_server*> consist 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]@]<dst>[:port]
```

where <*dst*> is the destination server IP address or hostname.

The [*options*] you can use with `pmigrate` depend on whether you are migrating to a virtual machine or a Container. This section describes the parameters for migrating Containers between hosts and for moving virtual machines and physical servers to Containers. For information on performing these operations for virtual machines, see **pmigrate** (p. 173).

Container-related Options

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

Name	Description
-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

	Virtuozzo kernel on the destination server (for the description of the `TECHNOLOGIES` parameter, see **Container Configuration File**). • `license`: do not check the license installed on the destination server. • `rate`: do not check the value of the `RATE` parameter in the Virtuozzo global file.
-b, --batch	Normally, you do not have to specify this option. It is used by Virtuozzo 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: 1. The main amount of Container memory is transferred to the destination server. 2. The Container is 'dumped' and saved to an image file. 3. 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.
--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, 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 using the 'simple online migration' type.
--readonly	Just copy the specified Container to the destination server without making any changes to the Container on the Source Server.
--convert-vzfs	Convert legacy Containers to the Container-in-an-image-file layout.
--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.
--non-sharedfs	Use to migrate Containers from a shared file system to a non-shared one.

Virtual Machine-related Options

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 host 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 per-user/group 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 host 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 host. 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 host. 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 host. 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 host) before the migration. This will prevent the running services from modifying any files being moved.

prlctl clone

Creates an exact copy of the specified Container.

Syntax

```
prlctl clone <CT_ID|CT_name> --name <new_name> [--template] [--dst <path>]
```

Options

Name	Description
<CT_ID>	ID of the Container to clone.
--name *<new_name>*	Name or ID to be assigned to the new Container.
--template	Create a Container template instead of a clone. You cannot start a template. **Note:** Container templates do not have any practical use in Virtuozzo 6 and are mainly used in Odin Automation for Cloud Infrastructure.
--dst *<path>*	Full path to the directory for storing the contents of the cloned Container. If this parameter is omitted, the clone is created in the default directory.

Backup and Restoration Utilities

Any Container is defined by its private area, configuration files, action scripts, and quota information. Backing up these components allows you to restore all the content of a Container on any Virtuozzo-based system at any time if the Container gets broken.

Note: For the `vzabackup`/`vzarestore` functionality to work, forward and reverse DNS lookups must be correctly configured for both the source and backup servers.

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 (the default directory is `/var/parallels/backups`). Later on, the Container backups may be restored from this directory.

Syntax

```
pbackup [backup_options] server1 ... [CT_options]
```

You may specify any number of servers in the command-line by names or IP addresses. 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.

Notes:

1. This section describes backup options for Containers. For backup options that can be used with virtual machines, see **pbackup** (p. 176).

2. 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.

Backup Options

Name	Description
-F, -I	Create a full backup. A full backup contains 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.
--ssh-opts <*options*>	Options to be passed to `ssh`. See examples in the global backup configuration file.

These options define the list of Containers to back up:

Name	Description
-e <*CT_ID*|*CT_name*> [...]	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 <*CT_ID*|*CT_name*> [...]	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.

Container-specific Backup Options

These options must be used with the `vzctl` command instead of `prlctl`.

Name	Description
-Cg **-Cb** **-Cn**	Compress the resulting Container backups with `gzip` (`-Cg`), `bzip2` (`-Cb`) or do not use compression (`-Cn`). This option overrides 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.
-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 unattended mode as a cron job. It overrides the `CRON_BACKUP` parameter in the global backup configuration file.
--desc <*desc*>	The description of the backup archive. Descriptions with white spaces must be enclosed in quotation marks.
-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).
--vzcache	Back up not the Containers themselves, but the cache area of the server (`/vz/template/vzcaches`).

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).

Syntax

prestore [*restoration_options*] *server* [...] [*CT_options*]

You can specify multiple servers (by names or IP addresses) whose Containers were at one time backed up and now need to be restored.

Restoration Options

Name	Description
-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 `/var/parallels/backups/`*`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:

Name	Description
-e <*CT_ID*|*CT_name*> `[...]`	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 <*CT_ID*|*CT_name*> `[...]`	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.

prlctl backup, backup-list, backup-delete, restore

Creates, lists, deletes or restores Container backups.

Syntax

```
prlctl backup <CT_ID> [-f,--full] [-i,--incremental] [--description <desc>]
                      [-s,--storage [user[:passwd]@]server[:port]
prlctl backup-list [CT_ID] [-f,--full] [--localvms]
                           [-s,--storage [user[:passwd]@]server[:port]]
```

