# OpenVZ: manual pages

## Table of Contents

## NAME

vzctl − perform various operations on an OpenVZ container

## SYNOPSIS

**vzctl** [*flags*] **create** *CTID* --*parameter value* [...]
**vzctl** [*flags*] **start** *CTID* [**--wait**] [**--force**] [**--skip-fsck**] [**--skip-remount**]
**vzctl** [*flags*] **stop** *CTID* [**--fast**] [**--skip-umount**]
**vzctl** [*flags*] **restart** *CTID* [**--wait**] [**--force**] [**--fast**] [**--skip-fsck**] [**--skip-remount**]
**vzctl** [*flags*] **suspend** | **resume** *CTID* [**--dumpfile** *name*]
**vzctl** [*flags*] **snapshot** *CTID* [**--id** *uuid*] [**--name** *name*] [**--description** *desc*] [**--skip-suspend**] [**--skip-config**]
**vzctl** [*flags*] **snapshot-switch** *CTID* [**--skip-resume** | **--must-resume**] [**--skip-config**] **--id** *uuid*
**vzctl** [*flags*] **snapshot-delete** *CTID* **--id** *uuid*
**vzctl** [*flags*] **snapshot-mount** *CTID* **--id** *uuid* **--target** *dir*
**vzctl** [*flags*] **snapshot-umount** *CTID* **--id** *uuid*
**vzctl** [*flags*] **snapshot-list** *CTID* [**-H**] [**-o** *field*[,*field*...] [**--id** *uuid*]
**vzctl** [*flags*] **set** *CTID* --*parameter value* [...] [**--save**] [**--force**] [**--setmode restart**|**ignore**]
**vzctl** [*flags*] **set** *CTID* **--reset_ub**
**vzctl** [*flags*] **destroy** | **delete** | **mount** | **umount** | **status** | **quotaon** | **quotaoff** | **quotainit** *CTID*
**vzctl** [*flags*] **console** *CTID* [*ttynum*]
**vzctl** [*flags*] **convert** *CTID* [**--layout ploop**[**:**{**expanded**|**plain**|**raw**}]]
**vzctl** [*flags*] **compact** *CTID*
**vzctl** [*flags*] **exec** | **exec2** *CTID command* [*arg* ...]
**vzctl** [*flags*] **enter** *CTID* [**--exec** *command* [*arg* ...]]
**vzctl** [*flags*] **runscript** *CTID script*
**vzctl --help** | **--version**

## DESCRIPTION

Utility **vzctl** runs on the host system (otherwise known as Hardware Node, or HN) and performs direct manipulations with containers (CTs).

Containers can be referred to by either numeric *CTID* or by name (see **--name** option). Note that CT ID <= 100 are reserved for OpenVZ internal purposes. A numeric ID should not be more than **2147483644**.

## OPTIONS

### Flags

These flags come before a command, and can be used with any command. They affect logging to console (terminal) only, and do not affect logging to a log file.

**--quiet**
Disables output. Note that scripts run by vzctl are still able to produce some output.

**--verbose**
Increments logging level up from the default. Can be used multiple times. Default value is set to the value of **VERBOSE** parameter in the global configuration file **vz.conf**(5), or to **0** if not set by **VERBOSE** parameter.

### Setting container parameters

**set** *CTID* [**--onboot yes**|**no**] [**--bootorder** *number*] [**--root** *path*] [**--private** *path*] [**--mount_opts** *options*] [**--userpasswd** *user*:*pass*] [**--disabled yes**|**no**] [**--name** *name*]

[**--description** *string*] [**--ostemplate** *string*] [**--stop-timeout** *seconds*] [**--ipadd** *addr*]
[**--ipdel** *addr*|**all**] [**--hostname** *name*] [**--nameserver** *addr*] [**--searchdomain** *name*]
[**--netif_add** *dev*[,*params*...]] [**--netif_del** *dev*|**all**] [**--ifname** *dev* [**--mac** *hwaddr*]
[**--host_ifname** *dev*] [**--host_mac** *hwaddr*] [**--bridge** *name*] [**--mac_filter on**|**off**]]
[**--numproc** *items*] [**--numtcpsock** *items*] [**--numothersock** *items*] [**--vmguarpages** *pages*]
[**--kmemsize** *bytes*] [**--tcpsndbuf** *bytes*] [**--tcprcvbuf** *bytes*] [**--othersockbuf** *bytes*]
[**--dgramrcvbuf** *bytes*] [**--oomguarpages** *pages*] [**--lockedpages** *pages*]
[**--privvmpages** *pages*] [**--shmpages** *pages*] [**--numfile** *items*] [**--numflock** *items*]
[**--numpty** *items*] [**--numsiginfo** *items*] [**--dcachesize** *bytes*] [**--numiptent** *num*]
[**--physpages** *pages*] [**--swappages** *pages*] [**--ram** *bytes*] [**--swap** *bytes*]
[**--vm_overcommit** *float*] [**--cpuunits** *num*] [**--cpulimit** *num*] [**--cpus** *num*]
[**--cpumask** *cpus*|**auto**|**all**] [**--nodemask** *nodes*|**all**] [**--meminfo none**|*mode*:*value*]
[**--iptables** *name*[,...]] [**--netfilter disabled**|**stateless**|**stateful**|**full**] [**--netdev_add** *ifname*]
[**--netdev_del** *ifname*] [**--diskquota yes**|**no**] [**--diskspace** *num*] [**--diskinodes** *num*]
[**--quotatime** *seconds*] [**--quotaugidlimit** *num*] [**--capability** *capname*:**on**|**off**[,...]]
[**--devnodes** *param*] [**--devices** *param*] [**--pci_add** *dev*] [**--pci_del** *dev*]
[**--features** *name*:**on**|**off**[,...]] [**--applyconfig** *name*] [**--applyconfig_map** *group*]
[**--ioprio** *num*] [**--iolimit** *mbps*] [**--iopslimit** *iops*] [**--save**] [**--force**] [**--reset_ub**]
[**--setmode restart**|**ignore**]

This command sets various container parameters. If the container is currently running, **vzctl** applies these parameters to the container. The following options can be used with **set** command.

## Flags

**--save**
If this flag is given, parameters are saved in container configuration file **ctid.conf**(5).

**--force**
If this flag is given together with **--save**, parameters are saved even if the current kernel doesn't support OpenVZ. Note this flag does not make sense without **--save**, so **--save** is required.

**--reset_ub**
If this flag is given, **vzctl** applies all User Beancounter parameters from the configuration file to a running container. This is helpful in case configuration file is modified manually. Please note this flag is exclusive, i.e. it can not be combined with any other options or flags.

**--setmode restart** | **ignore**
A few parameters can only be applied by restarting the container. By default, **vzctl** prints a warning if such parameters are supplied and a container is running. Use **--setmode restart** together with **--save** flag to restart a container in such a case, or **--setmode ignore** to suppress the warning.

## Miscellaneous

**--onboot yes** | **no**
Sets whether the container will be started during system boot. The container will be started on boot by **vz** initscript if either this parameter is set to **yes**, or the container was running just before last reboot, and this parameter is not set to **no**. Default value is unset, meaning the

container will be started if it was running before the last reboot.

**--bootorder** *number*
Sets the boot order priority for this CT. The higher the *number* is, the earlier in the boot process this container starts. By default this parameter is unset, which is considered to be the lowest priority, so containers with unset **bootorder** will start last.

**--root** *path*
Sets the path to root directory (**VE_ROOT**) for this container. This is essentially a mount point for container's root directory. Argument can contain literal string **$VEID**, which will be substituted with the numeric CT ID.

**--private** *path*
Sets the path to private directory (**VE_PRIVATE**) for this container. This is a directory in which all the container's files are stored. Argument can contain literal string **$VEID**, which will be substituted with the numeric CT ID.

**--mount_opts** *option*[**,***option*...]
Sets additional mount options for container file system. Only applicable for **ploop** layout, ignored otherwise.

**--userpasswd** *user*:*password*
Sets password for the given user in a container, creating the user if it does not exists. Note that this option is not saved in configuration file at all (so **--save** flag is useless), it is applied directly to the container, by running distribution-specific programs inside the container. It is not recommended to combine this option with any other options.

In case container was not running, it is automatically started then all the appropriate changes are applied, then it is stopped.

Note that container should be created before using this option.

**--disabled yes** | **no**
Disable container start. To force the start of a disabled container, use **vzctl start --force**.

**--name** *name*
Add a name for a container. The *name* can later be used in subsequent calls to **vzctl** in place of *CTID*. Note this option can not be used without **--save**.

**--description** *string*
Add a textual description for a container.

**--ostemplate** *string*
Sets a new value of **OSTEMPLATE** parameter in container configuration file **ctid.conf**(5). Requires **--save** flag. Useful after a change/upgrade of a distribution running inside container, as vzctl uses the value of OSTEMPLATE to run distribution-specific scripts.

**--stop-timeout** *seconds*
Sets a time to wait for container to stop on **vzctl stop** before forcibly killing it, in seconds. Note this option can not be used without **--save** flag.

Special value of **0** means to use compiled-in default.

## Networking

**--ipadd** *addr*

Adds an IP address *addr* to a given container. Address can optionally have a netmask specified in the CIDR notation (e.g. **10.1.2.3/25**). Note that this option is incremental, so *addr* are added to already existing ones.

**--ipdel** *addr* | **all**

Removes IP address *addr* from a container. If you want to remove all the addresses, use **--ipdel all**.

**--hostname** *name*

Sets container hostname. **vzctl** writes it to the appropriate file inside a container (distribution-dependent).

**--nameserver** *addr*

Sets DNS server IP address for a container. If you want to set several nameservers, you should do it at once, so use **--nameserver** option multiple times in one call to **vzctl**, as all the name server values set in previous calls to **vzctl** are overwritten.

A special value of **inherit** can be used to auto-propagate nameserver value(s) from the host system's **/etc/resolv.conf** file.

**--searchdomain** *name*

Sets DNS search domains for a container. If you want to set several search domains, you should do it at once, so use **--searchdomain** option multiple times in one call to **vzctl**, as all the search domain values set in previous calls to **vzctl** are overwritten.

A special value of **inherit** can be used to auto-propagate search domain value(s) from the host system's **/etc/resolv.conf** file.

**--netif_add** *ifname[*,*mac*,*host_ifname*,*host_mac*,*bridge]*

Adds a virtual Ethernet device (veth) to a given container. Here *ifname* is the Ethernet device name in the container, *mac* is its MAC address, *host_ifname* is the Ethernet device name on the host, and *host_mac* is its MAC address. MAC addresses should be in the format like XX:XX:XX:XX:XX:XX. *bridge* is an optional parameter which can be used in custom network start scripts to automatically add the interface to a bridge. All parameters except *ifname* are optional and are automatically generated if not specified.

**--netif_del** *dev_name* | **all**

Removes virtual Ethernet device from a container. If you want to remove all devices, use **all**.

## veth interface configuration

The following options can be used to reconfigure the already-created virtual Ethernet interface. To select the interface to configure, use **--ifname** *name* option.

**--mac** *XX:XX:XX:XX:XX:XX*

MAC address of interface inside a container.

**--host_ifname** *name*

interface name for virtual interface in the host system.

**--host_mac** *XX:XX:XX:XX:XX:XX*
MAC address of interface in the host system.

If you want an independent communication with the Container through the bridge, you should specify a multicast MAC address here (FE:FF:FF:FF:FF:FF).

**--bridge** *name*
Bridge name. Custom network start scripts can use this value to automatically add the interface to a bridge.

**--mac_filter on** | **off**
Enables/disables MAC address filtering for the Container veth device and the possibility of configuring the MAC address of this device from inside the Container. If the filtering is turned on:
• the veth device accepts only those packets that have a MAC address in their headers corresponding to that of this device (excluding all broadcast and multicast packets);
• it is impossible to modify the veth MAC address from inside the Container.

By default, this functionality is enabled for all veth devices existing inside the Container.

## VSwap limits

The following options sets memory and swap limits for VSwap-enabled kernels (kernel version 042stab042 or greater).

Argument is in bytes, unless otherwise specified by an optional suffix. Available suffixes are:

| | |
|---|---|
| • **T**, **t** | - terabytes; |
| • **G**, **g** | - gigabytes; |
| • **M**, **m** | - megabytes; |
| • **K**, **k** | - kilobytes; |
| • **P**, **p** | - memory pages (arch-specific, usually 4KB); |
| • **B**, **b** | - bytes (this is the default). |

**--ram** *bytes*
Sets physical memory (RAM) available to a container. Actually, the option is a shortcut for setting **--physpages** limit (the barrier is set to 0).

**--swap** *bytes*
Set swap space available to a container. Actually, the option is a shortcut for setting **--swappages** limit (the barrier is set to 0).

**--vm_overcommit** *float*
Set VM overcommitment value to *float*. If set, it is used to calculate **privmmpages** parameter in case it is not set explicitly (see below). Default value is **0**, meaning unlimited privvmpages.

**vzctl** checks if running kernel is VSwap capable, and refuses to use these parameters otherwise. This behavior can be overriden by using **--force** flag before parameters.

In VSwap mode, all beancounters other than RAM and swap become optional. Note though that if some optional beancounters are not set, they are calculated and set by vzctl implicitly, using the following formulae:

• **lockedpages.barrier = oomguarpages.barrier = ram**

• **lockedpages.limit = oomguarpages.limit = unlimited**

• **vmguarpages.barrier = vmguarpages.limit = ram + swap**

• **privvmpages.barrier = privvmpages.limit = (ram + swap) * vm_overcommit**

(if **vm_overcommit** is **0** or not set, **privvmpages** is set to "unlimited")

Here is an example of setting container 777 to have 512 megabytes of RAM and 1 gigabyte of swap:

```
   vzctl set 777 --ram 512M --swap 1G --save
```

**User Beancounter limits**

The following options sets barrier and limit for various user beancounters.