```
prlctl backup-delete {<CT_ID> | -t,--tag <backup_ID>}
                     [-s,--storage [user[:passwd]@]server[:port]]
prlctl restore {<CT_ID> | -t,--tag <backup_ID>} [-n, --name <new_name>] [--dst <path>]
               [-s,--storage [user[:passwd]@]server[:port]]
```

Options

Name	Description
<CT_ID>	Container ID. • Use with `prlctl backup` to create a backup of the specified Container. • Use with `prlctl backup-list` to list backups of the specified Container. • Use with `prlctl backup-delete` to delete all backups of the specified Container. • Use with `prlctl restore` to restore the most recent backup of the specified Container.
-s, --storage [*user*[:*passwd*]@]*server*[:*port*]]	Specifies a remote backup server address, port, and credentials. If this option is omitted, the backup will be saved on the default backup server that can be configured using the **prlsrvctl set** command.
--description <*desc*>	Backup description. Descriptions with white spaces must be enclosed in quotation marks.
-f, --full	• Use with `prlctl backup` to create a full backup of the Container. A full backup contains all Container data. • Use with `prlctl backup-list` to display full backup information.
-i, --incremental	Create an incremental backup of the Container. An incremental backup contains only the files changed since the previous full or incremental backup. This is the default backup type.
--localvms	List local backups only.
-t, --tag <*backup_ID*>	The ID of the backup to restore or delete.
-n, --name <*new_name*>	A new name to assign to the restored Container. If this option is omitted, the Container will be restored with the original name.
--dst <*path*>	Restore the Container to the specified directory on the host. If this option is omitted, the Container will be restored to `/var/parallels/`<*vm_name*>.

EZ Template Management Utilities

This section described the utilities you can use to manage OS and application templates.

vzpkg

The `vzpkg` utility is used 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., `mysql.rpm`) inside Containers.

Syntax

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

Commands

Name	Description
install template	Installs OS and application EZ templates on the server.
update template	Updates OS and application EZ templates installed on the server.
remove template	Removes OS and application EZ templates from the server.
list	Outputs a list of EZ templates, OS template caches with preinstalled application templates, or software packages either on the server or inside a particular Container.
info	Outputs information on any EZ templates or software packages available on the server or inside the Container.
status	Outputs information on updates for the packages installed inside a Container.
install	Adds application EZ templates to or to install software packages inside the Container.
update	Updates application EZ templates and software packages inside the Container.
remove	Removes application EZ templates or software packages from the Container.
create cache	Creates a tarball (cache) for the given OS EZ template.
update cache	Updates the existing tarball (cache) for the given OS EZ template.
remove cache	Removes a tarball (cache) for the given OS EZ template.
create appcache	Creates a cache of an OS EZ template with preinstalled application templates.
update appcache	Updates or recreates a cache of an OS EZ template with preinstalled application templates.
remove appcache	Removes a cache of an OS EZ template with preinstalled application templates.
localinstall	Installs a software package inside a Container from the corresponding file on the server.
localupdate	Updates the software packages installed inside your Container(s) by means of the `vzpkg install` or `vzpkg localinstall` commands.
upgrade	Upgrades an OS EZ template the Container is based on to a newer version.

fetch	Downloads packages included in EZ templates to the server and to store them in the `vzpkg` local cache.
clean	Removes all locally cached data from the template directories on the server.
update metadata	Updates 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 an RPM package or official repositories.

Syntax

vzpkg install template [*options*] <*object*> [...]

where <*object*> is a path to an RPM package or an EZ template name.

Options

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

Note: To install multiple EZ templates, specify multiple RPM package or EZ template names separated by white spaces.

vzpkg update template

This command is used to update an OS or application EZ template on the server from an RPM package or official repositories.

Syntax

vzpkg update template [*options*] <*object*> [...]

where <*object*> is a path to an RPM package or an EZ template name.

Options

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

Note: To update multiple EZ templates, specify multiple RPM package or EZ template names separated by white spaces.

vzpkg remove template

This command removes an OS or application EZ template from the server.

Syntax

vzpkg remove template [*options*] <*template_name*> [...]

Options

Name	Description
-F, --for-os <*OS_template*>	Specifies the OS EZ template to delete the application EZ template from.
-q, --quiet	Disables logging to screen and file.
-f, --force	Forces deletion of the EZ template.

When executed, the `vzpkg remove template` command removes the specified OS EZ template from the server. To delete an application EZ template, 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

- EZ templates installed on the server, in a Container, or available in remote EZ template repositories
- YUM software groups or individual packages installed in a Container

Syntax

vzpkg list [*options*] [*OS_template*|*CT_ID*|*CT_name* [...]]

If you indicate a Container ID or name, the command will list all EZ templates applied to the specified Container. If you indicate an OS EZ template, `vzpkg list` will display a list of application EZ templates available for this OS EZ template. Without any options, the utility lists all EZ templates installed on the server.

Options

Name	Description
-p, --package	Lists the software packages installed in the Container or included in the OS EZ template.
-g, --groups	Lists the YUM software groups installed in the Container or available for the OS EZ template. The `-g` option works only for Containers running RPM-based Linux distributions.
-O, --os	Displays the OS EZ template the Container is based on.

-A, --app	Displays the application EZ templates installed in the Container or included in the OS EZ template.
-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.
-S, --with-summary	In addition to listing the EZ templates available either in 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.
appcache	Outputs a list of OS EZ template caches with preinstalled applications.
-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 displays information about EZ templates, YUM software packages, and individual software packages.

Syntax

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

Options

Name	Description
CT_ID|CT_name	Container ID or name.
OS_template	OS EZ template.
<*app_template*>	Application EZ template.
<*package_name*>	Software package name.
<*yum_package_group*>	YUM software group name.
-F, --for-os <*OS_template|CT_ID|CT_name*>	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 Container.
-p, --package	Displays information about the specified software package. Must be used with the `-F` option.
-g, --groups	Displays information about the packages included in the specified YUM software group.
-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.

Output Parameters

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`); • 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).
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 for managing the packages included in the specified EZ template. It can be one of the following: **x86 Linux distributions** • `rpm49db5x86`: Fedora 17 • `rpm49x86`: Fedora 15 and 16 • `rpm47x86`: Red Hat Enterprise Linux 6 and CentOS 6 • `rpm44x86`: Red Hat Enterprise Linux 5 and CentOS 5 • `rpm43x86`: Red Hat Enterprise Linux 3 and 4, CentOS 3 and 4

	• `rpmzypp44x86`: SUSE Linux Enterprise Server 11 with Service Pack 2 • `rpm41x86`: SUSE Linux Enterprise Server 10 and SUSE Linux 10.*x* • `rpm41s9x86`: SUSE Linux Enterprise Server 9 • `rpmzypp49x86`: openSUSE 12.1 • `dpkg`: Debian and Ubuntu **x86-64 Linux distributions** • `rpm49db5x64`: Fedora 17 • `rpm49x64`: Fedora 15 and 16 • `rpm47x64`: Red Hat Enterprise Linux 6 and CentOS 6 • `rpm44x64`: Red Hat Enterprise Linux 5 and CentOS 5 • `rpm43x64`: Red Hat Enterprise Linux 3 and 4, CentOS 3 and 4 • `rpmzypp44x64`: SUSE Linux Enterprise Server 11 with Service Pack 2 • `rpm41x64`: SUSE Linux Enterprise Server 10 and SUSE Linux 10.*x* • `rpm41s9x64`: SUSE Linux Enterprise Server 9 • `rpmzypp49x64`: openSUSE 12.1 • `dpkgx64`: Debian and Ubuntu

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.

Syntax

vzpkg status [*options*] <*CT_ID*|*CT_name*|*OS_template*>

Options

Name	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 install application EZ templates, YUM software groups, or individual software packages into Containers.

Syntax

vzpkg install [*options*] <*CT_ID*|*CT_name*> <*object*> [...]

The `vzpkg install` command will add an <*object*> to the specified Container. An object can be an application EZ template, a YUM software group, or a standard software package. You can specify several objects to install into the Container by separating them by spaces.

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

Options

Name	Description
-p, --package	Installs a software package instead of an EZ template.
-g, --groups	Installs a YUM software group instead of an EZ template. The `-g` option works only for Containers running RPM-based Linux distributions.
-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` or `-g` option to explicitly specify the type of the object.

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 the following components of a Container:

- OS EZ template
- application EZ templates
- YUM software groups
- individual software packages

Syntax

vzpkg update [*options*] <*CT_ID*|*CT_name*> [*object* [...]]

Options

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 in the Container by using the `vzpkg install` command.

-g, --groups	Updates the YUM software group in the Container. The `-g` option works only for Containers running RPM-based Linux distributions.
-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 options specified, `vzpkg update` updates all EZ templates (including the OS EZ template) in the specified Container. However, you can make the command update a particular EZ template by specifying its name as *`object`*. You can also use the `-p` or `-g` option to update YUM software groups or individual software packages in the Container.

vzpkg remove

This command is used to remove an application EZ template, YUM software group, or a software package from a Container.

Syntax

vzpkg remove [*options*] <*CT_ID*|*CT_name*> <*object*> [...]

This command will remove <*object*> from the Container with the ID of <*CT_ID*> or with the name of <*CT_name*>. The <*object*> can be an application EZ template, a YUM software group, or a software package installed with the `vzpkg install` command. You may specify a number of objects for removing.

Options

Name	Description
-p, --package	Removes the specified package(s) from the Container.
-g, --groups	Removes the specified YUM software group from the Container. The `-g` option works only for Containers running RPM-based Linux distributions.
-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` or `-g` option to explicitly specify the type of the object.

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

vzpkg create cache

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

Syntax

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

Options

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, `prlctl` 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.

Syntax

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

Options

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.

Syntax

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

Options

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 create appcache

This command combines an OS EZ template cache and one or more application EZ templates into a new OS and applications cache. If the OS EZ template cache has not been created yet, this will be done before application templates are added.

Syntax

vzpkg create appcache --config <*config*> [*options*]

Options

Name	Description
--config <*config*>	Specifies the path to the configuration file with the information on what OS and application templates to use.
--ostemplate <*OS_template*>	Specifies the OS EZ template to use in cache creation. This option redefines the OS EZ template specified in the configuration file.
--apptemplate <*app_template*>	Specifies one or more application EZ templates (comma-separated) to be added to the resulting cache.This option redefines the application EZ templates specified in the configuration file.
-d, --debug <*num*>	Sets the debugging level (0 to 10), 10 being the highest.
-q, --quiet	Disables logging to screen and log file.
-f, --force	Forces cache creation.

vzpkg update appcache

This command updates an existing OS EZ template cache with preinstalled application templates if the `--update-cache` option is provided. Otherwise, the cache is created anew.

Syntax

vzpkg update appcache --config <*config*> [*options*]

Options

Name	Description
--config <*config*>	Specifies the configuration file with the information on what OS and application templates to use.
--ostemplate <*OS_template*>	Specifies the OS EZ template, cache of which needs to be recreated or updated. This option redefines the OS EZ template specified in the configuration file.
--apptemplate <*app_template*>	Specifies all application EZ templates (comma-separated) preinstalled in the cache which needs to be updated. This option redefines the application EZ templates specified in the configuration file.
--update-cache	Instructs the command to check for updates for the existing OS and application cache. Otherwise, the cache is created anew.
-d, --debug <*num*>	Sets the debugging level (0 to 10), 10 being the highest.
-q, --quiet	Disables logging to screen and log file.
-f, --force	Forces cache creation.

vzpkg remove appcache

This command removes an existing OS EZ template cache with preinstalled application templates.

Syntax

vzpkg remove appcache --config <*config*> [*options*]

Options

Name	Description
--config <*config*>	Specifies the configuration file with the information on what OS and application templates to use.
--ostemplate <*OS_template*>	Specifies the OS EZ template, cache of which needs to be removed. This option redefines the OS EZ template specified in the configuration file.
--apptemplate <*app_template*>	Specifies all application EZ templates (comma-separated) preinstalled in the cache which needs to be removed. This option redefines the application EZ templates specified in the configuration file.
-d, --debug <*num*>	Sets the debugging level (0 to 10), 10 being the highest.
-q, --quiet	Disables logging to screen and log file.
-f, --force	Forces cache creation.

vzpkg localinstall

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

Syntax

vzpkg localinstall [*options*] <*CT_ID*|*CT_name*> <*rpm_file_path*> [...]

Options

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 multiple packages 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.

Syntax

vzpkg localupdate [*options*] <*CT_ID*|*CT_name*> <*rpm_file_path*> [...]

Options

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.

Syntax

vzpkg upgrade [*options*] <*CT_ID*|*CT_name*>

Options

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 upgrade of 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 installation.

Syntax

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

Options

Name	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).

Syntax

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

Options

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 behavior 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.

Syntax

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

Options

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.

Syntax

vzmktmpl [*options*] <*metafile*>

Options

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 only required parameter for `vzmktmpl` is the path to the metafile containing the template configuration. In most cases, however, you also need to include a number of scripts in the template. These scripts will be executed at different stages of the template life cycle (e.g., when you add the template to a Container) and ensure the correct template operation. To learn what scripts are required for your template:

1. Install an official template for the same Linux distribution for which you are creating the template. For example, if you are making a template for CentOS 5, install the CenOS 5 OS template (use the `vzupdate -z` command, if necessary).
2. Go to the configuration directory (`/vz/template/`*`TemplateName/TemplateArch`*`/config/os/default`) of the installed template.
3. Look for the files starting with "`pre-`" and "`post-`" (e.g, `pre-cache` or `post-install`).

Note: The `vzmktmpl` utility is a part of the `vztt-build` package, which is not installed by default. If installed, the package can be found in the `/virtuozzo/RPMS/optional` directory of your Virtuozzo distribution. So, before you can start using the `vzmktmpl` utility, you first need to install the package on your server with the `rpm -i` command.

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>
```

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).
%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. Should be omitted for application templates.
%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: **x86 Linux distributions** • `rpm49db5x86`: Fedora 17 • `rpm49x86`: Fedora 15 and 16 • `rpm47x86`: Red Hat Enterprise Linux 6 and CentOS 6 • `rpm44x86`: Red Hat Enterprise Linux 5 and CentOS 5 • `rpm43x86`: Red Hat Enterprise Linux 3 and 4, CentOS 3 and 4 • `rpmzypp44x86`: SUSE Linux Enterprise Server 11 with Service Pack 2 • `rpm41x86`: SUSE Linux Enterprise Server 10 and SUSE Linux 10.*x* • `rpm41s9x86`: SUSE Linux Enterprise Server 9 • `rpmzypp49x86`: openSUSE 12.1 • `dpkg`: Debian and Ubuntu **x86-64 Linux distributions** • `rpm49db5x64`: Fedora 17 • `rpm49x64`: Fedora 15 and 16 • `rpm47x64`: Red Hat Enterprise Linux 6 and CentOS 6 • `rpm44x64`: Red Hat Enterprise Linux 5 and CentOS 5 • `rpm43x64`: Red Hat Enterprise Linux 3 and 4, CentOS 3 and 4 • `rpmzypp44x64`: SUSE Linux Enterprise Server 11 with Service Pack 2 • `rpm41x64`: SUSE Linux Enterprise Server 10 and SUSE Linux 10.*x* • `rpm41s9x64`: SUSE Linux Enterprise Server 9

	• `rpmzypp49x64`: openSUSE 12.1 • `dpkgx64`: Debian and Ubuntu This parameter must be specified for all base OS EZ templates and should 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. Should be omitted for application templates.
%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 OS and application EZ templates. 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. Detailed information on how to set up caching proxy servers is given in the *Virtuozzo 6 Templates Management Guide*.

vzcreaterepo

The `vzcreaterepo` utility is used to set up repositories for RHEL-like distributions, including Red Hat Enterprise Linux 5.

Syntax

vzcreaterepo [*options*] [*mount_points*]

Options

Name	Description
mount_points	Path to the directory with Linux distribution files. If omitted, `vzcreaterepo` searches for files in the `/media/cdrom` directory.
-d, --dir <*repo_path*>	Path to the directory for setting up the repository. If omitted, the repository is created in the `/vz/template/repos` directory.
-t, --template <*template*>	Name of the template for which to create the repository (e.g., `redhat-el5-x86_64`).
-m, --mirror	Creates a Container and configures it to act as an HTTP mirror for servers in your network. Requires the `-v`, `-n`, `--nameserver`, and `--ip` parameters to be specified.
-q, --quiet	Disables logging to the screen.
-u, --update	Creates a repository with updates. At the moment, this parameter is valid for Red Hat Enterprise Linux 5 only.
-i, --iso <*path1*>[,*path2...*]	Path to the an ISO image containing Linux distribution files. You can specify several images and separate them by commas.
-a, --cache	Caches the template after setting up the repository.
-p, --proxy http://<*IP_or_hostname*>:<*port*>	Hostname or the IP address and the port number of the HTTP proxy server, if you use any to connect to the Internet.
--proxy_user <*proxy_user*>	User name used by the HTTP proxy server for your authentication.
--proxy_pass <*proxy_passwd*>	Password of the user specified in the `--proxy_user` parameter and used for your authentication by the HTTP proxy server.
-e, --email <*email*>	Email address for registering in the Red Hat Network. This is the email address you specified when creating an account in the Red Hat Network.
--systemname <*system_name*>	Name of your system to use for retrieving packages in the Red Hat Network. If omitted, the name `updates` is used.
-r, --repotemplate <*repo_template*>	OS template to use for creating the Container that will act as an HTTP mirror. If omitted, the `centos-5-x86_64` template is used.
-v, --ctid <*CT_ID*>	ID to assign to the Container.
-n, --hostname <*hostname*>	Hostname to assign to the Container.
--nameserver <*nameserver*>	Nameserver to set for the Container.
--ip <*IP_address*>	IP address to assign to the Container.

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

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.

Syntax

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

Options

Name	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.

Supplementary Tools

pcompact

Utility to compact Containers by removing unused blocks from their virtual disks. By compacting virtual disks, you can increase free disk space on the physical server.

Syntax

```
pcompact [-v] [-n] [-s] [-t <timeout>[s|m|h]
```

Options

Name	Description
-v	Increase the command output verbosity. Multiple `-v` options can be specified to produce a more verbose output.
-n	Display the actions the command will execute but do not actually compact the disks.
-s	Stop the command execution after compacting the first virtual disk.

-t <*timeout*>**[s|m|h]**	Terminate the command after the specified timeout, in seconds (default), minutes or hours.

pfcache

Memory and IOPS deduplication management utility that enables/disables caching for Container directories and files, verifies cache integrity, checks Containers for cache errors, and purges the cache if needed.

Note: The utility does not support additional disks attached to Container.

Syntax

pfcache <*command*> <*file|dir*> [*options*]

Commands

Name	Description
mark (p. 135)	Enables caching of the specified files, directories or subdirectories in Containers.
unmark	Disables caching of the specified files, directories or subdirectories in Containers.
purge (p. 136)	Frees up space in the memory and IOPS deduplication cache image.
verify (p. 137)	Verifies the integrity of the specified mounted memory and IOPS deduplication cache and deletes corrupted files.
check (p. 134)	Checks for and fixes memory and IOPS deduplication cache errors in the specified Container root directory.
stat (p. 136)	Displays inodes summary for a running Container.
dump (p. 135)	In addition to the summary provided by stat, shows detailed information on PFCache inodes of a running Container. If no options are specified, outputs full information on checksummed peer inodes.

pfcache check

Checks for and fixes memory and IOPS deduplication cache errors in the specified Container root directory.

Syntax

pfcache check <*dir*> [**--dry-run**]

Options

Name	Description
<*dir*>	Container root directory.

--dry-run	Report errors but do not make changes to the file system.

pfcache dump

In addition to the summary provided by stat, shows detailed information on PFCache inodes of a running Container. If no options are specified, outputs full information on checksummed peer inodes.

Syntax

```
pfcache dump <dir> [--csummed | {--all|-a} | <csum>] [--column <col>...]
```

Options

Name	Description
<dir>	Container root directory.
--csummed	Outputs information on checksummed inodes.
--all|**-a**	Outputs information on all inodes.
<csum>	Outputs information on inodes with the specified checksum.
--column *<col>...*	Displays only the chosen column(s). *<col>* can be: • a — All • h — File handle • p — Page cache size, pages • c — Checksum • f — Filter state • s — File size, bytes

pfcache mark, unmark

Enables or disables caching of the specified files, directories or subdirectories in Containers.

Syntax

```
pfcache mark <file>
pfcache mark <dir> [subdir ...] [--recursive]
pfcache unmark <file>
pfcache unmark <dir> [subdir ...] [--recursive]
```

Options

Name	Description
<file>	File to enable/disable caching of.
<dir>	Directory to enable/disable caching of.
subdir	Subdirectory to enable/disable caching of.
--recursive	Process directory's or subdirectory's current contents.

pfcache purge

Frees up space in the memory and IOPS deduplication cache image. If no options are specified, purges entire cache.

Syntax

pfcache purge *<cache_dir>* [**--unused** | **--size** *<size>* | **--expire** *<date>*]

Options

Name	Description
<cache_dir>	Memory and IOPS deduplication cache image location.
--unused	Remove only files unused at the moment.
--size *<size>*	Attempt to free *size* bytes in the memory and IOPS deduplication cache image.
--expire *<date>*	Remove files accessed before the specified date. A date can be specified in the ISO 8601 format or in the format defined in your system locale, with optional hours, minutes, and seconds. Examples: • 05/21/12 [09:50[:33]] • 2012-05-21 [09:50[:33]]

pfcache stat

Displays a summary of all files in the specified filesystem which have been accessed recently or are being accessed now.

Syntax

pfcache stat *<dir>*

Options

Name	Description
<dir>	Container root directory.

Displayed Information

Name	Description
csums	The number of checksummed files and their percentage in the total number of files fetched by the command. **Note:** Only checksummed files can be cached.
inodes	The number of files which have been accessed recently or are being accessed now.
size	The size of the files, in kilobytes.
RAM	Memory used by the files, in kilobytes.
fetched	The number of files fetched by the command.
uncached	The number of files only in Container's private area.
cached	The number of files which have copies in the PFCache area and their percentage in **fetched**.

pfcache verify

Verifies the integrity of the specified mounted memory and IOPS deduplication cache and deletes corrupt files.

Syntax

```
pfcache verify <cache_dir>
```

Options

Name	Description
<cache_dir>	Memory and IOPS deduplication cache image location.

pnetstat

This utility outputs traffic usage statistics for Containers.

Syntax

```
pnetstat [-v <CT_ID>] [-c <class>] [-a] [-r] [-h] [-t <ct|vm|all>]
```

The utility displays input and output traffic for Containers for each defined network class, in bytes. The network classes are described in the `/etc/vz/conf/networks_classes` file. `pnetstat` displays statistics only on Containers that were started at least once.

Options

Name	Description
-v <*ID*>	Display statistics for the specified Container. Multiple `-v` options can be given to a single `pnetstat` invocation.
-c <*class*>	Display statistics for the <*class*> class only.
-a	Display statistics for all classes.
-r	Rounds down the statistics results, which is shown in bytes by default. In this case, the statistics is displayed in the following units of measurement, depending on the amount of traffic: • `K(b)` — kilobytes • `M(b)` — megabytes • `G(b)` — gigabytes
-t <ct | vm | all>	Display statistics for • Containers only: `ct` • virtual machines only: `vm` • both Containers and virtual machines: `all`
-h, -help	Display the utility usage information.

prl_disk_tool

The `prl_disk_tool` utility is used to manage virtual hard disk drives.

prl_disk_tool compact

Removes all empty blocks from the expanding virtual disk to reduce its size on the physical hard disk. The virtual disk must be formatted to NTFS, FAT16, FAT32, ext2, or ext3. You can also try to compact virtual disks with other filesystems using the `--buildmap` option.

Syntax

```
prl_disk_tool compact [--buildmap] --hdd <disk_path> [--force]
prl_disk_tool compact -i,--info --hdd <disk_path>
```

Options

Name	Description
--hdd <*disk_path*>	Full path to the virtual disk.

--buildmap	Compacts 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.

prl_disk_tool merge

Merges all snapshots of the virtual hard disk.

Syntax

```
prl_disk_tool merge --hdd <disk_path>
```

Options

Name	Description
--hdd <*disk_path*>	Full path to the virtual disk.

prl_disk_tool resize

Changes the capacity of the specified virtual disk. During resizing, all data present on the disk volumes are left intact. You can also resize the last partition using the `--resize_partition` option. The supported file systems are NTFS, FAT16/FAT32, and ext2/ext3.

Syntax

```
prl_disk_tool resize --size <size>[M|G] [--resize_partition] --hdd <disk_path> [--force] [--split]
prl_disk_tool resize -i,--info --hdd <disk_path>
```

Options

Name	Description
--size	New size of the virtual disk. It can be set either in megabytes (`M`, default) or gigabytes (`G`).
--resize_partition	Resizes the last partition of the specified virtual disk. **Note:** You cannot reduce XFS filesystems on LVM (the default choice for CentOS 7 and Red Hat Enterprise Linux 7).
--hdd <*disk_path*>	Full path to the virtual disk.
--force	Forces the resizing operation for suspended virtual disks.
--split	Splits the virtual hard disk into 2 GB files.
-i, --info	Do not resize the virtual disk; just show the size the disk will have after resizing.

vzcalc

This utility is used to calculate Container resource usage.

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.

vzcfgscale

This utility is used to "scale" Container configuration. It multiplies Container resource control parameters by the number passed as an argument.

Syntax

vzcfgscale [*options*] *<CT_config_file>*

Container configuration file shall be always the last parameter and the utility uses it to produce scaled Container configuration.

Options

Name	Description
-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 *<factor>*.
-c *<factor>*	Multiply CPU parameters by *<factor>*.
-d *<factor>*	Multiply disk related parameters by *<factor>*.
-u *<factor>*	Multiply system resource control parameters by *<factor>*.
-n *<factor>*	Multiply bandwidth parameters by *<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 `prlctl 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 the resource management parameters consistency in a Container configuration file.

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, the utility prints a message with its severity status. Three severity statuses are defined in Virtuozzo:

Name	Description
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 the 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 Container behave in an unexpected way and there seems to be no resource shortage for the Container.

vzcheckovr

This utility is used to check the current system overcommitment and safety of the total resource control settings.

Syntax

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 Virtuozzo 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.

vzcpucheck

This utility displays the current server utilization in allocated CPU units as well as total server CPU unit capacity.

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.

vzctl convert

Command to convert Containers based on VZFS, the layout used in earlier versions of Parallels Server Bare Metal, to the Container-in-an-image-file layout used in Virtuozzo 6. The new layout has been redesigned to store all files of a Container in a single image, similar to a virtual machine's hard disk.

Syntax

vzctl convert *<CT_ID>* [**--velayout** *<layout_ID>*]

Options

Name	Description
<CT_ID>	ID of the Container to convert.
--velayout *<layout_ID>*	ID associated with the Container layout. To convert a Container to the Container-in-an-image-file layout, specify **5** after the option. If this option is omitted, the command converts a Container to the layout defined by the **VEFSTYPE** parameter in the global configuration file `/etc/vz/vz.conf`. For the Container-in-an-image layout, this parameter should be set to `ext4`.

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.