Note that for VSwap-enabled kernels (version 042stab042 or greater) these limits are optional, you must only set **--ram** and **--swap** (see above). For older kernels, these limits are obligatory.

Each option requires one or two arguments. In case of one argument, **vzctl** sets barrier and limit to the same value. In case of two colon-separated arguments, the first is a barrier, and the second is a limit. Each argument is either a number, a number with a suffix, or a special value **unlimited**.

Arguments are in items, pages or bytes. Note that page size is architecture-specific, it is 4096 bytes on x86 and x86_64 platforms.

You can also specify different suffixes for User Beancounter parameters (except for those which names start with **num**). For example, **vzctl set** *CTID* **--privvmpages 5M:6M** should set **privvmpages**' barrier to 5 megabytes and its limit to 6 megabytes.

Available suffixes are:

• **T**, **t** - terabytes;
• **G**, **g** - gigabytes;
• **M**, **m** - megabytes;
• **K**, **k** - kilobytes;
• **P**, **p** - memory pages (arch-specific, usually 4KB);
• **B**, **b** - bytes.

You can also specify the literal word **unlimited** in place of a number. In that case the corresponding value will be set to **LONG_MAX**, i. e. the maximum possible value.

**--numproc** *items*[:*items*]
: Maximum number of processes and kernel-level threads. Setting the barrier and the limit to different values does not make practical sense.

**--numtcpsock** *items*[:*items*]
: Maximum number of TCP sockets. This parameter limits the number of TCP connections and, thus, the number of clients the server application can handle in parallel. Setting the barrier and the limit to different values does not make practical sense.

**--numothersock** *items*[:*items*]
: Maximum number of non-TCP sockets (local sockets, UDP and other types of sockets). Setting the barrier and the limit to different values does not make practical sense.

**--vmguarpages** *pages*[:*pages*]
Memory allocation guarantee. This parameter controls how much memory is available to a container. The barrier is the amount of memory that container's applications are guaranteed to be able to allocate. The meaning of the limit is currently unspecified; it should be set to **unlimited**.

**--kmemsize** *bytes*[:*bytes*]
Maximum amount of kernel memory used. This parameter is related to **--numproc**. Each process consumes certain amount of kernel memory - 16 KB at least, 30-50 KB typically. Very large processes may consume a bit more. It is important to have a certain safety gap between the barrier and the limit of this parameter: equal barrier and limit may lead to the situation where the kernel will need to kill container's applications to keep the **kmemsize** usage under the limit.

**--tcpsndbuf** *bytes*[:*bytes*]
Maximum size of TCP send buffers. Barrier should be not less than 64 KB, and difference between barrier and limit should be equal to or more than value of **numtcpsock** multiplied by 2.5 KB.

**--tcprcvbuf** *bytes*[:*bytes*]
Maximum size of TCP receive buffers. Barrier should be not less than 64 KB, and difference between barrier and limit should be equal to or more than value of **numtcpsock** multiplied by 2.5 KB.

**--othersockbuf** *bytes*[:*bytes*]
Maximum size of other (non-TCP) socket send buffers. If container's processes needs to send very large datagrams, the barrier should be set accordingly. Increased limit is necessary for high performance of communications through local (UNIX-domain) sockets.

**--dgramrcvbuf** *bytes*[:*bytes*]
Maximum size of other (non-TCP) socket receive buffers. If container's processes needs to receive very large datagrams, the barrier should be set accordingly. The difference between the barrier and the limit is not needed.

**--oomguarpages** *pages*[:*pages*]
Guarantees against OOM kill. Under this beancounter the kernel accounts the total amount of memory and swap space used by the container's processes. The barrier of this parameter is the out-of-memory guarantee. If the **oomguarpages** usage is below the barrier, processes of this container are guaranteed not to be killed in out-of-memory situations. The meaning of limit is currently unspecified; it should be set to **unlimited**.

**--lockedpages** *pages*[:*pages*]
Maximum number of pages acquired by **mlock**(2).

**--privvmpages** *pages*[:*pages*]
Allows controlling the amount of memory allocated by the applications. For shared (mapped as **MAP_SHARED**) pages, each container really using a memory page is charged for the fraction of the page (depending on the number of others using it). For "potentially private" pages (mapped as **MAP_PRIVATE**), container is charged either for a fraction of the size or for the full size if the allocated address space. In the latter case, the physical pages associated with the allocated address space may be in memory, in swap or not physically allocated yet.

The barrier and the limit of this parameter control the upper boundary of the total size of allocated memory. Note that this upper boundary does not guarantee that container will be able to allocate that much memory. The primary mechanism to control memory allocation is the **--vmguarpages** guarantee.

**--shmpages** *pages*[:*pages*]
Maximum IPC SHM segment size. Setting the barrier and the limit to different values does not make practical sense.

**--numfile** *items*[:*items*]
Maximum number of open files. In most cases the barrier and the limit should be set to the same value. Setting the barrier to **0** effectively disables pre-charging optimization for this beancounter in the kernel, which leads to the held value being precise but could slightly degrade file open performance.

**--numflock** *items*[:*items*]
Maximum number of file locks. Safety gap should be between barrier and limit.

**--numpty** *items*[:*items*]
Number of pseudo-terminals (PTY). Note that in OpenVZ each container can have not more than 255 PTYs. Setting the barrier and the limit to different values does not make practical sense.

**--numsiginfo** *items*[:*items*]
Number of siginfo structures. Setting the barrier and the limit to different values does not make practical sense.

**--dcachesize** *bytes*[:*bytes*]
Maximum size of filesystem-related caches, such as directory entry and inode caches. Exists as a separate parameter to impose a limit causing file operations to sense memory shortage and return an errno to applications, protecting from memory shortages during critical operations that should not fail. Safety gap should be between barrier and limit.

**--numiptent** *num*[:*num*]
Number of iptables (netfilter) entries. Setting the barrier and the limit to different values does not make practical sense.

**--physpages** *pages*[:*pages*]
On VSwap-enabled kernels, this limits the amount of physical memory (RAM) available to a container. The barrier should be set to **0**, and the limit to a total size of RAM that can be used used by a container.

For older kernels, this is an accounting-only parameter, showing the usage of RAM by this container. Barrier should be set to **0**, and limit should be set to **unlimited**.

**--swappages** *pages*[:*pages*]
For VSwap-enabled kernels (042stab042 or greater), this parameter limits the amount of swap space available to a container. The barrier should be set to **0**, and the limit to a total size of swap that can be used by a container.

For older (pre-VSwap) kernels, the limit is used to show a total amount of swap space available inside the container. The barrier of this parameter is ignored. The default value is **unlimited**, meaning total swap will be reported as 0.

## CPU fair scheduler parameters

These parameters control CPU usage by container.

**--cpuunits** *num*

CPU weight for a container. Argument is positive non-zero number, passed to and used in the kernel fair scheduler. The larger the number is, the more CPU time this container gets. Maximum value is 500000, minimal is 8. Number is relative to weights of all the other running containers. If **cpuunits** are not specified, default value of 1000 is used.

You can set CPU weight for CT0 (host system itself) as well (use **vzctl set 0 --cpuunits** *num*). Usually, OpenVZ initscript (**/etc/init.d/vz**) takes care of setting this.

**--cpulimit** *num*[**%**]

Limit of CPU usage for the container, in per cent. Note if the computer has 2 CPUs, it has total of 200% CPU time. Default CPU limit is **0** (no CPU limit).

**--cpus** *num*

sets number of CPUs available in the container.

**--cpumask** *cpus* | **auto** | **all**

Sets list of allowed CPUs for the container. Input format is a comma-separated list of decimal numbers and/or ranges. Consecutively set bits are shown as two hyphen-separated decimal numbers, the smallest and largest bit numbers set in the range. For example, if you want the container to execute on CPUs 0, 1, 2, 7, you should pass **0-2,7**. Default value is **all** (the container can execute on any CPU). If used with the **--nodemask** option, value of **auto** assigns all CPUs from the specified NUMA node to a container.

**--nodemask** *nodes* | **all**

Sets list of allowed NUMA nodes for the container. Input format is the same as for **--cpumask**. Note that **--nodemask** must be used with the **--cpumask** option.

## Memory output parameters

For VSwap-enabled kernels (042stab042 or greater), this parameter is ignored. For older kernels, it controls the output of /proc/meminfo inside a container.

**--meminfo none**

No /proc/meminfo virtualization (the same as on host system).

**--meminfo** *mode*:*value*

Configure total memory output in a container. Reported free memory is evaluated accordingly to the mode being set. Reported swap is evaluated according to the settings of **--swappages** parameter.

You can use the following modes for *mode*:

- **pages**:*value* - sets total memory in pages;
- **privvmpages**:*value* - sets total memory as **privvmpages** * *value*.

Default is **privvmpages:1**.

## Netfilter (iptables) control parameters

**--netfilter disabled|stateless|stateful|full**

Restrict access to netfilter/iptables modules for a container. This option replaces obsoleted **--iptables**.

Note that changing this parameter requires container restart, so consider using **--setmode** option.

The following arguments can be used:

- **disabled**
  no modules are allowed
- **stateless**
  all modules except NAT and conntracks are allowed (i.e. filter and mangle); this is the default
- **stateful**
  all modules except NAT are allowed
- **full** all modules are allowed

**--iptables** *name*[**,**...]

**Note** this option is obsoleted, **--netfilter** should be used instead.

Allow to use the functionality of *name* iptables module inside the container. Multiple comma-separated *name*s can be specified.

The default list of enabled iptables modules is defined by the **IPTABLES** variable in **vz.conf**(5).

You can use the following values for *name*: **iptable_filter**, **iptable_mangle**, **ipt_limit**, **ipt_multiport**, **ipt_tos**, **ipt_TOS**, **ipt_REJECT**, **ipt_TCPMSS**, **ipt_tcpmss**, **ipt_ttl**, **ipt_LOG**, **ipt_length**, **ip_conntrack**, **ip_conntrack_ftp**, **ip_conntrack_irc**, **ipt_conntrack**, **ipt_state**, **ipt_helper**, **iptable_nat**, **ip_nat_ftp**, **ip_nat_irc**, **ipt_REDIRECT**, **xt_mac**, **ipt_recent**, **ipt_owner**.

## Network devices control parameters

**--netdev_add** *name*

move network device from the host system to a specified container

**--netdev_del** *name*

delete network device from a specified container

## Disk quota parameters

**--diskquota yes** | **no**

allows to enable or disable disk quota for a container. By default, a global value (**DISK_QUOTA**) from **vz.conf**(5) is used.

Note that this parameter is ignored for **ploop** layout.

**--diskspace** *num*[:*num*]

For **simfs** layout, sets soft and hard disk quota limits. First parameter is soft limit, second is hard limit.

For **ploop** layout, initiates the procedure of resizing the ploop image file to the new size. Since there is no soft/hard limit concept in ploop, second *num*, if specified, is ignored.

By default, ploop resize is done online, i.e. on a mounted ploop. This is a preferred way of doing resize. Although, in a rare case a container was using lots of disk space and should now be resized to a much smaller size, an offline resize might be more appropriate. In this case, make sure the container is stopped and unmounted and use additional **--offline-resize** option

Note that ploop resize is NOT performed on container start, so for consistency **--diskspace** must be used together with **--save** flag.

Suffixes **G**, **M**, **K** can also be specified (see **Resource limits** section for more info on suffixes). If suffix is not specified, value is in kilobytes.

**--diskinodes** *num*[:*num*]

sets soft and hard disk quota limits, in i-nodes. First parameter is soft limit, second is hard limit.

Note that this parameter is ignored for **ploop** layout.

**--quotatime** *seconds*

sets quota grace period. Container is permitted to exceed its soft limits for the grace period, but once it has expired, the soft limit is enforced as a hard limit.

Note that this parameter is ignored for **ploop** layout.

**--quotaugidlimit** *num*

Enables or disables in-container per-user and per-group disk quotas. If the value is set to **0** or not set, disk quotas inside the container is disabled and not accounted.

For **simfs** layout containers, non-zero value sets maximum number of user/group IDs for which disk quota is accounted.

For **ploop** layout containers, any non-zero value enables disk quota inside the container; the number of user/group IDs used by disk quota is not limited by OpenVZ.

Note that enabling or disabling in-container disk quotas requires container restart, so consider using **--setmode** option.

### Capability option

**--capability** *capname*:**on**|**off**[**,**...]

Sets a capability for a container. Multiple comma-separated capabilities can be specified.

Note that setting a capability when the container is running does not take immediate effect; restart the container in order for the changes to take effect (consider using **--set-mode** option).

A container has the default set of capabilities, thus any operation on capabilities is "logical AND" with the default capability mask.

You can use the following values for *capname*: **chown**, **dac_override**, **dac_read_search**, **fowner**, **fsetid**, **kill**, **setgid**, **setuid**, **setpcap**, **linux_immutable**, **net_bind_service**, **net_broadcast**, **net_admin**, **net_raw**, **ipc_lock**, **ipc_owner**, **sys_module**, **sys_rawio**, **sys_chroot**, **sys_ptrace**, **sys_pacct**, **sys_admin**, **sys_boot**, **sys_nice**, **sys_resource**, **sys_time**, **sys_tty_config**, **mknod**, **lease**, **setveid**, **ve_admin**. For detailed description, see **capabilities**(7).

**WARNING**: setting some of those capabilities may have far reaching security implications, so do not do it unless you know what you are doing. Also note that setting **setpcap:on** for a container will most probably lead to inability to start it.

## Device access management

**--devnodes** *device*:[**r**][**w**][**q**]|**none**

Give the container an access (**r** - read, **w** - write, **q** - disk quota management, **none** - no access) to a device designated by the special file /dev/*device*. Device file is created in a container by **vzctl**. Example:

```
vzctl set 777 --devnodes sdb:rwq
```

**--devices b**|**c**:*major*:*minor*|**all**:[**r**][**w**][**q**]|**none**

Give the container an access to a **b**lock or **c**haracter device designated by its *major* and *minor* numbers. Device file have to be created manually.

## PCI device management

**--pci_add** [*domain*:]*bus*:*slot*.*func*

Give the container an access to a specified PCI device. All numbers are hexadecimal (as printed by **lspci**(8) in the first column).

**--pci_del** [*domain*:]*bus*:*slot*.*func*

Delete a PCI device from the container.

Note that **vps-pci** configuration script is executed by **vzctl** then configuring PCI devices. The script is usually located at **/usr/libexec/vzctl/scripts/**.

## Features management

**--features** *name*:**on**|**off**[**,**...]

Enable or disable a specific container feature. Known features are: **sysfs**, **nfs**, **sit**, **ipip**, **ppp**, **ipgre**, **bridge**, **nfsd**. A few features can be specified at once, comma-separated.

## Apply config

**--applyconfig** *name*

Read container parameters from the container sample configuration file `/etc/vz/conf/ve-`*name*`.conf-sample`, and apply them, if **--save** option specified save to the container config file. The following parameters are not changed: **HOSTNAME**, **IP_ADDRESS**, **OSTEMPLATE**, **VE_ROOT**, and **VE_PRIVATE**.

**--applyconfig_map** *group*

Apply container config parameters selected by *group*. Now the only possible value for *group* is **name**: to restore container name based on **NAME** variable in container configuration file.

### I/O scheduling

**--ioprio** *priority*

Assigns disk I/O priority to container. *Priority* range is **0**-**7**. The greater *priority* is, the more time for I/O activity container has. By default each container has *priority* of **4**.

**--iolimit** *limit*[**B**|**K**|**M**|**G**]

Assigns disk I/O bandwidth limit for a container. Value is either a number with an optional suffix, or a literal string **unlimited**. Value of **0** means "unlimited". By default a container has no I/O limit. Maximum allowed limit is 2 gigabytes per second; values exceeding the limit are truncated.

If no suffix is provided, the *limit* is assumed to be in megabytes per second. Available suffixes are:
• **b**, **B** -- bytes per second;
• **k**, **K** -- kilobytes per second;
• **m**, **M** -- megabytes per second (default);
• **g**, **G** -- gigabytes per second;

**--iopslimit** *iops*

Assigns IOPS limit for a container, in number of input/output operations per second. Value is a number or a literal string **unlimited**. Value of **0** means "unlimited". By default a container has no IOPS limit.

## Suspending and resuming

Checkpointing is a feature of OpenVZ kernel which allows to save a complete in-kernel state of a running container, and to restore it later.

**suspend**|**chkpnt** *CTID* [**--dumpfile** *name*]

This command suspends a container to a dump file If an option **--dumpfile** is not set, default dump file name **/vz/dump/Dump.***CTID* is used.

**resume**|**restore** *CTID* [**--dumpfile** *name*]

This command restores a container from the dump file created by the **suspend** command.

## Snapshotting

Snapshotting is a feature based on checkpointing and ploop shapshots. It allows to save a complete state of container file system. Plus, if the container is running, it's in-memory state (as in checkpointing). Note that snapshot functionality is only working for containers on ploop device.

**snapshot** *CTID* [**--id** *uuid*] [**--name** *name*] [**--description** *desc*] [**--skip-suspend**] [**--skip-config**]

Creates a container snapshot, i.e. saves the current container state, including its file system state, running processes state, and configuration file.

If a container is running, and **--skip-suspend** option is not specified, a container is checkpointed and then restored, and CT memory dump becomes the part of snapshot.

Unless **--skip-config** option is given, container configuration file is saved to the snapshot.

If *uuid* is not specified, it is auto-generated. Options **--name** and **--description** can be used to specify the snapshot name and description, respectively. Name is displayed by **snapshot-list**.

**snapshot-switch** *CTID* [**--skip-resume** | **--must-resume**] [**--skip-config**] **--id** *uuid*

Switches the container to a snapshot identified by *uuid*, restoring its file system state, configuration (if available) and its running state (if available).

**Note that the current state of a container (including its file system state and its configuration file) is lost!**

Option **--skip-resume** is used to ignore a CT memory dump file in a snapshot, as a result the container will end up being in a stopped state (same as if a snapshot has been taken with **--skip-suspend**).

If option **--must-resume** is set, absense of a memory dump is treated as an error, and the inability to restore from the memory dump is treated as an error rather than warning.

Option option **--skip-config** is used to ignore the CT configuration file in a snapshot, i.e. the current configuration file will be left as is.

**snapshot-delete** *CTID* **--id** *uuid*

Removes a specified snapshot.

**snapshot-mount** *CTID* **--id** *uuid* **--target** *directory*

Mounts a snapshot specified by *uuid* to a *directory*. Note this mount is read-only.

**snapshot-umount** *CTID* **--id** *uuid*

Unmounts a specified snapshot.

**snapshot-list** *CTID* [**-H**] [**-o** *field*[,*field*...] [**--id** *uuid*]

List container's snapshots.

You can suppress displaying header using **-H** option.

You can use the **-o** option to display only the specified *field*(s). List of available fields can be obtained using **-L** option.

## Performing container actions

**create** *CTID* [**--ostemplate** *name*] [**--config** *name*]
[**--layout simfs**|**ploop**[**:**{**expanded**|**plain**|**raw**}]] [**--diskspace** *kbytes*] [**--diskinodes** *num*]
[**--private** *path*] [**--root** *path*] [**--ipadd** *addr*] [**--hostname** *name*] [**--name** *name*]

[**--local_uid** *uid*] [**--local_gid** *gid*]

Creates a new container area. This operation should be done once, before the first start of the container.

By default, an OS template denoted by **DEF_OSTEMPLATE** parameter of **vz.conf**(5) is used to create a container. This can be overwritten by **--ostemplate** option.

By default, a new container configuration file is created from a sample configuration denoted by value of **CONFIGFILE** parameter of **vz.conf**(5). If the container configuration file already exists, it will not be modified.

The value of **CONFIGFILE** can be overwritten by using the **--config** *name* option. This option can not be used if the container configuration file already exists.

A new container can either be created using **simfs** filesystem or on a **ploop** device. The default is set by value of **VE_LAYOUT** parameter of **vz.conf**(5) and can be overwritten by **--layout** option. In case **ploop** is used, one can additionally specify ploop disk image format after a colon. Possible ploop formats are **expanded**, **plain** and **raw**. Default is **expanded**. Using value other than **expanded** is not recommended and is currently not supported.

You can use **--diskspace** and **--diskinodes** options to specify container file system size. Note that for **ploop** layout, you will not be able to change inodes value later.

If **DISKSPACE** is not specified either in the sample configuration file used for creation or in global configuration file **vz.conf**(5), **--diskspace** parameter is required for **ploop** layout.

Suffixes **G**, **M**, **K** can also be specified (see **Resource limits** section for more info on suffixes).

You can use **--root** *path* option to sets the path to the mount point for the container root directory (default is **VE_ROOT** specified in **vz.conf**(5) file). Argument can contain literal string **$VEID**, which will be substituted with the numeric CT ID.

You can use **--private** *path* option to set the path to directory in which all the files and directories specific to this very container are stored (default is **VE_PRIVATE** specified in **vz.conf**(5) file). Argument can contain literal string **$VEID**, which will be substituted with the numeric CT ID.

You can use **--ipadd** *addr* option to assign an IP address to a container. Note that this option can be used multiple times.

You can use **--hostname** *name* option to set a host name for a container.

When running with an upstream Linux Kernel that supports user namespaces (>= 3.8), the parameters **--local_uid** and **--local_gid** can be used to select which *uid* and *gid* respectively will be used as a base user in the host system. Note that user namespaces provide a 1:1 mapping between container users and host users. If these options are not specified, the values **LOCAL_UID** and **LOCAL_GID** from global configuration file **vz.conf**(5) are used. An explicit **--local_uid** value of 0 will disable user namespace support, and run the container as a privileged user. In this case, **--local_gid** is ignored.

**Warning:** use **--local_uid** and **--local_gid** with care, specially when migrating containers. In all situations, the container's files in the filesystem needs to be correctly owned by the host-side users.

**destroy** | **delete** *CTID*

Removes a container private area by deleting all files, directories and the configuration file of this container.

**start** *CTID* [**--wait**] [**--force**] [**--skip-fsck**] [**--skip-remount**]

Mounts (if necessary) and starts a container. Unless **--wait** option is specified, **vzctl** will return immediately; otherwise an attempt to wait till the default runlevel is reached will be made by **vzctl**.

Specify **--force** if you want to start a container which is disabled (see **--disabled**).

Specify **--skip-fsck** to skip fsck for ploop-based container filesystem (this option is used by vz initscript).

By default, if a container to be started happens to be already mounted, it is unmounted and mounted again. This behavior can be turned off by using **--skip-remount** flag.

Note that this command can lead to execution of **premount**, **mount** and **start** action scripts (see **ACTION SCRIPTS** below).

**stop** *CTID* [**--fast**] [**--skip-umount**]

Stops a container and unmounts it (unless **--skip-umount** is given). Normally, **halt**(8) is executed inside a container; option **--fast** makes **vzctl** use **reboot**(2) syscall instead which is faster but can lead to unclean container shutdown.

Note that **vzctl stop** is not asyncronous, in other words vzctl waits for container's init to exit (unless **--fast** is given), which can take up to a few minutes. Default wait timeout is 120 seconds; it can be changed globally, by setting **STOP_TIMEOUT** in **vz.conf**(5), or per container (**STOP_TIMEOUT** in **ctid.conf**(5), see **--stop-timeout**).

Note that this command can lead to execution of **stop**, **umount** and **postumount** action scripts (see **ACTION SCRIPTS** below).

**restart** *CTID* [**--wait**] [**--force**] [**--fast**] [**--skip-fsck**]

Restarts a container, i.e. stops it if it is running, and starts again. Accepts all the **start** and **stop** options.

Note that this command can lead to execution of some action scripts (see **ACTION SCRIPTS** below).

**status** *CTID*

Shows a container status. This is a line with five or six words, separated by spaces.

First word is literally **CTID**.

Second word is the numeric *CT ID*.

Third word is showing whether this container exists or not, it can be either **exist** or **deleted**.

Fourth word is showing the status of the container filesystem, it can be either **mounted** or **unmounted**.

Fifth word shows if the container is running, it can be either **running** or **down**.

Sixth word, if exists, is **suspended**. It appears if a dump file exists for a stopped container (see **suspend**).

This command can also be usable from scripts.

**mount** *CTID*
: Mounts container private area. Note that this command can lead to execution of **premount** and **mount** action scripts (see **ACTION SCRIPTS** below).

**umount** *CTID*
: Unmounts container private area. Note that this command can lead to execution of **umount** and **postumount** action scripts (see **ACTION SCRIPTS** below).

Note that **stop** does **umount** automatically.

**convert** *CTID* [**--layout ploop**[**:**{**expanded**|**plain**|**raw**}]]
: Convert CT private area to reside on a ploop device (available in kernel version 042stab052.8 and greater). Conversion should be performed when a container is stopped, plus disk space quota should be set.

**compact** *CTID*
: Compact container image. This only makes sense for ploop layout.

**quotaon** *CTID*
: Turn disk quota on. Not that **mount** and **start** does that automatically.

**quotaoff** *CTID*
: Turn disk quota off. Not that **umount** and **stop** does that automatically.

**quotainit** *CTID*
: Initialize disk quota (i.e. run **vzquota init**) with the parameters taken from the CT configuration file **ctid.conf**(5).

**exec** *CTID command*
: Executes *command* in a container. Environment variables are not set inside the container. Signal handlers may differ from default settings. If *command* is **-**, commands are read from stdin.

**exec2** *CTID command*
: The same as **exec**, but return code is that of *command*.

**runscript** *CTID script*

Run specified shell script in the container. Argument *script* is a file on the host system which contents is read by vzctl and executed in the context of the container. For a running container, the command jumps into the container and executes the script. For a stopped container, it enters the container, mounts container's root filesystem, executes the script, and unmounts CT root. In the latter case, the container is not really started, no file systems other than root (such as **/proc**) are mounted, no startup scripts are executed etc. Thus the environment in which the script is running is far from normal and is only usable for very basic operations.

**enter** *CTID* [**--exec** *command* [*arg* ...]]

Enters into a container (giving a container's root shell). This option is a back-door for host root only. The proper way to have CT root shell is to use **ssh**(1).

Option **--exec** is used to run *command* with arguments after entering into container. This is useful if command to be run requires a terminal (so **vzctl exec** can not be used) and for some reason you can not use **ssh**(1).

You need to log out manually from the shell to finish session (even if you specified **--exec**).

**console** *CTID* [*ttynum*]

Attach to a container console. Optional *ttynum* argument is tty number (such as **4** for **tty4**), default is **1** which is used for container's **/dev/console**.

Note the consoles are persistent, meaning that:
• it can be attached to even if the container is not running;
• there is no automatic detachment upon the container stop;
• detaching from the console leaves anything running in this console as is.