Syntax

```
vzhwcalc [options]
```

Options

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).

vziotop

Just like the Linux standard `iotop` utility, `vziotop` monitors I/O usage data output by the Linux kernel and displays the current I/O usage by processes or threads on the system.

In addition to the options of the standard `iotop` utility, the `vziotop` utility has one extra option: `-E <CT_ID>`. When used with this option, `vziotop` shows the current I/O usage by processes or threads running in the specified Container.

Another difference from `iotop` is that `vziotop` shows actual disk read/write alongside the standard total disk read/write. The latter values represent total read/write bandwidth between processes and kernel threads on one side and kernel block device subsystem on the other. In turn, actual disk read/write values represent corresponding bandwidths for actual disk I/O between kernel block device subsystem and underlying hardware (HDD, SSD, etc.). Thus, total and actual values may differ at any given moment due to data caching and I/O operations reordering in the Linux kernel.

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.

Syntax

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

Options

Name	Description
-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 do without any options.

vzpid

This utility prints the ID of the Container where the process is running.

Syntax

```
vzpid <pid> [...]
```

Multiple process IDs can be specified as arguments.

vzps, 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 their 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.

vzsplit

This utility is used to generate a sample Container configuration file with a set of system resource control parameters.

Syntax

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.

Options

Name	Description
-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 `prlctl create` command. If a sample with this name already exists, the utility will output an error message and will not overwrite the existing configuration.

CHAPTER 4

Managing Virtual Machines

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

In This Chapter

prlctl

Virtuozzo virtual machines can be managed using the `prlctl` command-line utility.

General Syntax

The `prlctl` 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 getting statistics and generating problem reports.

Syntax

```
prlctl <command> <vm_ID|vm_name> [options] [-v, --verbose <number>] [--timeout <sec>]
        [-l, --login [user[:passwd]@]server [-p, --read-passwd <file>]]
```

Options

Name	Description
<command>	The name of the command to execute.
<vm_ID|vm_name>	The ID or name of the virtual machine to perform the operation on. To obtain the list of the available virtual machines, use the **prlctl list** command.
options	Command options. See individual commands for available options.
-v, --verbose *<number>*	Enables verbose output. The greater the *<number>*, the higher is the verbosity.

-l, --login [*user*[:*passwd*]@]*server*]	Connect to a remote server *server* with the specified credentials. If this flag is omitted, the `prlctl` command is assumed to be run locally.
-p, --read-passwd <*file*>	Use the password from the file <*file*> to log in to a remote server, other credentials being specified with the `--login` option. The `--read-passwd` option can be specified multiple times in order to form a password stack for operations requiring multiple passwords. Each password must be supplied in a separate file.

Remarks

To display help, enter `prlctl` without any options.

prlctl 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

prlctl capture <*vm_ID*|*vm_name*> **--file** <*name*>

Options

Name	Description
<*vm_ID*|*vm_name*>	The virtual machine ID or 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.

prlctl clone

Creates an exact copy of the specified virtual machine.

Syntax

prlctl clone <*vm_ID*|*vm_name*> **--name** <*new_name*> [**--template**] [**--dst** <*path*>] [**--changesid**] [**--linked**] [**--detach-external-hdd** <**yes**|**no**>]

Options

Name	Description
<*vm_ID*|*vm_name*>	ID or name of the virtual machine to clone.
--name <*new_name*>	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.
--dst <*path*>	Full path to the directory where the new virtual machine will be stored. If this option is omitted, the new virtual machine will be created in the default directory.
--changesid	Generate a new Windows security identifier (SID) for a Windows-based virtual machine. For this parameter to work, guest tools must be installed in the virtual machine.
--linked	Create a linked virtual machine.
--detach-external-hdd <yes|no>	If set to `no`, hard disks located outside the source virtual machine are not removed from the configuration of the resulting clone. Setting the parameter to `yes` removes outside hard disks from the configuration. **Note**: Outside hard disks are not copied to the cloned virtual machine.

prlctl convert

This command is used to convert third-party virtual machines and disks to Virtuozzo virtual machines and disks. The following third-party virtual machines and disks are supported:

- Microsoft Hyper-V
- Microsoft Virtual PC
- Virtual Box
- VMware

Syntax

prlctl convert <*path*> [**--dst** <*path*>] [**--force**]

Options

Name	Description
<*path*>	Full path to the third-party virtual machine's configuration file on the local server.
--dst=<*path*>	Set the destination directory for the resulting virtual machine and its configuration file. If omitted, the default directory (`/var/parallels`) is used.
--force	Convert the third-party virtual machine even if its guest OS cannot be identified.

prlctl 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

prlctl create *<vm_name>* [*options*]

Options

Name	Description
<vm_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.
-d, --distribution *<name | list>*	The operating system distribution the virtual machine will be optimized for. For the full list of supported distributions, refer to the `prlctl` man pages.
--ostemplate *<template_name>*	The name of the virtual machine template from which to create the new virtual machine. Use the `prlctl list --template` command to obtain the list of the available templates.
--location *<vm_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.
--uuid *<uuid>*	A custom UUID to assign to the virtual machine.

Note: You can use either `--distribution` or `--ostemplate`, not both options at once.

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 Virtuozzo and are kept in sync with other Virtuozzo management tools such as Virtuozzo Automator. The best way to find out the default versions used in your Virtuozzo installation is by creating a sample virtual machine.

prlctl delete

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

Syntax

prlctl delete *<vm_ID|vm_name>*

Options

Name	Description
<vm_ID|vm_name>	The ID or name of the virtual machine to delete.

prlctl 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. Guest tools must be installed in a virtual machine to use this utility.

Syntax

prlctl enter exec *<vm_ID|vm_name>*

Options

Name	Description
<vm_ID|vm_name>	The ID or the name of the virtual machine.

prlctl exec

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

Syntax

prlctl exec *<vm_ID|vm_name> <command>*

Options

Name	Description
<vm_ID|vm_name>	The ID or the name of the virtual machine.
<command>	A command to execute.

prlctl list

Displays a list of virtual machines on the Hardware Node. Displays information on virtual machines on the Hardware Node.

Syntax

```
prlctl list --vmtype vm [-a,--all] [-o,--output <field>[,...]] [-s,--sort <field|-field>]
                        [-t,--template] [-j,--json]
prlctl list -i,--info --vmtype vm [vm_id|vm_name] [-f,--full] [-t,--template] [-j,--json]
```

Options

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 available virtual machine templates instead of actual virtual machines.
-o, --output <*field*>[,...]	Display only the specified fields. For the list of fields, see **prlctl list Output Parameters** (p. 151). Type field names in lower case. Separate multiple fields with commas.
-s, --sort <*field*|*-field*>	Sort virtual machines by the specified field in either ascending or descending order.
-i, --info	Display detailed information about the specified virtual machine.
-f, --full	Display detailed information about network cards in virtual machines. Used with the `--info` option.
<*vm_ID*|*vm_name*>	The ID or 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.
-j, -json	Produce machine-readable output in the JSON format.

prlctl list Output Parameters

Listed below are the parameters that can be specified after the `-o` switch.

Name	Output Column	Description
uuid	`UUID`	Virtual machine ID.
hostname	`HOSTNAME`	Virtual machine hostname.
name	`NAME`	Virtual machine name.
description	`DESCRIPTION`	Virtual machine description.
ostemplate	`OSTEMPLATE`	Specifies the name of the OS template the virtual machine is based on (e.g., `redhat-el5-x86`).
ip	`IP_ADDR`	Virtual machine IP address.
status	`STATUS`	Virtual machine status (e.g., running or stopped).

numproc	NPROC	The number of processes and threads allowed.
mac	MAC	Network device's MAC address.
netif	NETIF	Network devices in the virtual machine .
iolimit	IOLIMIT	The bandwidth the virtual machine is allowed to use for its disk input and output (I/O) operations, in bytes per second.
ha_enable	HA_ENABLE	Indicates whether the virtual machine is joined to the High Availability Cluster.
ha_prio	HA_PRIO	Virtual machine priority in the High Availability Cluster (0 is the lowest). Higher-priority virtual environments are restarted first in case of failures.

prlctl migrate

Migrates a virtual machine from one host to another.

Syntax

```
prlctl migrate [source_server/]<vm_ID|vm_name> <destination_server>[/vm_name]
               [--dst <path>] [--keep-src] [--switch-template] [--changesid]
               [--ssh <options>]
```

Options

Name	Description
<*vm_ID*|*vm_name*>	The source virtual machine ID or name.
source_server	The source server information. Use the following format to specify this info: `[user[:password]@]<server_IP_address_or_hostname>[:port]`
destination_server	The destination server information. If omitted, the migration will be performed locally. Use the following format to specify this info: `[user[:password]@]<server_IP_address_or_hostname>[:port]`
--dst <*path*>	Name and path of the directory on the destination server where the virtual machine files should be stored.
--keep-src	If this option is included, the original virtual machine will be left intact on the source server. If this option is omitted, the original virtual machine will be removed from the source server.
--switch-template	Converts the virtual machine to a template and a template to a virtual machine. For example, if the source virtual machine was a template, it becomes a full featured virtual machine after the migration, and vice versa.
--changesid	Changes the resulting virtual machine SID.
--ssh	Additional options to pass to `ssh` to connect to the destination server. All standard `ssh` options are supported. **Note:** Do not specify the destination server hostname or IP address as an

	`ssh` option.

prlctl mount, umount

Mounts or unmounts the hard disks of a virtual machine to the `/vz/mnt/<vm_ID>` directory on the host.

Syntax

prlctl mount *<vm_ID|vm_name>* [**-o** *<***ro|rw***>* | **--info**]
prlctl umount *<vm_ID|vm_name>*

Options

Name	Description
<vm_ID|vm_name>	Virtual machine ID or name.
-o <ro|rw>	Sets access rights: • `ro` — read-only, • `rw` — read-write.
--info	Show information about the mounted virtual disks.

prlctl move

Moves the files of a virtual machine to a new location on the same server. The virtual machine must be stopped or suspended.

Syntax

prlctl move *<vm_ID|vm_name>* **--dst** *<path>*

Options

Name	Description
<vm_ID|vm_name>	Virtual machine ID or name.
--dst *<path>*	Path to the new virtual machine files location.

prlctl pause, suspend, resume

Pause, suspend, and resume a virtual machine.

Syntax

```
prlctl pause <vm_ID|vm_name>
prlctl suspend <vm_ID|vm_name>
prlctl resume <vm_ID|vm_name>
```

Options

Name	Description
<vm_ID|vm_name>	The ID or 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 prlctl start (p. 171) command.