The following escape sequences are recognized by **vzctl console**. Note that these sequences are only recognized at the beginning of a line.

• **Esc** then **.** to detach from the console.

• **Esc** then **!** to kill anything running on the console (SAK). This is helpful when one expects a login prompt but there isn't one.

### Other options

**--help**

Prints help message with a brief list of possible options.

**--version**

Prints **vzctl** version.

## ACTION SCRIPTS

**vzctl** has an ability to execute user-defined scripts when a specific **vzctl** command is run for a container. The following **vzctl** commands can trigger execution of action scripts: **start**, **stop**, **restart**, **mount** and **umount**.

Action scripts are located in the **/etc/vz/conf/** directory. There are global and per-CT scripts.

Global scripts have a literal prefix of **vps.** and are executed for all containers. Per-CT scripts have a *CTID***.** numeric prefix and are executed for the given container only.

Please note scripts are executed in a host system (CT0) context, with the exception of **.start** and **.stop** scripts, which are executed in a container context.

The following action scripts are currently defined:

**vps.premount**, *CTID***.premount**
Global and per-CT mount scripts which are executed for a container before it is mounted. Scripts are executed in the host system context, while a CT is not yet mounted or running. Global script, if exists, is executed first.

**vps.mount**, *CTID***.mount**
Global and per-CT mount scripts which are executed for a container right after it is mounted. Otherwise they are the same as **.premount** scripts.

*CTID***.start**
Right after **vzctl** has started a container, it executes this script in a container context.

*CTID***.stop**
Right before **vzctl** has stopped a container, it executes this script in a container context.

**vps.umount**, *CTID***.umount**
Global and per-CT umount scripts which are executed for a container before it is unmounted. Scripts are executed in the host system context, while a CT is mounted. Global script, if exists, is executed first.

**vps.postumount**, *CTID***.postumount**
Global and per-CT umount scripts which are executed for a container right after it is unmounted. Otherwise they are the same as **.umount** scripts.

The environment passed to all the ***mount** scripts is the standard environment of the parent (i.e. **vzctl**) with two additional variables: **$VEID** and **$VE_CONFFILE**. The first one holds the ID of the container, and the second one holds the full path to the container configuration file. If the script needs to get other CT configuration parameters, such as **$VE_ROOT**, it needs to get those from global and per-CT configuration files.

Here is an example of a mount script, which makes host system's /mnt/disk available to container(s). Script name can either be **/etc/vz/conf/vps.mount** or **/etc/vz/conf/***CTID***.mount**.

```
# If one of these files does not exist then something
# is really broken
[ -f /etc/vz/vz.conf ] || exit 1
[ -f $VE_CONFFILE ] || exit 1
# Source both files. Note the order is important.
. /etc/vz/vz.conf
. $VE_CONFFILE
SRC=/mnt/disk
DST=/mnt/disk
mount -n -t simfs $SRC ${VE_ROOT}${DST} -o $SRC
```

## EXIT STATUS

Returns 0 upon success, or an appropriate error code in case of an error:

| | |
|---|---|
| 1 | Failed to set a UBC parameter |
| 2 | Failed to set a fair scheduler parameter |
| 3 | Generic system error |
| 5 | The running kernel is not an OpenVZ kernel (or some OpenVZ modules are not loaded) |
| 6 | Not enough system resources |
| 7 | **ENV_CREATE** ioctl failed |
| 8 | Command executed by **vzctl exec** returned non-zero exit code |
| 9 | Container is locked by another **vzctl** invocation |
| 10 | Global OpenVZ configuration file **vz.conf**(5) not found |
| 11 | A vzctl helper script file not found |
| 12 | Permission denied |
| 13 | Capability setting failed |
| 14 | Container configuration file **ctid.conf**(5) not found |
| 15 | Timeout on **vzctl exec** |
| 16 | Error during **vzctl suspend** |
| 17 | Error during **vzctl resume** |
| 18 | Error from **setluid()** syscall |
| 20 | Invalid command line parameter |
| 21 | Invalid value for command line parameter |
| 22 | Container root directory (**VE_ROOT**) not set |
| 23 | Container private directory (**VE_PRIVATE**) not set |
| 24 | Container template directory (**TEMPLATE**) not set |
| 28 | Not all required UBC parameters are set, unable to start container |
| 29 | OS template is not specified, unable to create container |
| 31 | Container not running |
| 32 | Container already running |
| 33 | Unable to stop container |
| 34 | Unable to add IP address to container |
| 40 | Container not mounted |
| 41 | Container already mounted |
| 43 | Container private area not found |
| 44 | Container private area already exists |

| | |
|---|---|
| 46 | Not enough disk space |
| 47 | Bad/broken container (**/sbin/init** or **/bin/sh** not found) |
| 48 | Unable to create a new container private area |
| 49 | Unable to create a new container root area |
| 50 | Unable to mount container |
| 51 | Unable to unmount container |
| 52 | Unable to delete a container |
| 53 | Container private area not exist |
| 60 | **vzquota on** failed |
| 61 | **vzquota init** failed |
| 62 | **vzquota setlimit** failed |
| 63 | Parameter **DISKSPACE** not set |
| 64 | Parameter **DISKINODES** not set |
| 65 | Error setting in-container disk quotas |
| 66 | **vzquota off** failed |
| 67 | ugid quota not initialized |
| 71 | Incorrect IP address format |
| 74 | Error changing password |
| 78 | IP address already in use |
| 79 | Container action script returned an error |
| 82 | Config file copying error |
| 86 | Error setting devices (**--devices** or **--devnodes**) |
| 89 | IP address not available |
| 91 | OS template not found |
| 99 | Ploop is not supported by either the running kernel or vzctl. |
| 100 | Unable to find container IP address |
| 104 | **VE_NETDEV** ioctl error |
| 105 | Container start disabled |
| 106 | Unable to set iptables on a running container |
| 107 | Distribution-specific configuration file not found |
| 109 | Unable to apply a config |
| 129 | Unable to set meminfo parameter |
| 130 | Error setting veth interface |
| 131 | Error setting container name |

| | |
|---|---|
| 133 | Waiting for container start failed |
| 139 | Error saving container configuration file |
| 148 | Error setting container IO parameters (ioprio) |
| 150 | Ploop image file not found |
| 151 | Error creating ploop image |
| 152 | Error mounting ploop image |
| 153 | Error unmounting ploop image |
| 154 | Error resizing ploop image |
| 155 | Error converting container to ploop layout |
| 156 | Error creating ploop snapshot |
| 157 | Error merging ploop snapshot |
| 158 | Error deleting ploop snapshot |
| 159 | Error switching  ploop snapshot |
| 166 | Error compacting ploop image |
| 167 | Error listing ploop snapsots |

## EXAMPLES

To create and start "basic" container with ID of 1000 using **centos-5** OS template and IP address of 192.168.10.200:

```
vzctl create 1000 --ostemplate centos-5 --config basic
vzctl set 1000 --ipadd 192.168.10.200 --save
vzctl start 1000
```

To set number of processes barrier/limit to 80/100, and PTY barrier/limit to 16/20 PTYs:

```
vzctl set 1000 --numproc 80:100 -t 16:20 --save
```

To execute command **ls -la** in this container:

```
vzctl exec 1000 /bin/ls -la
```

To execute command pipe **ls -l / | sort** in this container:

```
vzctl exec 1000 ’ls -l / | sort’
```

To enter this container and execute command **apt-get install vim**:

```
vzctl enter 1000 --exec apt-get install vim
```

Note that in the above example you will need to log out from the container’s shell after apt-get finishes.

To enter this container, execute command **apt-get install vim** and logout after successful installation (or stay inside the container if installation process failed) use **&&**:

```
vzctl enter 1000 --exec "apt-get install vim && logout"
```

To enter this container, execute command **apt-get install vim** and logout independently of exit code of installation process use **;**:

```
   vzctl enter 1000 --exec "apt-get install vim ; logout"
```

Note that you need to quote the command if you use **&&** or **;**.

To stop this container:

```
   vzctl stop 1000
```

To permanently remove this container:

```
   vzctl destroy 1000
```

## FILES

```
/etc/vz/vz.conf
/etc/vz/conf/CTID.conf
/etc/vz/conf/vps.{premount,mount,umount,postumount}
/etc/vz/conf/CTID.{premount,mount,start,stop,umount,postumount}
/proc/vz/veinfo
/proc/vz/vzquota
/proc/user_beancounters
/proc/bc/*
/proc/fairsched
```

## SEE ALSO

**vz.conf**(5), **ctid.conf**(5), **arpsend**(8), **vzcalc**(8), **vzcfgvalidate**(8), **vzcpucheck**(8), **vzifup-post**(8), **vzlist**(8), **vzmemcheck**(8), **vzmigrate**(8), **vzpid**(8), **vzquota**(8), **vzsplit**(8), **vzubc**(8), **http://wiki.openvz.org/UBC**.

## LICENSE

Copyright (C) 2000-2013, Parallels, Inc. Licensed under GNU GPL.

## NAME

vzlist − show various information about containers

## SYNOPSIS

**vzlist** [**-a** | **-S**] [**-n**] [**-H**] [**-t**] [**-j**] [**-o** *name*[,*name*...] | **-1**] [**-s** [-]*name*] [**-h** *pattern*] [**-N** *pattern*] [**-d** *pattern*] [*CTID* [*CTID* ...]]

**vzlist -L** | **--list**

**vzlist --help**

## DESCRIPTION

This utility is used for listing containers and their parameters. By default only running containers are listed. If one or more *CTID*s are specified, only specified containers are displayed.

For some fields that can have long values (e.g. **ip**, **hostname**, **description** or **features**), the value string is trimmed to some predefined width (in order not to break the columned layout), unless this field is the last one. So, in order to get the full non-trimmed value for such a field, put the field name last in the list of fields for **-o** option. If using **vzlist** from a script, add **-t** to disable trimming.

## OPTIONS

**-a**, **--all** List all containers.

**-S**, **--stopped**
List only not running containers (including the ones with status shown as **mounted** or **suspended**).

**-n**, **--name**
Display container names instead of hostnames.

**-H**, **--no-header**
Suppress displaying the header row. Usable for scripts.

**-t**, **--no-trim**
Suppress trimming long fields. Usable for scripts.

**-j**, **--json**
Output in JSON format. By default, all possible fields are printed.

**-o**, **--output** *field*[,*field*...]
Display only the specified *field*s (see **Possible fields** subsection below).

**-1** Synonym for **-H -octid**, i.e. only show container IDs, one per line.

**-s**, **--sort** [-]*field*
Sort by the value of *field* (possible arguments are the same as for **-o**). The **-** before the field name means sorting in the reverse order.

### Output filters

List of CTs can be further filtered using the following options. For patterns one can use the same wildcards as in shell (i.e. *****, **?**, and **[]**). Note: to avoid expansion of wildcards by the shell, one have to escape the pattern by either putting it into single quotes (like **'a*a'**) or by adding a backslash before the wildcard character (like **a*a**).

**-h**, **--hostname** *pattern*
List only containers whose hostnames matches the *pattern*.

**-N**, **--name_filter** *pattern*
List only containers whose names matches the *pattern*.

**-d**, **--description** *pattern*
List only containers with descriptions matching the *pattern*.

### Possible fields

**-L**, **--list**
list all available format specifiers that can be used for both output (**-o**, **--output**) and the sorting order (**-s**, **--sort**).

For the user beancounter fields, if suffix is not specified, current usage (a.k.a. "held") value is show. One can also use the following suffixes:

| | |
|---|---|
| **.m** | maxheld |
| **.b** | barrier |
| **.l** | limit |
| **.f** | fail counter |

For the disk quota fields, if suffix is not specified, current usage is shown. One can also use the following suffixes:

| | |
|---|---|
| **.s** | soft limit |
| **.h** | hard limit |

Note that for JSON output suffixes are not allowed.

## EXAMPLES

**vzlist -o ctid,kmemsize,kmemsize.l -s kmemsize**
Show CTIDs, kmemsize usage, and kmemsize limit for all running containers, sorted by the kmemsize usage.

## EXIT STATUS

Returns 0 upon success.

## COPYRIGHT

Copyright (C) 2000-2011, Parallels, Inc. Licensed under GNU GPL.

## NAME

vzcpucheck − show information about the CPU power and utilization.

## SYNOPSIS

**vzcpucheck** [**-v**]

## DESCRIPTION

This script outputs information on the CPU power and utilization.

## OPTIONS

**-v** Display information for each container.

## EXIT STATUS

Returns 0 upon success.

## SEE ALSO

**vzctl**(8).

## LICENSE

Copyright (C) 2000-2011, Parallels, Inc. Licensed under GNU GPL.

## NAME

vzmemcheck − show information about host memory parameters

## SYNOPSIS

**vzmemcheck** [**-v**] [**-A**]

## DESCRIPTION

This utility shows host system memory parameters:

- **low memory utilization;**
- **low memory commitment;**
- **RAM utilization;**
- **mem+swap utilization;**
- **mem+swap commitment;**
- **allocmem utilization;**
- **allocmem commitment;**
- **allocmem limit.**

## OPTIONS

**-v** Display information for each container.

**-A** Display absolute values (in megabytes). In this mode, the last two lines are the numerator and the denominator of the corresponding field.

## EXIT STATUS

Normally, exit status is 0. On error, exit status is 1.

## SEE ALSO

**vzcfgvalidate**(8), **vzubc**(8), **http://wiki.openvz.org/UBC_systemwide_configuration**.

## LICENSE

Copyright (C) 2000-2011, Parallels, Inc. Licensed under GNU GPL.

## NAME

vzcfgvalidate − validate a container configuration file

## SYNOPSIS

**vzcfgvalidate** [**-r**] [**-i**] [**-v yes**|**no**] *configfile*

## DESCRIPTION

This utility checks validity of resource control parameters in a container configuration file *configfile*. Some of the User Beancounter parameters have interdependencies, and if those are not met, configuration is inconsistent. The utility finds and reports such inconsistencies.

There are three severity levels in the output: Error, Warning, and Recommendation.

## OPTIONS

**−r** Repair mode (corrects *configfile*).

**−i** Interactive repair mode.

**-v yes**|**no**

Whether to treat *configfile* as VSwap enabled configuration. Default is auto-detect by checking if physpages.limit is not set to unlimited; this option overrides the auto-detection.

## EXIT STATUS

Normally, exit status is 0. On program execution error, exit status is 1. If the validation fails, exit status is 2.

## SEE ALSO

**ctid.conf**(5), **http://wiki.openvz.org/UBC_consistency_check**.

## LICENSE

## NAME

vzcalc − calculate resource usage of a container

## SYNOPSIS

**vzcalc** [**-v**] **CTID**

## DESCRIPTION

This utility displays the share of the host system resources a particular container is using. If the container is running, the current usage is displayed. High utilization values (>100%) mean the system is overloaded (or the container has an invalid configuration).

**Current**
Shows the amount of the resources consumed by the container at a given time.

**Promised**
Shows the resources soft limit values "promised" for a given container.

**Max** Shows the resources hard limit values "promised" for a given container.

If the **-v** option is specified, the following additional information is also displayed:

**Low Mem**
The part of memory residing at lower addresses and directly accessed by the kernel (only makes sense for 32-bit architectures).

**Total RAM**
Total memory.

**Mem+Swap**
Amount of memory available for applications (both RAM and swap space).

**Alloc Mem**
Standard memory allocations made for applications in a container. This is a more "virtual" system resource than RAM or RAM and swap.

**Num. Proc**
Number of processes.

## OPTIONS

**-v** Display additional information.

## EXIT STATUS

Normally, the exit status is 0. On error, the exit status is 1.

## LICENSE

Copyright (C) 2000-2009, Parallels, Inc. Licensed under GNU GPL.

## NAME

vzsplit − generate a sample container configuration file

## SYNOPSIS

**vzsplit** [**-n** *numve*] [**-f** *conf_name*] [**-s** *swapsize*] [**-v** **yes**|**no**]

## DESCRIPTION

The **vzsplit** utility is used to split the Hardware Node into equal parts. It generates a full set of system resource control parameters for the given number of containers. The values are calculated from the total physical memory of the Hardware Node the utility runs on, and the number of containers the Hardware Node shall be able to run even if the given number of containers consume all the resources available.

Without any option given, **vzsplit** prompts for the desired number of containers and outputs the resulting resource control parameters to stdout.

If there are not enough system resources to run the specified number of containers, an appropriate message is shown and the sample configuration file is not generated.

## OPTIONS

**-n** *numve*
: Specify the number of containers.

**-f** *conf_name*
: Specify the configuration sample name to write configuration to, instead of standard output. The file created will be named /etc/vz/conf/ve-*conf_name*.conf-sample.

**-s** *swapsize*
: Specify the swap size in Kbytes. If this option is not given, the swap size is read from **/proc/meminfo**.

**-v yes**|**no**
: Whether to generate VSwap enabled configuration. Default is auto-detect by checking if running kernel is VSwap capable; this option overrides auto-detection.

## EXIT STATUS

**vzsplit** returns 0 upon a successful execution. If anything goes wrong, it returns 1.

## SEE ALSO

**ctid.conf**(5).

## LICENSE

## NAME

vzubc − show User Beancounters in a human-readable format

## SYNOPSIS

vzubc [*option* ...] [*CTID* ...]

## DESCRIPTION

This utility aims to show current values for User Beancounter in a human-readable format. Values that are in pages are converted into bytes, then long values are converted into kilo-, mega- giga-bytes etc. For held and maxheld, it shows how close the values are to the barrier and to the limit. Zero and unlimited values are shown as -.

One or several *CTID*s can be specified to limit the output to the given containers. Each *CTID* can be either a name or a numeric ID. Note that names can only be used if there is a container on the system with that name (**vzlist -o ctid** *name* command is used for name to ID conversion). Unknown *CTID*s are ignored.

The utility can also be used from inside the container, in this case it only shows the values for that container (and it doesn't make sense to specify *CTID* argument).

## OPTIONS

**-w**, **--watch**
Watch mode: run itself under **watch**(1), redisplaying the output every 2 seconds (by default) until interrupted by Ctrl-C.

**-wd** Make **watch**(1) highlight the differences between current and previous output. Corresponds to **watch -d** option.

**-wt** Instruct **watch**(1) to not show its title (interval, command, and current time at the top of the display, as well as the following blank line). Corresponds to **watch -t** option.

**-wn** *time*
Refresh interval for **watch**(1), in seconds (corresponds to **watch -n** *time*).

**-q**, **--quiet**
Quiet mode. In this mode, **vzubc** only shows beancounters with fails and those with held/maxheld values close to limits. **-v**, **--verbose** Verbose mode. In this mode, **vzubc** also shows beancounters with barrier set to **unlimited** (those are hidden by default).

**-qh** *ratio*
Quiet threshold for held to limit ratio. Default is **0.5** (50%).

**-qm** *ratio*
Quiet threshold for maxheld to limit ratio. Default is **0.8** (80%).

**-r**, **--relative**
Relative mode: for fail counters, instead of showing the absolute value, calculate the difference from the previous run. This mode is denoted by a **+** sign before the **FAIL** column header.

**-rd** *dir* Set a directory for saving fail counters to *dir* (default is **/tmp/vzubc.store**).

**-rc** Clear all saved fail counter data.

**-i**, **--incremental**
Incremental mode. Shows an additional column with a difference in held value from the previous run. This option also affects quiet mode: all lines with changed held values are shown. Held data is saved to the same directory as fail counter data.

**-id** *dir* A synonym for **-rd**.

**-ic** Clear all saved held data.

**-c**, **--color**
Enable color highlighting. Not compatible with **--watch**. Same thresholds as for quiet mode are used to highlight "more important" lines, plus the lines with non-zero fail counters are highlighted.

**-f** | **--file** *filename*
Read User Beancounters from *filename*. By default this is **/proc/bc/resources** or, if that one is not available, **/proc/user_beancounters**. Use **-** to read from standard input.

## EXAMPLES

**vzubc 101 web dns**
Show all beancounters for CTID 101 and for CTs named **web** and **dns**.

**vzubc -w -wd -wn 10 101**
Display beancounters for CT 101 every 10 seconds, highlighting the changes. Interrupt with Control-C.

**vzubc -q -c**
Show beancounters with held or maxheld close to limits, plus those with non-zero fail counters, with colors.

**vzubc -q -qh 0.8 -qm 1**
Show beancounters with held value equal to or more than 80% of a limit, and/or with maxheld value equal or more than a limit, plus those with non-zero fail counters.

**vzubc -w -q -r**
Display beancounters with held and maxheld close to limits, plus those with increasing fail counters.

**vzubc -r -q -qh 2 -qm 2**
Only show beancounters with increased (since the last run) fail counters.

**vzubc -rc -ic**
Show all beancounters, and clear all saved data for relative and incremental mode (i.e. saved values for fail counter and held).

## FILES

```
/proc/bc/resources
/proc/user_beancounters
/tmp/vzubc.store/ubc.*
```

## EXIT STATUS

Returns 0 upon successful execution, 1 otherwise.

## SEE ALSO

**watch**(1), **vzlist**(8), **vzmemcheck**(8), **vzcfgvalidate**(8), **http://wiki.openvz.org/UBC**.

## LICENSE

## NAME

vzmigrate − migrate a container between two OpenVZ servers

## SYNOPSIS

**vzmigrate** [**-r**|**--remove-area yes**|**no**] [**--ssh=***ssh_options*] [**--ssh-mux**] [**--rsync=***rsync_options*]
[**--keep-dst**] [**--live**] [**-c**|**--compact**] [**-s**|**--snapshot**] [**--check-only**|**--dry-run**]
[**-f**|**--nodeps**[=*check*[**,***check* ...]]] [**-t**|**--times**] [**-v**] *destination_address CTID*
**vzmigrate -h**|**--help**|**--usage**

## DESCRIPTION

This utility is used to migrate a container from one (source) Hardware Node (HN) to another (destination) HN. The utility can migrate either stopped or running container. For a stopped container, simple CT private area transfer is performed (**rsync**(1) is used for file transfer). For running containers, either traditional (with CT stop and start) or live migration is possible.

This program uses **ssh** as a transport layer. You will need to put ssh public key to destination node and be able to connect to node without entering password.

## OPTIONS

**-r**, **--remove-area yes** | **no**
Whether to remove a container area on source HN for the successfully migrated container. Default is **yes**.

**--ssh=***options*
Additional *options* that will be passed to ssh while establishing connection to destination HN.

**--ssh-mux**
Enable ssh channel multiplexing, establishing and reusing a single ssh connection to remote server. This speeds up live migration.

**--rsync=***options*
Additional *options* that will be passed to **rsync**(8). You may add options like **-z** to enable data compression if you are migrating over a slow link.

**--keep-dst**
Do not clean synced destination container private area in case of some error. It makes sense to use this option on big container migration to avoid syncing container private area again in case some error (on container stop for example) occurs during first migration attempt.

**--live** Perform live migration: instead of restarting a container, checkpoint and restore are used, so there is no container downtime or service interruption. Additional steps are performed to minimize the time when a container is in suspended state. Option **--online** can be used as a backward-compatible alias.

**-c**, **--compact**
Compact a container image (i.e. run **vzctl compact**) before migration. Works for ploop only, ignored otherwise.

**-s**, **--snapshot**
Create a container snapshot (i.e. run **vzctl snapshot**) before migration. Works for ploop only, ignored otherwise.

**--check-only**, **--dry-run**
Do not perform actual migration, stop after preliminary checks. This is used to check if a CT can possibly be migrated. Combine with **--live** to enable more checks for live migration case.

**-f**|**--nodeps**[=*check*[**,***check* ...]]
Continue migration, ignoring some or all preliminary check failures. Particular checks can be ignored by providing an argument to **--nodeps** option. The following options can be used (comma-separated):
• **cpu** − ignore cpu capabilities check;
• **ipv6** − ignore ipv6 module check.

**-t**, **--times**
At the end of live migration, output various timings for migration stages that affect total suspended CT time. Note that it only makes sense with **--live**.

**-v** Verbose mode. Causes **vzmigrate** to print debugging messages about its progress. Note that **-v** automatically implies **-t**. Multiple **-v** options increase the verbosity.

**-h**|**--help**|**--usage**
Print usage info and exit.

## EXAMPLES

Migration of CT 101 to **192.168.1.130** with downtime:

```
vzmigrate 192.168.1.130 101
```

Online migration of CT 102 to **192.168.1.130**:

```
vzmigrate --live 192.168.1.130 102
```

## EXIT STATUS

**0 EXIT_OK**
Command completed successfully.

**1 EXIT_USAGE**
Bad command line options.

**2 EXIT_VE_STOPPED**
Container is stopped.

**4 EXIT_CONNECT**
Can’t connect to destination (source) HN.

**6 EXIT_COPY**
Container private area copying/moving failed.

**7 EXIT_VE_START**
Can’t start or restore destination CT.

**8 EXIT_VE_STOP**
Can’t stop or checkpoint source CT.

**9 EXIT_EXISTS**
Container already exists on destination HN.

**10 EXIT_NOTEXIST**
Container does not exists on source HN.

**12 EXIT_IP_INUSE**
You attempt to migrate CT which IP address(es) are already in use on the destination node.

**13 EXIT_QUOTA**
Operation with CT quota failed.

**14 EXIT_OVZ_NOT_RUNNING**
OpenVZ is not running, or some required kernel modules are not loaded.

**15 EXIT_APPLY_CONFIG**
Unable to set CT name on destination node.

**16 EXIT_PLOOP_UNSUP**
Ploop is not supported by destination node.

**17 EXIT_UNSUP_CPT_VER**
CPT version incompatibility with the destination node.

**18 EXIT_UNSUP_CPU**
Destination node CPU incompatibility.

**19 EXIT_CANT_READ_REMOTE_CONFIG**
Unable to read remote vz.conf.

**20 EXIT_LOCKED**
Can’t lock container (already locked).

## SEE ALSO

**rsync**(1), **vzcptcheck**(8).

## COPYRIGHT

Copyright (C) 2001-2013, Parallels, Inc. Licensed under GNU GPL.

## NAME

vzcptcheck − show/test CPT properties

## SYNOPSIS

**vzcptcheck** *property* [**argument** ...]
**vzcptcheck help**

## DESCRIPTION

This low-level utility is used by **vzmigrate**(8) to check if a container can be succesfully restored on a destination system before performing live migration. Currently, it is able to check CPT version and CPU flags compatibility.

There are two modes of operation: to show a property and to test a property. Showing mode is invoked if a property specified with no arguments; testing mode requires one or many arguments after a property name. Generally, one runs **vzcptcheck** on one node to get a property, and on the other node to check for the property.

Note that this utility does not guarantee that migration will succeed even if all tests are passed. It merely gives a way to prevent migration which will definitely fail.

## OPTIONS

**vzcptcheck version**
**vzcptcheck version** *num*
To get CPT version number, run **vzcptcheck version** on the source node. Then, to check if this version number is supported, run **vzcptcheck version** *num* on the destination node.