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.

prlctl problem-report

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

Syntax

```
prlctl problem-report <vm_ID|vm_name> <-d,--dump|-s,--send [--proxy
[user[:password]@]<proxyhost>[:port]] [--no-proxy]>
```

Options

Name	Description
<vm_ID|vm_name>	The ID or name of the virtual machine for which to obtain the problem report. If the name consists of separate words, it must be enclosed in quotes.
-d, --dump	Collect technical data about a virtual machine and display it on the screen. You can also pipe the output to a file and then send it to technical support to analyze your problem.
-s, --send	Send the generated problem report to technical support.
--proxy [user[:password]@]<proxyhost>[:port]	Use the specified information to send the generated report through a proxy server, if you use one to connect to the Internet.
--no-proxy	Do not use a proxy server to send the generated report. This is the default behavior, so you can omit this parameter.

prlctl register, unregister

The `register` command is used to register a virtual machine with Virtuozzo.

The `unregister` command removes a virtual machine from the Virtuozzo registry.

Syntax

prlctl register *<path>* **--preserve-uuid** **<yes|no>**
prlctl unregister *<vm_ID|vm_name>*

Options

Name	Description
<path>	An absolute path to the virtual machine directory.
<vm_ID|vm_name>	The ID or the name of the virtual machine to remove from the Virtuozzo registry.
--preserve-uuid <yes|no>	Specifies what to do with the UUID (universally unique identifier). If you specify `yes`, the UUID is preserved. If you specify `no`, the UUID is regenerated. **Note:** By default, IDs are regenerated.

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 Virtuozzo. 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 Virtuozzo registry, but does not delete the virtual machine files from the server. You can re-register such a machine with Virtuozzo later using the `register` command.

prlctl reset-uptime

Resets a virtual machine uptime counter as well as count start date and time.

Syntax

prlctl reset-uptime *<vm_ID|vm_name>*

Options

Name	Description
<vm_ID|vm_name>	Virtual machine ID or name. Names consisting of multiple words must be enclosed in quotes.

prlctl set

The `prlctl 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 `prlctl set` command is used to modify the virtual machine configuration parameters.

Syntax

```
prlctl set <vm_ID|vm_name> [--cpus <number>] [--memsize <number>] [--videosize <number>]
       [--memquota <auto|guarantee:limit[:priority[:maxballoon]]>] [--mem-hotplug
<on|off>]
       [--distribution <name>]
       [--description <desc>]
       [--autostart <on|off|auto>] [--autostart-delay <number>]
       [--autostop <stop|suspend>] [--applyconfig <conf>] [--name <new_name>]
       [--start-as-user <administrator|owner|user:passwd>]
       [--vnc-mode <auto|manual|off> {--vnc-passwd <passwd> | --vnc-nopasswd}]
       [--vnc-port <port>] [--vnc-address <address>]
       [--cpu-hotplug <on|off>]
       [--cpuunits <units>] [--cpulimit <percent|megahertz>]
       [--ioprio <priority>] [--iolimit <limit>] [--iopslimit <limit>]
       [--cpumask <N>[,N,N1-N2]]
       [--offline-management <on|off>] [--offline-service <service_name>]
       [--userpasswd <user:passwd> [--crypted]]
       [--rate <rate>] [--ratebound <on|off>]
       [--apply-iponly <yes|no>]
       [--efi-boot <on|off>]
       [--tools-autoupdate <on|off>]
```

Options

Name	Description
<*vm_ID*|*vm_name*>	Target virtual machine ID or name.
--cpus <*number*>	Number of virtual CPUs in the virtual machine. If the server has several CPU cores, this option also defines the number of CPUs shown to users from inside a 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.
--memquota <auto|*guarantee*:*limit*[:*priority*[:*maxballoon*]]>	Sets the parameters of the memory consumption by the virtual machine: • *guarantee*. The amount of memory a virtual machine is guaranteed to get on demand. By

	default, the guaranteed memory is calculated on the basis of RAM and video memory assigned to a virtual machine and is about a half of its total memory. • *limit*. The maximum amount of memory a virtual machine is allowed to consume. By default, no limit is set for all newly created virtual machines, and any virtual machine may consume all free memory on the server. • *priority*. The priority (from 1 to 100) that defines which virtual machine will get memory first. The higher the priority of a virtual machine, the more chances it has to get memory when the host has insufficient memory resources. By default, the priority is set to 50. • *ballooning*. The maximum amount of memory the balloon driver in a virtual machine may allocate for its needs. By default, the balloon driver can allocate up to 60% of RAM set for a virtual machine. • `auto`. Resets all memory parameters to their default values.
--mem-hotplug <on | off>	Enables or disables memory (RAM) hotplug support in the virtual machine. This feature is disabled in the virtual machine by default. The guest operating system must support memory hotplug for this functionality to work.
--distribution <*name*>	Optimize the virtual machine for use with the operating system <*name*>. You can get the list of available distributions using the `prlctl set <vm_name> -d list` command.
--description <*desc*>	Sets virtual machine description. Descriptions with white spaces must be enclosed in quotation marks.
--autostart <on | off | auto>	Defines the virtual machine start-up options: • `on` — the virtual machine is started automatically wen the host starts or the Virtuozzo 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 host or disabling the Virtuozzo component responsible for managing virtual machines. If you set this option to `on` or `auto`, you must

	additionally specify the `--start-as-user` option.
--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 host or disable the Virtuozzo component responsible for managing virtual machines. • `suspend` — the virtual machine is suspended when the host is shut down or the Virtuozzo component responsible for managing virtual machines is disabled.
--applyconfig <*conf*>	Applies the resource parameter values from the specified VM sample file in `/etc/parallels/samples` to the virtual machine. The following parameters are applied: • all memory-related parameters (both RAM and video) • all CPU-related parameters • IO and IOPS limits • disk size
--name <*new_name*>	Changes the virtual machine name.
--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.
--vnc-mode <auto | manual | off>	Enables or disables access to the virtual machine via the VNC protocol.
--vnc-port <*port*>	Sets the VNC port number.
--vnc-passwd <*passwd*> | **--vnc-nopasswd**	Sets the VNC password or specifies that no password is needed for VNC connections. Either of these options is mandatory for any VNC connection.
--vnc-address <*address*>	Sets the IP address to use for logging in to the virtual machine via VNC. It must be one of the IP addresses assigned to the host. By default, you can use any of the IP addresses of the host to log in to the virtual machine.

--cpu-hotplug <on|off>	Enables or disables CPU hotplug support in the virtual machine. This feature is disabled by default. The guest operating system must support CPU hotplug for this functionality to work.
--cpuunits <*units*>	Sets the CPU weight for the virtual machine. This is a positive integer number that defines how much CPU time the virtual machine can get as compared to the other virtual machines and Containers running on the server. The larger the number, the more CPU time the virtual machine can receive. Possible values range from 8 to 500000. If this parameter is not set, the default value of 1000 is used.
--cpulimit <*percent*|*megahertz*>	CPU limit, in percent or megahertz (MHz) the virtual machine is not allowed to exceed. By default, the limit is set in percent. To set the limit in MHz, specify "m" after the value. **Note**: If the server has 2 processors, the total CPU time equals 200%.
--ioprio <*priority*>	Disk I/O priority level from 0 to 7. The default is 4.
--iolimit <*limit*>	Disk I/O bandwidth limit. The default is 0 (no limit). By default the limit is set in megabytes per second. You can use the following letters following the number to specify units of measure: • G — gigabytes per second (e.g., 1G). • K — kilobytes per second (e.g., 10K). • B — bytes per second (e.g., 100B). The default I/O bandwidth limit for all newly created virtual machines is set to 0, which means that no limits are applied to them.
--iopslimit <*limit*>	Maximum number of disk input and output operations per second a virtual machine is allowed to perform. By default, any newly created Container does not have the IOPS limit set and can perform so many disk I/O operations per second as necessary.
--cpumask <*N*>[,*N*,*N1*-*N2*]	An affinity mask indicating what CPU(s) the virtual machine processes should be run on. You can specify a list of CPUs identified by their index numbers separated by commas (0, 1, 2, 3, etc.) or a range (4-6). To make all CPUs available for the virtual machine processes specify `--cpumask all`.
--offline-management <on|off>	Turns the offline management on or off.
--offline-service <*service_name*>	The name of the service to use for offline management.
--userpasswd <*user*:*passwd*>	Sets the password for the specified user in the virtual machine. If the user account does not exist, it will be created. Guest tools must be installed in the virtual

	machine for the command to work.
--crypted	Used with `--userpasswd`. Indicates that the specified password is already a hash.
--rate <*rate*>	Sets the guaranteed outgoing traffic rate in Kbps for the virtual machine.
--ratebound <on | off>	Turns the network traffic rate limitation set by the `--rate` parameter (above) on or off. The default value is `off`.
--apply-iponly <yes | no>	If set to `yes`, the hostname, nameserver, and search domain settings from the virtual machine configuration file are ignored.
--efi-boot <on | off>	If set to `on`, the virtual machine will boot using the EFI firmware. If set to `off` (default), the virtual machine will boot using the BIOS firmware.
--tools-autoupdate <on | off>	Enables or disables automatic update of guest tools inside Windows virtual machines. If set to `on`, guest tools are updated automatically on user log in. A reboot is required to complete the update. If set to `off`, guest tools are not updated automatically, so that you can do it manually at a convenient time.

Managing Virtual Devices

The `prlctl set` command allows to add, modify, and delete virtual devices of virtual machines.

Syntax

```
prlctl set <vm_ID|vm_name> --device-add <dev_type> <options>
prlctl set <vm_ID|vm_name> --device-set <dev_name> <options>
prlctl set <vm_ID|vm_name> --device-del <dev_name> <options> --destroy-image-force
prlctl set <vm_ID|vm_name> --device-connect <dev_name>
prlctl set <vm_ID|vm_name> --device-disconnect <dev_name>
prlctl set <vm_ID|vm_name> --device-bootorder "<dev_name1> <dev_name2> [...]"
```

Options

Name	Description
<*vm_ID*|*vm_name*>	The virtual machine ID or name.
--device-add <*dev_type*> <*options*>	Adds a virtual device of the type <*dev_type*> to a stopped virtual machine. The <*dev_type*> parameter can be: `hdd`, `cdrom`, `net`, `fdd`, `serial`, `parallel`, `sound`, `usb`, `pci`.
--device-set <*dev_name*> <*options*>	Modifies the configuration of the virtual device <*dev_name*> in a stopped virtual machine.

--device-del <*dev_name*> <*options*>	Deletes the virtual device <*dev_name*> from a stopped virtual machine.
--destroy-image-force	Used with the `--device-del` option. Deletes a virtual machine HDD even if it is used in that virtual machine's snapshots.
--device-connect <*dev_name*>	Connects the virtual device <*dev_name*> to a running virtual machine.
--device-disconnect <*dev_name*>	Disconnects the virtual device <*dev_name*> from a running virtual machine.
--device-bootorder "<*dev_name1*> <*dev_name2*> [...]**"**	Specifies the boot order for a virtual machine.

Note: Device names can be obtained with the `prlctl list -i` command.

The device-related <*options*> can be subdivided into the following categories:

- Hard disk drives (p. 161)
- Optical disk drives (p. 163)
- Network cards (p. 164)
- Floppy disk drives (p. 166)
- Serial ports (p. 166)
- Parallel ports (p. 167)
- USB devices (p. 168)
- Sound cards (p. 168)

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

Hard Disk Drive Management Options

This group of options is used to add and configure virtual hard disks in a virtual machine. The first syntax uses a file to emulate a hard disk drive. The second syntax connects a physical hard disk on the host server to the virtual machine.

Syntax

```
prlctl set <vm_ID|vm_name> {--device-add hdd | --device-set hdd<N>}
          [--image <file>] [--type <expanded|plain>] [--size <size>] [--split]
          [--iface <sata|ide|scsi>] [--position <pos>]
          [--subtype <buslogic|lsi-spi|lsi-sas>] [--enable|--disable]
prlctl set <vm_ID|vm_name> --device-add hdd --device <dev_name> [--position <pos>]
          [--iface <sata|ide|scsi>] [--subtype <buslogic|lsi-spi|lsi-sas>]
prlctl set <vm_ID|vm_name> --backup-add <backup_ID> [--disk <disk_name>]
          [--iface <sata|ide|scsi>] [--position <pos>] [--subtype <buslogic|lsi-spi|lsi-sas>]
prlctl set <vm_ID|vm_name> --backup-del <backup_ID>|all
```

Options

Name	Description
<*vm_ID*|*vm_name*>	The virtual machine ID or name.
--device-add hdd	Adds a virtual hard disk drive to the virtual machine. You can connect up to four SATA/IDE devices and up to seven SCSI devices to a virtual machine. This includes hard disks and optical disk drives. New hard disks are created in the virtual machine directory and are automatically named `harddisk<N>.hdd`, where <*N*> is the next available disk index.
--device-set hdd<*N*>	Modifies the parameters of an existing virtual hard disk. Virtual hard disks are named using the `hdd<N>` format where <*N*> is the drive index number starting from 0 (e.g., `hdd0`, `hdd1`). To obtain the list of disk names, use the `prlctl list` command with the `--info` option.
--image <*file*>	Specifies an image file that will be used to emulate the virtual disk. • If the specified image does not exist, it is created and used to emulate the virtual hard disk. • If the specified image exists, it is used to emulate the virtual hard disk.
--device <*dev_name*>	This option is used to connect a physical hard disk on the host to the virtual machine. You can obtain the names of the existing hard disks on the server using the `prlsrvctl info` command.
--type <**expanded**|**plain**>	Sets the disk type for image file based virtual hard disks: • `expanded` — 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 (the space is allocated to the drive fully). Plain disks perform faster than expanding disks.
--size <*size*>	The size of the virtual hard disk, in megabytes. The default size is 65536 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).
--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.
--backup-add <*backup_ID*>	Attach the backup with the identifier <*backup_ID*> to the virtual machine as a virtual hard disk. To obtain the backup ID, use the `prlctl backup-list -f` command (p. 107).
--disk <*disk_name*>	Used with `--backup-add`. The name of the disk in the backup to attach. If a disk is not specified, all disks contained in the backup will be attached. To obtain the disk name(s), use the `prlctl backup-list -f` command (p. 107).
--backup-del <*backup_ID*>|**all**	Detach either the backup with the identifier <*backup_ID*> or detach all backups from the virtual machine.

--iface <sata|ide|scsi>	The disk drive Interface type. If omitted, the SATA interface will be used.
--subtype <buslogic|lsi-spi|lsi-sas>	Virtual hard disk subtype.
--position *<pos>*	The SCSI, SATA or IDE device identifier to be used for the virtual disk. The allowed ID ranges are: • IDE devices: 0-3 • SATA devices: 0-5 • SCSI device: 0-6

•

Optical Disk Drive Management Options

This group of options is used to add and configure virtual optical disk drives, such as DVD or CD drives.

Syntax

```
prlctl set <vm_ID|vm_name> {--device-add cdrom | --device-set cdrom<N>}
          {--device <dev_name> | --image <file>} [--iface <ide|scsi>]
          [--position <pos>] [--subtype <buslogic|lsi-spi|lsi-sas>]
          [--enable|--disable] [--connect|--disconnect]
```

Options

Name	Description
<vm_ID|vm_name>	The virtual machine ID or name.
--device-add cdrom	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 cdrom*<N>*	Modifies the parameters of an existing virtual optical disk. The *N* postfix indicates the drive index number. To obtain the list of the available drives, use the `prlctl list` command with the `--info` option.
--device *<dev_name>*	The name of the physical optical disk to connect to the virtual machine.
--image *<file>*	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 *<pos>*	The SCSI, SATA or IDE device identifier to be used for the DVD/CD drive. The allowed ID ranges are the following: • IDE devices: 0-3 • SATA devices: 0-5

	• SCSI device: 0-6 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.
--subtype <**buslogic**|**lsi-spi**|**lsi-sas**>	Virtual hard disk subtype.
--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.

Network Adapter Management Options

This group of options is used to manage virtual network adapters in a virtual machine.

Syntax

```
prlctl set <vm_ID|vm_name> {--device-add net | --device-set net<N>}
           {--type <routed> | --network <network_ID>} [--mac <addr|auto>]
           [--ipadd <addr[/mask]> | --ipdel <addr[/mask]> |
           --dhcp <yes|no> | --dhcp6 <yes|no>] [--gw <gw>] [--gw6 <gw>]
           [--nameserver <addr>] [--searchdomain <addr>]
           [--configure <yes|no>] [--ipfilter <yes|no>] [--macfilter <yes|no>]
           [--preventpromisc <yes|no>] [--enable|--disable]
           [--connect|--disconnect] [--adapter-type <e1000|rtl|virtio>]
```

Options

Name	Description
<*vm_ID*|*vm_name*>	The virtual machine ID or name.
--device-add net	Adds a new virtual network adapter to the virtual machine.
--device-set net<*N*>	Modifies an existing virtual network adapter. To obtain the list of the available adapters, use the `prlctl list` command with the `--info` option.
--type <**routed**>	Sets the networking mode for the virtual network adapter to "routed". In this mode, the network adapter is communicating with the outside world through an internal virtual network adapter.
--network <*network_ID*>	Sets the networking mode for the virtual network adapter to "virtual_network". In this mode the adapter is connected to a virtual network specified by <*network_ID*>.

--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 generate the existing address automatically
--ipadd <*addr*[*/mask*]>	Adds an IP address and a mask (optional) to the network adapter.
--ipdel <*addr*[*/mask*]>	Deletes an IP address from the network adapter.
--dhcp <**yes** | **no**>	Specifies whether the virtual network adapter should obtain the IPv4 settings through a DHCP server.
--dhcp6 <**yes** | **no**>	Specifies whether the virtual network adapter should obtain the IPv6 settings through a DHCP server .
--gw <*gw*>	The default gateway to be used by the virtual machine.
--gw6 <*gw*>	The default IPv6 gateway to be used by the virtual machine.
--nameserver <*addr*>	The default DNS server address to be used by the virtual machine.
--searchdomain <*addr*>	The default search domain to be used by the virtual machine.
--configure <**yes** | **no**>	If set to "yes", the settings above are applied to the virtual network adapter instead of its original settings. Configuring any of the settings above automatically sets this option to "yes".
--ipfilter <**yes** | **no**>	Determines if the specified network adapter is configured to filter network packages by IP address. If set to "yes", the adapter is allowed to send packages only from IPs in the network adapter IP addresses list.
--macfilter <**yes** | **no**>	Determines if the specified network adapter is configured to filter network packages by MAC address. If set to "yes", the adapter is allowed to send packages only from its own MAC address.
--preventpromisc <**yes** | **no**>	Determines if the specified network adapter should reject packages not addressed to its virtual machine. If set to "yes", the adapter will drop such packages.
--enable | **--disable**	Enables or disable the network adapter. If omitted during the adapter creation, the adapter will be enabled.
--connect | **--disconnect**	Connects or disconnects the network adapter. When disconnected, the adapter is not removed from the virtual machine.
--adapter-type <**e1000** | **rtl** | **virtio**>	Emulated network adapter: • `e1000` — Intel 82545EM, • `rtl` — Realtek RTL8029, • `virtio` — VirtIO. **Note:** The adapter requires no additional configuration on supported Linux and FreeBSD guest operating systems. However, additional drivers need to be installed on Windows Server 2012 R2 guest OSes. For the drivers, visit https://alt.fedoraproject.org/pub/alt/virtio-win/latest/ https://alt.fedoraproject.org/pub/alt/virtio-win/latest/. The VirtIO adapter is not supported on Windows Server 2003 and Windows Server 2008 guest operating systems.

Floppy Disk Drive Management Options

This group of options is used to add a floppy disk drive to a virtual machine and to modify the existing virtual floppy disk drive.

Syntax

```
prlctl set <vm_ID|vm_name> {--device-add fdd | --device-set fdd0}
           {--device <dev_name> | --image <file>}
           [--enable|--disable] [--connect|--disconnect]
```

Options

Name	Description
<*vm_ID*|*vm_name*>	The virtual machine ID or name.
--device-add fdd	Adds a new floppy disk drive to the virtual machine. You can connect only one floppy disk drive to a virtual machine.
--device-set fdd0	Modifies the parameters of an existing virtual floppy disk drive.
--device <*dev_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.
--image <*file*>	The name and path of an existing floppy disk image file (usually `floppy.fdd`) to mount in the virtual machine.
--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.

Serial Port Management Options

This group of options is used to manage serial ports in a virtual machine.

Syntax

```
prlctl set <vm_ID|vm_name> {--device-add serial | --device-add serial<N>}
           {--device <dev_name> | --output <file> | --socket <name>}
           [--enable|--disable] [--connect|--disconnect]
```

Options

Name	Description
<*vm_ID*|*vm_name*>	The virtual machine ID or name.
--device-add serial	Adds a new serial port to the virtual machine. You can connect up to four serial ports to a virtual machine.
--device-set serial<*N*>	Modifies the parameters of an existing serial port.
--device <*dev_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|**--disable**	Enables or disables the virtual serial port. All newly added serial ports are enabled by default (provided the `--disable` option is omitted).
--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.

Parallel Port Management Options

This group of options is used to manage parallel port in a virtual machine.