**vzcptcheck caps**
**vzcptcheck caps** *ctid capnum*
**vzcptcheck caps** *capnum*
To get CPU capabilities number, run **vzcptcheck caps** on the destination node. Then, to check if a running CT can be live migrated to a node with such a CPU, run **vzcptcheck caps** *ctid capnum* on the source node. To check if a suspended container can be migrated (i.e. it can be restored on the destination node), use the form without *ctid*: **vzcptcheck caps** *capnum*.

## EXIT STATUS

In showing mode, returns **0** upon success, or **1** in case of an error.

In testing mode, returns **0** if migration is possible (test passed), **1** in case of an error, or **2** if migration is not possible (test failed).

## EXAMPLES

To check if a CT 123 can be migrated to $DEST, run the following commands on the source node. Note that error checking is omitted for brevity.

```
VEID=123
VERSION=$(vzcptcheck version)
echo Got CPT version: $VERSION
ssh root@$DEST vzcptcheck version $VERSION
echo Version check result: $?
CAPS=$(ssh root@$DEST vzcptcheck caps)
echo Got caps: $CAPS
vzcptcheck caps $VEID $CAPS
echo Caps check result: $?
```

If both tests pass, you should get 0 as a result for both checks.

## SEE ALSO

**vzmigrate**(8).

## LICENSE

## NAME

vzfsync − perform fsync/fdatasync/fadvise on files

## SYNOPSIS

**vzfsync** { **-s**,**--sync** | **-d**,**--datasync** } [**-n**,**--dontneed**] **file** [**file** ...]

**vzfsync --help**

## DESCRIPTION

This utility performs **fsync**(2) or **fdatasync(2)** for every supplied file name. It can also optionally perform **posix_fadvise**(2) with the argument of **POSIX_FADV_DONTNEED**.

It is used from within **vzmigrate**(8) utility for live migration, right after copying ploop delta files (and before suspending a container) to sync those copied files to disk in order to optimize subsequent ploop mount time and, utlimately, container frozen time.

## OPTIONS

**-s**,**--sync**
Perform fsync.

**-d**,**--datasync**
Perform fdatasync.

**-n**,**--dontneed**
Perform posix_fadvise(POSIX_FADV_DONTNEED).

## EXIT STATUS

Returns 0 upon success, or an appropriate error code in case of an error:

| | |
|---|---|
| 1 | Invalid usage |
| 2 | Failed to perform some requested operations on one or many files. |

## SEE ALSO

**fsync**(2), **fdatasync**(2), **posix_fadvise**(2).

## LICENSE

Copyright (C) 2014, Parallels, Inc. Licensed under GNU GPL v2.

## NAME

vznnc − run a program connected to a socket

## SYNOPSIS

**vznnc** { **-l** | **-c** } **-p** *port* [**-f** *fd*] [--] *program* [*arg* ...]

## DESCRIPTION

This "nano-netcat" utility can be used to either listen on or connect to a TCP port at localhost, and run a specified *program* with its stdin and stdout (or a specified file descriptor) connected to the socket.

## OPTIONS

**-l** Listen on a specified TCP port at localhost.

**-c** Connect to a specified TCP port at localhost.

**-p** *port* Port number.

**-f** *fd* File descriptor ID. If this option is not set, stdin and stdout are closed and are connected to the socket, otherwise they are left intact, and the specified *fd* is used.

**--** This is a separator between **vznnc** own arguments and *program* arguments, so that latter won't be processed by **vznnc**. It is required in case there are any arguments to *program* that start with the dash (-) character, and is optional otherwise.

*program* [ *arg* ... ]
Program to run, with optional arguments.

## EXIT STATUS

Returns *program* exit status upon success, or one of the following codes in case of an error:

| | |
|---|---|
| 1 | Invalid usage |
| 127 | Error executing *program*. |
| 220 | Network-related error. |

## EXAMPLES

To run receiving side of **ploop copy** command on a remote server, using openssh port forwarding:

```
PORT=2345
ssh -L localhost:$PORT:localhost:$PORT $REMOTE_SERVER \
     vznnc -l -p $PORT -- ploop copy -d $FILE -i0 -o1
```

To do the same, but with stdin and stdout intact, using file descriptor 5 for communication:

```
ssh -L localhost:$PORT:localhost:$PORT $REMOTE_SERVER \
     vznnc -l -p $PORT -f 5 -- ploop copy -d $FILE -i5 -o5
```

## SEE ALSO

**nc**(3), **netcat**(3), **socat**(2).

## LICENSE

## NAME

vztmpl-dl − download/list/update OpenVZ templates

## SYNOPSIS

**vztmpl-dl** [ *option* ... ] *template* [*template* ...]
**vztmpl-dl** [ *option* ... ] **--update-all**
**vztmpl-dl --list-local**|**--list-remote**|**--list-all**|**--list-orphans**
**vztmpl-dl --config**|**--help**

## DESCRIPTION

The main purpose of this utility is to download a precreated container tarballs (also known as templates) from the OpenVZ download server. It can be used directly from the command line, and is also used by **vzctl create**.

Usually one or more template names are required, except if one of **--update-all**, **--list***, **--config** or **--help** is used.

## OPTIONS

**--gpg-check**
Check GPG signatures of downloaded files. By default, GPG check is performed if **CHECK_TEMPLATE_SIG** is set to **yes** in **/etc/vz/download.conf** file, **gpg**(1) tool is installed, and the OpenVZ public key is available in the gpg keyring. See **EXAMPLES** section below on how to install OpenVZ public key.

**--no-gpg-check**
Do not check GPG signatures.

**--ignore-errors**
Keep processing templates from the list and return 0 exit code, even if some template failed to download. Without this option, **vztmpl-dl** aborts upon first download failure.

**--update**
Update (re-download) existing templates.

**--no-update**
Do not try to update existing templates.

**--quiet** Be less talkative (currently it only adds --quiet flag to **wget**(1) ).

**--no-quiet**
Be more talkative (this can be used to negate the effect of **QUIET=yes** in **download.conf.**

**--update-all**
Try to update all templates that are available locally. Note that there is no need to specify individual templates. This options implicitly assumes **--update** and **--ignore-errors** options.

**--list-remote**
Output list of templates available for download. This option is also used by **vzctl**(8) bash-completion script to complete **--ostemplate** option arguments.

**--list-local**
Output list of templates available locally.

**--list-all**
Output combined list of templates.

**--list-orphans**
Output list of local templates not available remotely. In trivial scenario this will give the list of old templates that are no longer officially supported.

**--config**
Output current configuration. The tool has built-in configuration which can be overwritten by options in **/etc/vz/download.conf** file.

**--help** Output help.

## CONFIGURATION

The tool has a built-in configuration defaults, which can be changed by editing **/etc/vz/download.conf** file. The following parameters can be set:

**UPDATE_TEMPLATE**
Default is **yes**. Setting to **no** has the same effect as using **--no-update** option.

**CHECK_TEMPLATE_SIG**
Default is **no**. Setting to **yes** has the same effect as using **--gpg-check** option.

**QUIET**
Default is **no**. Setting to **yes** has the same effect as using **--quiet** option.

**TMPL_REPO_PREFIX**
URL repo prefix. Default is **http://download.openvz.org/template/precreated**. Setting this to location of your closest OpenVZ download mirror can speed up download. Current list of OpenVZ mirrors is available at **http://openvz.org/Download_mirrors**.

**TEMPLATE_REPOS**
List of URLs to get repositories from. Default is ${TMPL_REPO_PREFIX}. More URLs can be added to get access to beta or unsupported templates.

## EXIT STATUS

Returns 0 upon success, or an appropriate error code in case of an error:

| | |
|---|---|
| 1 | Download error |
| 2 | Local template file already present, not updating |
| 3 | Error in usage (no argument provided) |
| 4 | wget binary not found |
| 5 | GPG signature check failed |

## EXAMPLES

To add OpenVZ public key to gpg keyring:
```
  gpg --search-keys security@openvz.org
```

To list all available templates:
```
  vztmpl-dl --list
```

To download (or update) centos-6 templates:
```
  vztmpl-dl centos-6-x86 centos-6-x86_64
```

To use **yandex.ru** mirror (in **/etc/vz/download.conf**):
```
  TMPL_REPO_PREFIX="http://mirror.yandex.ru/mirrors/download.openvz.org/temp
```

To enable beta templates (in **/etc/vz/download.conf**):

```
  TEMPLATE_REPOS="${TEMPLATE_REPOS} ${TMPL_REPO_PREFIX}/beta/"
```

## FILES

```
/etc/vz/download.conf
/vz/template/cache
```

## SEE ALSO

**arpsend**(8), **download.conf**(5), **http://wiki.openvz.org/Package_signatures#Importing_the_public_key**.

## LICENSE

## NAME

arpsend − send ARP requests

## SYNOPSIS

**arpsend -D −e** *target_ip* [ **−e** *target_ip* ...] [**−c** *count*] [**−w** *timeout*] *interface*

**arpsend −U −i** *source_ip* [**−c** *count*] [**−w** *timeout*] *interface*

## DESCRIPTION

Utility **arpsend** sends ARP packets on device *interface* to detect or update neighbours' ARP caches with a given IP.

## COMMANDS

Run utility with one of the following commands:

**−D** Send broadcast ARP request to detect neighbours with *target_ip*. You have to specify *target_ip* (**-e** option).

**−U** Send broadcast ARP request to update neighbours' ARP caches with *source_ip*. You have to specify *source_ip* (**-i** option).

## OPTIONS

**−c count**
Number of packets to send. Default is infinity.

**−w interval**
Interval between packets, in seconds. Default is 1 second.

**−i source_ip_address**
Set source IP address field in ARP packet.

**−e target_ip_address**
Set target IP address field in ARP packet. Note that you can specify **-e** option multiple times to detect many IP addresses in one utility call.

## EXIT STATUS

**arpsend** returns 0 upon successful execution. If something goes wrong, it returns an appropriate error code.

1 EXC_USAGE
Usage error

2 EXC_SYSTEM
System error

3 EXC_RECV
ARP reply was received

## EXAMPLES

To send request on interface **eth0** to detect any neighbour computers with IP **192.168.10.200**:

```
arpsend -D -e 192.168.10.200 eth0
```

To send request on interface **eth0** to update neighbours' ARP caches with IP **192.168.10.200**:

```
arpsend -U -i 192.168.10.200 eth0
```

## NOTES

Interface you use have to be arpable and not be loopback (i.e. **/sbin/ip link show** *interface* should show neither **NOARP** nor **LOOPBACK** flags in interface parameters).

## SEE ALSO

**vzctl**(8).

## LICENSE

## NAME

ndsend − send a Neighbor Advertisement NDP packet

## SYNOPSIS

**ndsend** *address interface*

## DESCRIPTION

The **ndsend** utility is called by **arpsend**(8) for IPv6 addresses to send an unsolicited Neighbor Advertisement ICMPv6 multicast packet announcing a given IPv6 address to all IPv6 nodes as per RFC4861.

## OPTIONS

*address*

Specify the IPv6 address to be advertised.

*interface*

Specify the network interface to send an advertisement from.

## EXIT STATUS

**ndsend** returns 0 upon successful execution. If something goes wrong, it returns an appropriate error code:

**1 EXC_USAGE**

Usage error

**2 EXC_SYSTEM**

System error

## EXAMPLES

To send a Neighbor Advertisement ICMPv6 on interface **eth0** with IPv6 address **2001:DB8::1**:

```
ndsend 2001:DB8::1 eth0
```

## SEE ALSO

**arpsend**(8), **vzctl**(8).

## LICENSE

## AUTHOR

This manual page was initially written by Thorsten Schifferdecker <tsd@debian.systs.org> for the **Debian GNU/Linux** system (but may be used by others).

## NAME

vzpid − display the CT ID given the process ID

## SYNOPSIS

**vzpid** [**-p**] *pid* [...]
**vzpid** [**-p**] **-**

## DESCRIPTION

This script displays the CT ID the process with the given *pid* belongs to.

## OPTIONS

| | |
|---|---|
| *pid* | The *pid* of the process to display the CT ID of. Can be specified multiple times. |
| **-p** | Also show the corresponding in-container PID(s). |
| **-** | Read PID(s) from the standard input. |

## EXIT STATUS

Returns 0 upon a successful execution.

## SEE ALSO

**vzctl**(8).

## LICENSE

## NAME

vzifup-post − add containers' ARP records to an interface

## SYNOPSIS

vzifup-post *DEVICE*

## DESCRIPTION

This script adds the appropriate ARP records for containers' IP addresses to a network interface *DEVICE*. This needs to be done on interface start and the script is usually called from system networking scripts.

## OPTIONS

*DEVICE*

Network interface, e.g. **eth0**.

## EXIT STATUS

Returns 0 upon success.

## SEE ALSO

**vzctl**(8).

## LICENSE

## NAME

vzeventd − the OpenVZ events daemon.

## SYNOPSIS

**vzeventd** [**−v**] [**−d**]

**vzeventd −h**

## DESCRIPTION

This daemon takes care of events sent by the OpenVZ kernel (via a netlink socket) and performs required actions associated with those events, by running specific scripts. Every event received contains an event name and a container ID.

For every known event, the event script /usr/libexec/vzctl/scripts/vzevent-*event_name* is executed, with container ID being passed to the script as **VEID** environment variable. Not-existent events scripts are ignored. All unknown events are ignored but logged.

The following events are recognized:

- **start**
- **stop**
- **mount**
- **umount**
- **reboot**

The following event scripts are provided:

**vzevent-stop**
Takes care of removing ARP and routing records for the given CT from CT0.

**vzevent-reboot**
Takes care of rebooting a given CT.

## OPTIONS

**−v** Increase verbosity (can be used multiple times).

**−d** Debug mode (do not daemonize, run in foreground).

**-h** Display help and exit.

## EXIT STATUS

Returns 0 upon success.