Syntax

```
prlctl set <vm_ID|vm_name> {--device-add parallel | --device-set parallel<N>}
           {--device <dev_name> | --output <file>}
           [--enable|--disable] [--connect|--disconnect]
```

Options

Name	Description
<*vm_ID*|*vm_name*>	The virtual machine ID.
--device-add parallel	Adds a new parallel port to the virtual machine. You can connect up to three parallel ports to a virtual machine.
--device-set parallel<*N*>	Modifies the parameters of an existing virtual parallel port. To obtain the list of ports, use the `prlctl list` command with the `--info` option.
--device <*dev_name*>	The name of the physical parallel port to which to connect the virtual parallel port.
--output <*file*>	The name of the output file to which to connect the virtual parallel port.
--enable|**--disable**	Enables or disables the specified parallel port. All newly added parallel ports are enabled by default (provided the `--disable` option is omitted).
--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.

USB Controller Management Options

This group of options is used to manage the USB controller in a virtual machine.

Syntax

```
prlctl set <vm_ID|vm_name> --device-add usb [--enable|--disable]
```

Options

Name	Description
<*vm_ID*|*vm_name*>	The virtual machine ID or name.
--device-add usb	The type of the virtual device to add to the virtual machine (in this instance, a USB device).
--enable|**--disable**	Enables or disables the USB controller. The controller is enabled by default (provided the `--disable` option is omitted).

Sound Device Management Options

This group of options is used to manage sound devices in a virtual machine.

Syntax

```
prlctl set <vm_ID|vm_name> {--device-add sound | --device-set sound0}
          [--input <dev_name>] [--output <dev_name>]
             [--enable|--disable] [--connect|--disconnect]
```

Options

Name	Description
<*vm_ID*|*vm_name*>	The virtual machine ID or name.
--device-add sound	The type of the virtual device to add to the virtual machine (in this instance, a sound device).
--device-set sound0	Modifies the parameters of the existing virtual sound device.
--output <*dev_name*>	The name of a physical output device to connect to the virtual sound device.
--input <*dev_name*>	The name of the physical input device to connect to the virtual sound device.
--enable|**--disable**	Enables or disables the specified sound device. All newly added sound devices are enabled by default (provided the `--disable` option is omitted).
--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.

Removing Devices from Virtual Machines

The `--device-del` option is used to remove virtual devices from a virtual machine.

Syntax

```
prlctl set <vm_ID|vm_name> --device-del <dev_name> [--detach-only|--destroy-image]
```

Options

Name	Description
<dev_name>	The name of the virtual device to delete from the virtual machine. To obtain the list of virtual devices, use the `prlctl list -i` command.
--detach-only	Deletes the information about the specified device from the virtual machine configuration.
--destroy-image	Deletes the information about the specified device from the virtual machine configuration and removes the device from the server.

Managing Shared Folders

The `prlctl set` command can be used to add shared folders to a virtual machine and to modify and delete existing shared folders.

Syntax

```
prlctl set <vm_ID|vm_name> {--shf-host-add <folder> | --shf-host-set <folder>}
          --path <path>
          [--mode <ro|rw>] [--shf-description <desc>] [--enable|--disable]
prlctl set <vm_ID|vm_name> --shf-host <on|off>
prlctl set <vm_ID|vm_name> --shf-host-del <folder>
prlctl set <vm_ID|vm_name> --shf-guest <on|off>
prlctl set <vm_ID|vm_name> --shf-guest-automount <on|off>
```

Options

Name	Description
<vm_ID|vm_name>	The virtual machine ID or name.
--shf-host-add *<folder>*	Shares the specified folder on the host with the virtual machine.
--shf-host-set *<folder>*	Modifies the settings of an existing shared folder.
--shf-host-del *<folder>*	Removes the specified shared folder from the shared folder list.

--shf-host <on|off>	Turns the host folder sharing on or off.
--shf-guest <on|off>	Turns the guest folder sharing on or off.
--shf-guest-automount <on|off>	Mounts or unmounts virtual disks on the host.
--path <*path*>	Name and path of a folder on the host to share with the specified virtual machine.
--mode <ro|rw>	Sharing mode: • `ro` — read-only. • `rw` — read and write.
--shf-description <*desc*>	User-defined shared folder description. Descriptions with white spaces must be enclosed in quotation marks.
--enable|--disable	Enable or disable the shared folder.

prlctl snapshot, snapshot-list, snapshot-switch, snapshot-delete

Takes, displays, reverts to, and deletes snapshots of a running virtual machine.

Syntax

prlctl snapshot <*vm_ID|vm_name*> [**-n,--name** <*name*>] [**-d,--description** <*desc*>]
prlctl snapshot-list <*vm_ID|vm_name*> [**-t,--tree**] [**-i,--id** <*snapshot_ID*>]
prlctl snapshot-switch <*vm_ID|vm_name*> **-i,--id** <*snapshot_ID*> **--skip-resume**
prlctl snapshot-delete <*vm_ID|vm_name*> **-i,--id** <*snapshot_ID*>

Options

Name	Description
<*vm_ID|vm_name*>	The virtual machine ID or name.
-n, --name <*name*>	User-defined snapshot name. Names with white spaces must be enclosed in quotation marks.
-d, --description <*desc*>	User-defined snapshot description. Descriptions with white spaces must be enclosed in quotation marks.
-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*>	• Use with `prlctl snapshot-list` to specify the ID of the snapshot to use as the root. If this parameter is omitted, the entire snapshot tree will be displayed. • Use with `prlctl snapshot-switch` to specify the ID of the snapshot to revert to. • Use with `prlctl snapshot-delete` to specify the ID of the

	snapshot to delete.
--skip-resume	Skips automatic virtual machine resume when switching to snapshots of running virtual machines.

Note: If the snapshot you want to delete has child snapshots derived from it, they will not be deleted.

prlctl server

Obtains the information about the host and Virtuozzo. It also allows you to disable the Virtuozzo component responsible for managing virtual machines.

Syntax

prlctl server <**shutdown**|**info**>

Options

Name	Description
info	Displays the Virtuozzo information.
shutdown	Disables the Virtuozzo 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. 47)

prlsrvctl shutdown (p. 53)

prlctl start, stop, restart, reset, status

Start, stop, reset, and check the status of a virtual machine.

Syntax

prlctl start *<vm_ID|vm_name>*
prlctl stop *<vm_ID|vm_name>* [**--kill**]
prlctl restart *<vm_ID|vm_name>*
prlctl reset *<vm_ID|vm_name>*
prlctl status *<vm_ID|vm_name>*

Options

Name	Description
<vm_ID|vm_name>	The ID or name of the virtual machine to start, stop, restart, reset, or check the status of.

--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 guest tools package is installed in a virtual machine, the graceful shutdown will be performed using its facilities.
- If the guest 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 `restart` command first gracefully shuts down a virtual machine and then starts it again.

The `reset` command first performs a 'hard' virtual machine shutdown and then starts it again.

The `start` command can be used to start a stopped virtual machine or to resume a paused virtual machine (p. 153).

prlctl statistics

Obtains Virtuozzo statistics for a specified virtual machine.

Syntax

prlctl statistics *<vm_ID|vm_name>* [**--loop**] [**--filter** *<pattern>*]

Options

Name	Description
--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 *<pattern>*	Filters output by a specified pattern that supports wildcards (asterisk).

Guest Disk Usage Performance Counters

Name	Description
guest.fs<*N*>**.name**	Device name in a guest file system.
guest.fs<*N*>**.total**	Total size of a file system, in kilobytes.
guest.fs<*N*>**.free**	The amount of free space in a file system.
guest.fs<*N*>**.disk.**<*N*>	Disk indices.

Note: Guest disk usage performance counters are only available for virtual machines with installed guest tools.

pmigrate

The `pmigrate` utility is used to perform different kinds of migration.

Syntax

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

The <*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). The <*destination_server*> is the destination server—that is, the host where the virtual machine and Container or the physical server is to be migrated. If the source and/or destination server are not specified, the operation is performed on the local server.

Both <*source_server*> and <*destination_server*> consist of <*type*> and <*address*>:

- <*type*> denotes the type of computer to migrate and can be one of the following:
  - `h` for migrating physical computers
  - `c` for migrating Containers
  - `v` for migrating Virtuozzo virtual machines
  - `x` for migrating Xen virtual machines
- <*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>[:port]
```

The [*options*] you can use with `pmigrate` depend on whether you are migrating a virtual machine or a Container. This section describes the options you can use to migrate Virtuozzo and Xen virtual machines, Containers, and physical computers.

General Options

Name	Description
--dst=<*path*>	Specifies the name and path of the directory on the destination server to use for storing the virtual machine files. If this option is omitted, the default directory is used.
-h, --help	Displays the information on the utility usage.

Virtual Machine Migration Options

Name	Description
--keep-src	If this option is included, the original virtual machine is not removed from the source server after migration. By default, the original virtual machine is removed from the source server.
--switch-template	Switches the virtual machine to a template and a template to a virtual machine. For example, if the source virtual machine was a template, it becomes a full featured virtual machine after the migration, and vice versa.
--changesid	Changes the resulting virtual machine SID.

Container-to-Virtual Machine Migration Options

Name	Description
-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 host will be used. The following size modifiers can be used: G — gigabytes, M — megabytes (default), K — kilobytes.
-r, --reg[=<y|n>]	Specifies whether to register the resulting virtual machine on the host. 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 host. 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).

Physical Computer-to-Virtual Machine Migration Options

Name	Description
-r, --reg[=<y|n>]	Specifies whether to register the resulting virtual machine on the host. 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: • For a Windows source computer, only boot and system volumes are migrated. • For a Linux source computer, only the partitions specified in the `/etc/fstab` file on the source are 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).

Xen Virtual Machine-to-Virtual Machine Migration Options

Name	Description
-r, --reg[=<y|n>]	Specifies whether to register the resulting virtual machine on the host. 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 virtual machine.
--key=<*auth_key*>	Set the authentication key for passwordless access to the Xen server. To set this key on the Xen server, use the `parallels-transporter-agent --regkey` <*value*> command (<*auth_key*> can be any alphanumeric value).
--ip=<*IP*>[*/mask*]	Sets the IP address for the resulting virtual machine. If this option is omitted, the DHCP support is enabled for the virtual machine.
--gw=<*IP*>	Sets the default gateway for the resulting virtual machine. If this options is omitted, the DHCP support is enabled for the virtual machine.
--nameserver=<*IP*>	Sets the DNS server for the resulting virtual machine.
--searchdomain=<*domain*>	Sets the DNS search domain for the resulting virtual machine.

Managing Virtual Machine Backups

This section describes the utilities you can use for creating and managing virtual machine backups.

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 `/vz/vmprivate/backups` directory. Later on, the virtual machine backups can be restored from this directory.

Syntax

pbackup [*backup_options*] *server* [...] [*vm_options*]

You may specify any number of server names or IP addresses in the command-line.

Notes:

1. This section describes only backup options for virtual machines. For backup options that can be used with Containers, see **pbackup** (p. 104).

2. The backup settings in the `/etc/vzbackup.conf` file do not apply to virtual machines. To change some of the backup-related settings for virtual machines, you can use the `prlsrvctl` utility.

Backup Options

Name	Description
-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	Create a full backup. A full backup contains 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.

Virtual Machine Options

Name	Description
-e <*vm_ID*|*vm_name*> [...]	The virtual machines to back up on the server. Virtual machines can be specified using both their IDs and their names.
-x <*vm_ID*|*vm_name*> [...]	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 run on the Backup Server. It uses the virtual machine backups stored on the Backup Server to restore them to their original servers.

Syntax

```
prestore [restoration_options] server [...] [vm_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.

Restoration Options

Name	Description
-l	Do not restore any virtual machines. Show the information on the virtual machines available for restoring.
-n <*backup_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 format for the Backup Server is as follows: [*user*[:*password*]@]<*server_IP_address_or_hostname*>[:*port*]
-r <*backup_ID*>	Remove the backup with the specified ID.
-c <*path*>	Use an alternative configuration file.

Virtual Machine Options

Name	Description
-e <*vm_ID*|*vm_name*> [...]	The virtual machines to restore on the server. Virtual machines can be specified using both their IDs and their names.
-x <*vm_ID*|*vm_name*> [...]	The virtual machines that need not be restored (virtual machines to exclude). Virtual machines can be specified using both their IDs and their names.

prlctl backup, backup-list, backup-delete, restore

Creates, lists, deletes or restores virtual machine backups.

Syntax

```
prlctl backup <vm_ID|vm_name> [-f,--full] [-i,--incremental] [--description <desc>]
                  [-s,--storage [user[:passwd]@]server[:port]
prlctl backup-list [vm_ID|vm_name] [-f,--full] [--localvms]
                       [-s,--storage [user[:passwd]@]server[:port]]
prlctl backup-delete {<vm_ID|vm_name> | -t,--tag <backup_ID>}
                         [-s,--storage [user[:passwd]@]server[:port]]
```

prlctl restore {*vm_ID*|*vm_name* | **-t**,**--tag** <*backup_ID*>} [**-n**, **--name** <*new_name*>] [**--dst** <*path*>] [**-s**,**--storage** [*user*[**:***passwd*]**@**]*server*[**:***port*]]

Options

Name	Description
<*vm_ID*|*vm_name*>	Virtual machine ID or name. • Use with `prlctl backup` to create a backup of the specified virtual machine. • Use with `prlctl backup-list` to list backups of the specified virtual machine. • Use with `prlctl backup-delete` to delete all backups of the specified virtual machine. • Use with `prlctl restore` to restore the most recent backup of the specified virtual machine.
-s, --storage [*user*[**:***passwd*]**@**]*server*[**:***port*]]	Specifies a remote backup server address, port, and credentials. If this option is omitted, the backup will be saved on the default backup server that can be configured using the **prlsrvctl set** command.
--description <*desc*>	Backup description. Descriptions with white spaces must be enclosed in quotation marks.
-f, --full	• Use with `prlctl backup` to create a full backup of the virtual machine. A full backup contains all virtual machine data. • Use with `prlctl backup-list` to display full backup information.
-i, --incremental	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.
--localvms	List local backups only.
-t, --tag <*backup_ID*>	The ID of the backup to restore or delete.
-n, --name <*new_name*>	A new name to assign to the restored virtual machine. If omitted, the virtual machine will be restored with the original name.
--dst <*path*>	Restore the virtual machine to the specified directory on the host. If this option is omitted, the virtual machine will be restored to `/var/parallels/`<*vm_name*>.

prl_convert

This utility is used to convert third-party virtual disks to Virtuozzo virtual machines and disks. The following third-party disks are supported:

- Microsoft Hyper-V
- Microsoft Virtual PC
- Virtual Box
- VMware

Syntax

```
prl_convert <src> [options]
prl_convert <src> --estimate
```

If the disk is a data disk, `prl_convert` converts it to a Virtuozzo virtual disk. If a disk is a system disk, `prl_convert` converts it to a Virtuozzo virtual machine. If the utility cannot create a virtual machine for the disk (for example, it fails to detect the operating system on the disk), the disk is converted to a Virtuozzo virtual disk.

Options

Name	Description
--dst=<*path*>	Set the destination directory. If omitted, the default directory (`/var/parallels`) is used.
-r, --reg=<y|n>	Register the resulting virtual machine on the host. By default, the virtual machine is registered.
--os-files=<*path*>	Set the path to the operating system installation files. These files may be needed when reconfiguring the resulting virtual machine or disk.
--vbox-home=<*path*>	Specify the path to the directory storing virtual machine configuration files. This option should be used only when converting VirtualBox virtual machines.
--err-code	Show error codes instead of text messages.
--allow-no-os	Convert data disks or disks whose operating system cannot be detected.
--no-reconfig	Skip the reconfiguration step when converting a virtual disk.
--no-src-check	Do not check the source virtual disk state. Use this option with caution because disks of virtual machines that were running before the conversion may be in an inconsistent state after conversion.
--estimate	Estimate the disk size required for conversion, but do not perform the conversion.
-h, --help	Display the utility usage information.

prl_disk_tool

The `prl_disk_tool` utility is used to manage virtual hard disk drives.

Syntax

```
prl_disk_tool <command> [options] --hdd <disk_path> [options]
prl_disk_tool --help
```

prl_disk_tool compact

Removes all empty blocks from the expanding virtual disk to reduce its size on the physical hard disk. The virtual disk must be formatted to NTFS, FAT16, FAT32, ext2, or ext3. You can also try to compact virtual disks with other filesystems using the `--buildmap` option.

Syntax

```
prl_disk_tool compact [--buildmap] --hdd <disk_path> [--force]
prl_disk_tool compact -i,--info --hdd <disk_path>
```

Options

Name	Description
--hdd <*disk_path*>	Full path to the virtual disk.
--buildmap	Compacts 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.

prl_disk_tool configure

Prepares a disk or a real Boot Camp partition for booting into a virtual machine.

Syntax

```
prl_disk_tool configure --hdd <disk_path> [--para <paravirt_driver>] [--boot <boot_driver>]
```

Options

Name	Description
configure	Prepares a disk or a real Boot Camp partition for booting into a virtual machine.
--hdd <*disk_path*>	Full path to the virtual disk.

--para <*paravirt_driver*>	Specifies the full path to the Virtuozzo virtualization driver.
--boot <*boot_driver*>	Specifies the full path to the Virtuozzo boot driver.

prl_disk_tool convert

Converts the virtual disk between plain and expanding formats; splits the virtual disk into 2GB files or merges them back.

Syntax

```
prl_disk_tool convert --hdd <disk_path> {--plain|--expanding}
prl_disk_tool convert --hdd <disk_path> {--split|--merge}
prl_disk_tool convert --hdd <disk_path> --extend
prl_disk_tool convert -i,--info --hdd <disk_path>
```

Options

Name	Description
--hdd <*disk_path*>	Full path to the virtual disk to convert.
-i, --info	Do not convert the virtual disk; just show the size the disk will have after resizing.
--plain	Converts the disk to the plain format. A plain virtual hard disk has a fixed size.
--expanding	Converts the disk to the expanding format. An expanding virtual hard disk is small initially. Its size grows as you add applications and data to it.
--split	Splits the virtual hard disk into 2 GB files.
--merge	Merges all parts of the split disk into one file.
--extend	Enables sizes over 2TB for the specified expandable disk.

prl_disk_tool encrypt, decrypt

Encrypts or decrypts data on the specified virtual hard disk.

Syntax

```
prl_disk_tool encrypt --hdd <disk_path> [--force]
prl_disk_tool decrypt --hdd <disk_path>
```

Options

Name	Description
--hdd <*disk_path*>	Full path to the virtual disk to convert.
--force	Forcibly encrypt the virtual hard disk even if it is already encrypted.

prl_disk_tool merge

Merges all snapshots of the virtual hard disk.

Syntax

```
prl_disk_tool merge --hdd <disk_path>
```

Options

Name	Description
--hdd <*disk_path*>	Full path to the virtual disk.

prl_disk_tool resize

Changes the capacity of the specified virtual disk. During resizing, all data present on the disk volumes are left intact. You can also resize the last partition using the --resize_partition option. The supported file systems are NTFS, FAT16/FAT32, and ext2/ext3.

Syntax

```
prl_disk_tool resize --size <size>[M|G] [--resize_partition] --hdd <disk_path> [--force] [--split]
prl_disk_tool resize -i,--info [--units <K|M|G>] --hdd <disk_path>
```

Options

Name	Description
--size	New size of 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. **Note:** You cannot reduce XFS filesystems on LVM (the default choice for CentOS 7 and Red Hat Enterprise Linux 7).
--hdd <*disk_path*>	Full path to the virtual disk.
--force	Forces the resizing operation for suspended virtual disks.
--split	Splits the virtual hard disk into 2 GB files.
-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), or gigabytes (`G`).

prl_disk_tool validate

Validates the operating system on the virtual hard disk.

Syntax

prl_disk_tool validate --hdd *<disk_path>*

Options

Name	Description
--hdd *<disk_path>*	Full path to the virtual disk.

pnetstat

This utility outputs traffic usage statistics for virtual machines.

Syntax

pnetstat [**-v** *<ID>*] [**-c** *<class>*] [**-a**] [**-r**] [**-h**] [**-t** <**ct**|**vm**|**all**>]

The utility displays input and output traffic for virtual machines for each defined network class, in bytes. The network classes are described in the `/etc/vz/conf/networks_classes` file. `pnetstat` displays statistics only on virtual machines that were started at least once.

Options

Name	Description
-v *<ID>*	Display statistics for the specified virtual machine. Multiple `-v` options can be given to a single `pnetstat` invocation.
-c *<class>*	Display statistics for the *<class>* class only.
-a	Display statistics for all classes.
-r	Rounds down the statistics results, which is shown in bytes by default. In this case, the statistics is displayed in the following units of measurement, depending on the amount of traffic: • `K(b)` — kilobytes, • `M(b)` — megabytes, • `G(b)` — gigabytes.

-t <ct | vm | all>	Display statistics for: • `ct` — Containers only, • `vm` — virtual machines only, • `all` — both Container and virtual machines.
-h, -help	Display the utility usage information.

CHAPTER 5

Glossary

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

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

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

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

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

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

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

Package set. See *Template*.

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

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

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

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

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

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

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

UBC. An abbreviation of *User Beancounter*.

User Beancounter. The subsystem of the Virtuozzo software for managing *Container* memory and some system-related resources.

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

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

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

Index

P

R

S

U

V