## LICENSE

## AUTHOR

This manual page was initially written by Thorsten Schifferdecker <tsd@debian.systs.org> for the **Debian GNU/Linux** system (but may be used by others).

## NAME

ploop − ploop device management utility

## SYNOPSYS

**ploop init -s** *size* [**-f** *format*] [**-v** *version*] [**-t** *fstype*] [**-b** *blocksize*] [**-B** *fsblocksize*] [**--nolazy**] *delta_file*

**ploop mount** [**-r**] [**-F**] [**-f** *format*] [**-b** *blocksize*] [**-d** *device*] [**-m** *mount_point*] [**-o** *mount_options*] [**-t** *fstype*] *base_delta* [*... top_delta*]

**ploop mount** [**-r**] [**-F**] [**-d** *device*] [**-m** *mount_point*] [**-o** *mount_options*] [**-t** *fstype*] [**-u** *uuid*] *DiskDescriptor.xml*

**ploop umount** { **-d** *device* | **-m** *mount_point* | *DiskDescriptor.xml* | *image_file* }

**ploop replace** { **-u** *uuid* | **-l** *level* | **-o** *cur_image_file* } [**--keep-name**] **-i** *image_file* *DiskDescriptor.xml*

**ploop resize -s** *size DiskDescriptor.xml*

**ploop convert** { **-f** *format* | **-v** *version* } *DiskDescriptor.xml*

**ploop check** [**-u** *uuid*] *DiskDescriptor.xml*

**ploop check** [**--force**] [**--hard-force**] [**--check**] [**--ro**] [**--silent**] [**--drop-inuse**] [**--raw**] [**--blocksize** *size*] [**--repair-sparse**] *image_file*

**ploop info** [**-s**] [**-d**] *DiskDescriptor.xml*

**ploop list** [**-a**]

**ploop snapshot** [**-u** *uuid*] *DiskDescriptor.xml*

**ploop snapshot-merge** [**-u** *uuid* [**-U** *uuid2*] | **-A**] [**-n** *new_delta*] *DiskDescriptor.xml*

**ploop snapshot-switch -u** *uuid DiskDescriptor.xml*

**ploop snapshot-delete -u** *uuid DiskDescriptor.xml*

**ploop snapshot-list** [**-H**] [**-u** *uuid*] [**-s**] [**-o** *field*[,*field*...]] *DiskDescriptor.xml*

**ploop copy -s** *device* [**-F** *stop_command*] { [**-d** *file*] | [**-o** *output_fd*] [**-f** *feedback_fd*] }

**ploop copy -d** *file* [**-i** *input_fd*] [**-f** *feedback_fd*]

**ploop balloon discard** [**--automount**] [**--to-free** *size*] [**--min-block** *min_size*] [**--defrag**] *DiskDescriptor.xml*

**ploop restore-descriptor -f** *format* [**-b** *blocksize*] *disk_dir delta_file*

## DESCRIPTION

ploop is a kernel block device, similar to the traditional loop device (which is controlled by **losetup**(8)) but with more features added, such as dynamic disk space allocation, stackable images, online resize, snapshotting, and live migration helper (write tracker). This manual page describes the ploop user space tool which is used to perform various operations related to ploop devices and images.

Note that this ploop tool is not aware of container entities. Commands that have **DiskDescriptor.xml** as an argument work with an XML file that contains meta-information about a particular ploop device configuration: device characteristics (block size etc.), storage information (names and formats of images used for the device), snapshot information, etc. If a particular command can be used both with and without the DiskDescriptor.xml argument, it is strongly advised to use the form with DiskDescriptor.xml.

## OPTIONS

Run **ploop** without any options to show a short synopsys, including a list of commands.

Run **ploop** *command* to show synopsys and a short description for a particular *command*.

### Basic commands

#### init

Create and initalize a ploop image file and a corresponding **DiskDescriptor.xml** file.

**ploop init -s** *size* [**-f** *format*] [**-v** *version*] [**-t** *fstype*] [**-b** *blocksize*] [**-B** *fsblocksize*] [**--nolazy**] *delta_file*

**-s** *size* Image size. If no suffix is specified, the *size* is in sector units (one sector is 512 bytes). One can specify optional **K**, **M**, **G** or **T** suffix to set the size in kilo-, mega-, giga- or terabytes.

**-f** *format* Image format. See **Image formats** below.

**-v** *version*
Image version, can be **1** or **2**. Default is **2**, if supported by the kernel.

**-t none**|**ext3**|**ext4**
File system type to create, default is **ext4**. Unless **none** is specified, a partition, a filesystem, and a balloon file will be created inside the image. Using **ext3** is not recommended.

**-b** *blocksize*
Device block size, in 512 byte sectors. Default block size is 2048 sectors, or 1 megabyte.

**-B** *fsblocksize*
Filesystem block size, in bytes. Default is 4096 bytes.

**-n**, **--nolazy**
Disable lazy mkfs initialization (which is enabled by default). Currently that means if this flag is set, **mkfs.ext4**(8) is called with **-Elazy_itable_init=0,lazy_journal_init=0** options.

*delta_file* Path to a non-existent image file to be created.

#### mount

Assemble a ploop device from one or more delta images, start it, and optionally mount the file system residing on the device.

Two forms of this command are provided. The first one accepts a list of delta images to be used for assembling the ploop device, while the second one is using information from a DiskDescriptor.xml file. Please note that not all mount options are applicable to both forms.

**ploop mount** [**-r**] [**-F**] [**-f** *format*] [**-b** *blocksize*] [**-d** *device*] [**-m** *mount_point*]
[**-o** *mount_options*] [**-t** *fstype*] *base_delta* [. . . *top_delta*]

**ploop mount** [**-r**] [**-F**] [**-d** *device*] [**-m** *mount_point*] [**-o** *mount_options*] [**-t** *fstype*] [**-u** *uuid*]
*DiskDescriptor.xml*

**-r** Mount as read-only.

**-F** Run **fsck**(8) on inner filesystem before mounting it. This option is ignored if **-m** is not used.

**-f** *format* Image format. Ignored if DiskDescriptor.xml is specified. Otherwise, one need to specify **raw** as an argument, if raw image format is used.

**-b** *blocksize* Device block size, in 512 byte sectors. Ignored if DiskDescriptor.xml is specified. Otherwise, required for raw images.

**-d** *device* Ploop device to use, e.g. **/dev/ploop0**. If not specified, a randomly numbered ploop device will be used.

**-m** *mount_point*
If this option is specified, ploop goes on to mount the file system to directory denoted by *mount_point*.

**-o** *mount_options*
Any additional mount options, comma-separated. Used if **-m** is set.

**-t** *fstype* File system type used for mounting. Used if **-m** is set. The default is **ext4**.

**-u** *uuid* | **base**
GUID of the image from the DiskDescriptor.xml to be mounted. By default, top GUID is used. The special ’**base**’ value can be used to mount the base (lower-level) image.

*base_delta* [. . . *top_delta*]
List of image files to mount, with the first one being the base delta and the last one being the top delta (i.e. the one that will be writable unless **-r** is specified).

*DiskDescriptor.xml*
Path to the DiskDescriptor.xml file with information about images.

### umount

Unmount a ploop device. Since a mounted ploop device consists of an image (or multiple images), a device, and (optionally) a file system mounted to a directory, one can refer to any of the above entities to specify what to unmount. The recommended way is to use DiskDescriptor.xml.

**ploop umount** { **-d** *device* | **-m** *mount_point* | *DiskDescriptor.xml* | *image_file* }

**-d** *device*
Ploop device, e.g., **/dev/ploop0**.

**-m** *mount_point*
Mount point of a ploop device to unmount.

*DiskDescriptor.xml*
Path to the DiskDescriptor.xml file with information about images.

*image_file*
Path to a mounted image file.

## replace

Replaces a ploop image by a different (but identical) one, on a running ploop device. Only a read-only image (e.g. a non-top one in a stacked configuration) can be replaced. An image to be replaced is specified by either one of level, UUID, or the current image file.

If a new image is not identical to the old one (i.e. its content differs) or not suitable for ploop in any other way (e.g. it is sparse, or resides on a file system not supported by ploop), the result is undefined.

**ploop replace** { **-u** *uuid* | **-l** *level* | **-o** *cur_image_file* } [**--keep-name**] **-i** *image_file* *DiskDescriptor.xml*

**-u** *uuid*
A *uuid* of an image to be replaced.

**-l** *level*
A level of image to be replaced. A level is a distance of an image from the base delta.

**-o** *cur_image_file*
A current image file (the one to replace).

**-k**, **--keep-name**
A flag to keep the same image file name. If this flag is set, after the image is replaced, a new *image_file* is renamed to the old one (removing the old, now unused, image), so no modification to DiskDescriptor.xml is required.

**-i** *image_file*
A new replacement image.

*DiskDescriptor.xml*
Path to the DiskDescriptor.xml file with information about images.

## resize

Resize a ploop image. Both online (i.e. when ploop is mounted and used) and offline resize is supported, and the tool can either grow or shrink both the ploop image and the underlying file system.

**ploop resize -s** *size* *DiskDescriptor.xml*

**-s** *size* Image size. If no suffix is specified, *size* is in sector units (one sector is 512 bytes). One can specify optional **K**, **M**, **G** or **T** suffix to set the size in kilo-, mega-, giga- or terabytes.

*DiskDescriptor.xml*
Path to the DiskDescriptor.xml file with information about images.

### convert

Convert either ploop image format or version (but not both at the same time). Conversion can only be performed offline (i.e. image should not be in use).

**ploop convert** { **-f** *format* | **-v** *version* } *DiskDescriptor.xml*

**-f** *format* Image format. See **Image formats** below.

**-v** *version* Image version, can be **1** or **2**.

### check

Check the internal consistency of (and possibly repair) a ploop image (or images). Note that image(s) to be tested should not be in use.

**ploop check** [**-u** *uuid*] *DiskDescriptor.xml*

Check all the images in *DiskDescriptor.xml* up to the one denoted by the *uuid* (or default top delta, if UUID is not specified). Default built-in check options are used, and the ones specified on the command line, if any, are ignored.

**ploop check** [**--force**] [**--hard-force**] [**--check**] [**--ro**] [**--silent**] [**--drop-inuse**] [**--raw**]
[**--blocksize** *size*] [**--repair-sparse**] *DiskDescriptor.xml | image_file*

**-f**, **--force** Force check even if image's dirty flag is not set.

**-F**, **--hard-force**
Same as **-f**, plus try to fix even fatal errors (can be dangerous).

**-c**, **--check** Check for duplicated blocks and holes.

**-r**, **--ro** Read-only access, do not modify image(s).

**-s**, **--silent** Be more silent, only report errors.

**-d**, **--drop-inuse**
Drop image "in use" flag.

**-R**, **--raw** Specifies that *image_file* is a raw ploop image.

**-b**, **--blocksize** *size*
Image cluster block size, in sectors (for raw images).

**-S**, **--repair-sparse**
Repair sparse image(s).

## Miscellaneous commands

### info

**ploop info** *DiskDescriptor.xml*
When run without any options, show information about disk space and inodes usage and limits on the inner ploop filesystem, somewhat similar to **vzquota**(8) **stat** or **show** commands.

**ploop info** [**-s**] [**-d**] *DiskDescriptor.xml*
Either one or both options can be used together. Option **-s** is used to show information about ploop device size, block size, and format version. Option **-d** is used to show a corresponding ploop block device, it available. file.

### list

**ploop list** [**-a**]

Shows a list of running ploop devices (first column) and their corresponding base images. With option **-a** it also shows a mount point (third column).

### restore-descriptor

Create DiskDescriptor.xml file suitable for *delta_file* and put it into *disk_dir*.

**ploop restore-descriptor -f** *format* [**-b** *blocksize*] *disk_dir delta_file*

Read image header in case of ploop1 format or check raw image size and generate proper DiskDescriptor.xml file. You can specify blocksize for raw images. If it's not specified it will be choosen automatically **-** largest possible value between 32K and 1M. Raw image size must be aligned to blocksize.

This command works only for base images. Snapshots are not supported.

## Working with snapshots

Ploop snapshots is a mechanism for creating and managing instant states of a running file system. Creating a snapshot leads to creating a new empty ploop image which is layered on top of an old one, then all writes are ending up in the top image, and reads are falling through to a lower level. There can be up to 126) stacked ploop images (or snapshots). Online snapshot merging is also supported.

Snapshots are identified by a unique UUID. A snapshot can be mounted using **ploop mount -u** *uuid* command, see above.

**snapshot**

Create a ploop snapshot.

**ploop snapshot** [**-u** *uuid*] *DiskDescriptor.xml*

| | |
|---|---|
| **-u** *uuid* | Specify a *uuid* for a new snapshot. If option is not given, uuid is generated automatically. To generate uuid manually, one can use the **uuidgen**(1) utility. Note that UUID must be enclosed in curly brackets. |

**snapshot-merge**

Merge a snapshot with its parent. That is, contents of the delta file corresponding to the snapshot is merged to a parent delta, then the file is removed. Parent snapshot UUID is lost (as it is replaced with the *uuid* specified). All snapshots having the lost one as a parent are updated to have the *uuid* as its parent.

**ploop snapshot-merge** [**-u** *uuid* [**-U** *uuid2*] | **-A**] [**-n** *new_delta*] *DiskDescriptor.xml*

| | |
|---|---|
| **-u** *uuid* | Specify a single snapshot *uuid* to merge. If this option is not specified, the top delta will be used. |
| **-U** *uuid2* | Together with **-u** *uuid*, specify that all snapshots in the range *uuid*...*uuid2* are to be merged. |
| **-A** | Merge all snapshots down to base delta. If some snapshots have more than a single child, they will be impossible to merge. |
| **-n** *new_delta* | If this option is set, instead of merging the child delta into its parent, both the parent and the child deltas are merged into a newly created file *new_delta*, which replaces the parent delta. Both deltas are then removed. |

**snapshot-switch**

Switch to the specified snapshot. This operation can only be performed while ploop is not running (i.e. is unmounted). The current top delta image will be removed.

**ploop snapshot-switch -u** *uuid DiskDescriptor.xml*

| | |
|---|---|
| **-u** *uuid* | Specify a snapshot *uuid* to switch to. |

### snapshot-delete

Delete the specifed snapshot. This operation can only be performed if the specified snapshot is not active. In case snapshot doesn't have any children, it will simply be removed. In case snapshot has a single child, it will be merged to that child. Deleting a snapshot that has multiple children is currently not supported (but can be performed manually in an iterative fashion).

**ploop snapshot-delete -u** *uuid DiskDescriptor.xml*

**-u** *uuid* Specify a snapshot *uuid* to be deleted.

### snapshot-list

List available snapshots.

**ploop snapshot-list** [**-H**] [**-u** *uuid*] [**-s**] [**-o** *field*[,*field*...]] *DiskDescriptor.xml*

**-H**, **--no-header** Suppress displaying the header row. Usable for scripts.

**-u**, **--uuid**, **--id** *uuid*
Filter the output to a specified *uuid*.

**-s**, **--snapshot** List in terms of snapshots. By default (i.e. without this option) the command lists all deltas, and top delta is marked as current. When this option is used, top delta is not listed, and the snapshot which is the parent of top delta is marked as current.

**-o**, **--output** *field*[,*field*...]
Display only the specified *field*s. Possible fields are:

| | |
|---|---|
| • **uuid** | - snapshot's UUID; |
| • **parent_uuid** | - snapshot's parent UUID; |
| • **current** | - if this snapshot is the current one; |
| • **fname**- snapshot image file name. | |

### Image copying

**ploop copy** is a mechanism of effective copying of a top ploop image with the help of build-in ploop kernel driver feature called write tracker. Write tracker is a feature that lets **ploop copy** to iteratively obtain a list of modified image blocks from the kernel. Two **ploop copy** processes are required for iterative top delta transfer. These are used by **vzmigrate(8).**

### copy (sending)

**ploop copy -s** *device* [**-F** *stop_command*] { [**-d** *file*] | [**-o** *output_fd*] [**-f** *feedback_fd*] }

This command enables the in-kernel write tracker for the specified ploop *device*, then sends all the data blocks from the top delta image to a pipe specified by the *output_fd* argument (**stdout**, i.e. **1** by default), supposedly read by destination **ploop copy**, or a *file*. After that, it iteratively gets the list of the modified data blocks from the kernel and sends those blocks again. After a number of iterations (or when the list is empty), it executes the *stop_command* (this could be **vzctl stop** or **vzctl chkpnt**) and does the last iteration of sending the modified data blocks. Finally, it checks that the data were not modified, error is returned otherwise.

If *feedback_fd* is specified, it is used to read back from the ploop copy receiving side. The feedback channel is currently used to wait for **fdatasync**(2) completion.

## copy (receiving)

**ploop copy -d** *file* [**-i** *input_fd*] [**-f** *feedback_fd*]

Reads the data blocks (provided by the source **ploop copy**) from the file descriptor *input_fd* (**stdin**, i.e. **0** by default) and writes them to the *file*.

If *feedback_fd* is specified, it is used to send status back to the ploop copy sending side.

## Ballooning

Since there is no online shrink support in **ext4** file system, ploop internally uses a technique called "ballooning" as a work around to shrink its images.

Ballooning operation consists of inflating a special balloon file (invisible for ordinary users), loading fiemap info of the inflated balloon to the kernel, relocating blocks of the image file from the tail to the space specified by fiemap info, and truncating the tail of the image file. Result is the image file of a smaller size.

However, it is quite possible that inflated balloon file will only span blocks that were never touched before. Those will look like "not allocated" space from the kernel ploop point of view. In this case nothing will be relocated and nothing truncated.

So, if balloon operation succeeded, it's only guaranteed that a user of ploop device won't be able to consume more space than the initial block device size minus the size of the inflated balloon. On the other hand, if a user of block device used a lot of space on it, then freed the significant part of used space, balloon operation will result in significant truncate of image file.

All the ploop ballooning logic is hidden from the end user, so while a number of low-level commands exist for working with ploop ballooning, those are not needed and therefore are not documented here, except for a single command.

## balloon discard

In a situation when a lot of disk space were freed on an in-ploop filesystem, use **ploop balloon discard** to optimize the ploop image size.

**ploop balloon discard** [**--automount**] [**--to-free** *size*] [**--min-block** *min_size*] [**--defrag**] *DiskDescriptor.xml*

Iteratively try to relocate and discard unused blocks from a ploop image, reducing its size.

Note that ploop device and its inner file system should be mounted. If not, one can use **--automount** option to automatically mount ploop for the duration of the operation.

Option **--defrag** can be used to run a filesystem defragmentation utility (currently e4defrag2 on ext4 only) before the main operation.

Option **--to-free** can be used to specify a maximum disk space to be freed. In other words, stop the process once freed space exceeded requested *size*. Default is 0, meaning to try to free as much space as possible.

Option **--min-block** can be used to specify a minimum size of an extent to free. The smallest possible extent is 1 cluster (currently 1 MB), one can specify higher value to speed up the whole discarding operation.

Note that the same functionality is available by means of **vzctl compact** command.

### Image formats

The following image formats are currently supported.

**raw** Raw format, with 1:1 mapping between the image file and the ploop device.

**ploop1**, **expanded**
Expanded format. The image will grow according to the needs of the underlying file system. This format is the default. Names '**ploop1**' and '**expanded**' are aliases.

**preallocated**
This is the same as '**ploop1**' or '**expanded**', the only difference is all the file blocks are allocated during creation.

## EXIT STATUS

**ploop** exits with status 0 in case of successful execution. Any status greater than 0 signifies an error.

**1**, **SYSEXIT_CREAT**
Error creating a file.

**2**, **SYSEXIT_DEVICE**
Error getting or opening a ploop device.

**3**, **SYSEXIT_DEVIOC**
Error doing **ioctl**(2) on ploop device.

**4**, **SYSEXIT_OPEN**
Error opening a file.

**5**, **SYSEXIT_MALLOC**
Not enough memory (error from **malloc**(3), **realloc**(3), **calloc**(3), or **posix_memalign**(3)).

**6**, **SYSEXIT_READ**
Error during read.

**7**, **SYSEXIT_WRITE**
Error during write.

**9**, **SYSEXIT_SYSFS**
Error reading from a sysfs file (usually under **/sys/block/ploop...**).

**11**, **SYSEXIT_PLOOPFMT**
Corrupted ploop image detected.

**12**, **SYSEXIT_SYS**
Other system error.

**13**, **SYSEXIT_PROTOCOL**
Broken protocol (unexpected value received).

**14**, **SYSEXIT_LOOP**
**pcopy** command can't finalize copying (frozen filesystem is changing).

**15**, **SYSEXIT_FSTAT**
Error from **stat**(2), **fstat**(2), or **statfs**(2).

**16**, **SYSEXIT_FSYNC**
Error from **fsync**(2) or **syncfs**(2).

**17**, **SYSEXIT_EBUSY**
Can't continue, another operation is in progress.

**18**, **SYSEXIT_FLOCK**
Error from **flock**(2).

**19**, **SYSEXIT_FTRUNCATE**
Error from **ftruncate**(2) or **truncate**(2).

**20**, **SYSEXIT_FALLOCATE**
Error from **fallocate**(2).

**21**, **SYSEXIT_MOUNT**
Can't mount ploop image or file system.

**22**, **SYSEXIT_UMOUNT**
Can't unmount ploop image or file system.

**23**, **SYSEXIT_LOCK**
Locking failed (another operation in progress?).

**24**, **SYSEXIT_MKFS**
Can't create file system.

**26**, **SYSEXIT_RESIZE_FS**
Utility **resizefs** failed.

**27**, **SYSEXIT_MKDIR**
Error from **mkdir**(2).

**28**, **SYSEXIT_RENAME**
Error from **rename**(2).

**29**, **SYSEXIT_ABORT**
Operation aborted.

**30**, **SYSEXIT_RELOC**
Block relocation failed.

**33**, **SYSEXIT_CHANGE_GPT**
Error resizing GPT partition table.

**35**, **SYSEXIT_UNLINK**
Error from **unlink**(2).

**36**, **SYSEXIT_MKNOD**
Error from **mknod**(2).

**37**, **SYSEXIT_PLOOPINUSE**
Image is already in use.

**38**, **SYSEXIT_PARAM**
Invalid parameter.

**39**, **SYSEXIT_DISKDESCR**
Problem with DiskDescriptor.xml file.

**40**, **SYSEXIT_DEV_NOT_MOUNTED**
Ploop image is not mounted.

**41**, **SYSEXIT_FSCK**
Error from **fsck**(8).

**43**, **SYSEXIT_NOSNAP**
Can’t find specified snapshot UUID.

## SEE ALSO

**vzctl**(8), **vzmigrate**(8), **http://openvz.org/Ploop**.

## NAME

vzdqcheck − count disk usage

## SYNOPSIS

**vzdqcheck** [*option* ...] *path*

## DESCRIPTION

**vzdqcheck** scans all the files under the *path*, using the same code as **vzquota init**, and reports disk usage (in 1 KB blocks and inodes).

## OPTIONS

**-h** Print usage information.

**-V** Print utility version.

**-q** Quiet mode. Causes all warning and diagnostic messages to be suppressed. Only fatal errors are displayed.

**-v** Verbose mode. Causes the utility to print debugging messages about its progress. Multiple **-v** options increase verbosity. Maximum is 2.

## SEE ALSO

**vzquota**(8).

## COPYRIGHT

Copyright (C) 2000-2011, Parallels, Inc. Licensed under GNU GPL.

## NAME

vzdqdump, vzdqload − dump, load user/group quotas

## SYNOPSIS

**vzdqdump** [*general_options*] *quota_id* [**-f**] [**-c** *quota_file*] [**-G**] [**-U**] [**-T**] [**-F**]
**vzdqload** [*general_options*] *quota_id* [**-c** *quota_file*] [**-G**] [**-U**] [**-T**] [**-F**]

## DESCRIPTION

**vzdqdump** dumps user/group quota information obtained either from a quota file or the kernel to stdout.

**vzdqload** loads user/group quota information provided by **vzdqdump** from stdin into quota file. Quota must be stopped at load.

The *quota_id* must be numeric-only identifier. Note that quota ID is not the same as container ID (CTID). One container can mount several filesystems and each of them can have its own quotas.

## OPTIONS

### General

**-h** Print usage information.

**-V** Print utility version.

**-q** Quiet mode. Causes all warning and diagnostic messages to be suppressed. Only fatal errors are displayed.

**-v** Verbose mode. Causes the utilities to print debugging messages about their progress. Multiple **-v** options increase verbosity. Maximum is 2.

### Parameters

**-f** Dump user/group quota information from kernel rather than quota file.

**-c** *quota_file*
Specifies quota file to process.

**-G**, **--grace**
Dump (load) user/group grace times.

**-U**, **--limits**
Dump (load) disk limits of users/groups.

**-T**, **--exptimes**
Dump (load) expiration times of users/groups.

**-F**, --first
Dump (load) first level quota.

## EXIT STATUS

See **vzquota**(8).

## SEE ALSO

**vzquota**(8).

## COPYRIGHT

## NAME

vzquota − manipulate containers disk quotas

## SYNOPSIS

**vzquota** [*quota_options*] *command quota_id* [*command_options*]

## DESCRIPTION

**vzquota** controls disk quotas for Virtuozzo/OpenVZ container. These are per-container disk quotas set from Virtuozzo/OpenVZ host system.

The *command* can be one of the following: **init**, **drop**, **on**, **off**, **setlimit**, **setlimit2**, **reload2**, **stat**, **show**.

The *quota_id* must be numeric-only identifier. Note, that quota ID is not the same as container ID (CTID). One container can mount several filesystems and each of them can have its own quotas.

## OPTIONS

### General

**-h** Print usage information.

**-V** Print utility version.

**-q** Quiet mode. Causes all warning and diagnostic messages to be suppressed. Only fatal errors are displayed.

**-v** Verbose mode. Causes **vzquota** to print debugging messages about its progress. Multiple **-v** options increase verbosity. Maximum is 2.

**-b** Batch mode. in this mode outputs (usually on **stat** and **show** commands) will be in format better suitable for parsing by a script.

### Quota Commands

The following commands are available:

**init** A necessary preliminary for any other quota command work: create a new quota file, calculating current disk usage from given path. This command requires full set of quota soft- and hardlimits given as command-line options. Limits are also stored in quota file, so subsequent **vzquota on** doesn't requires any quota limit as command-line parameter, although accepts them as well. New specified limits and flags will also be stored in the quota file.

You can also create your own quota files for arbitrary quota accounting points. Quota file location and path to quota accounting point can be specified via **-c** and **-p** options (see below). You can use also **-R** option instead of **-c** option, to set relative quota file location. In this case quota file resides one dirrectory upper than quota accounting point and has special name (see below).

**drop** Remove quota file. Command checks if quota is running and refuses to remove file in this case, option **-f** allows to override that rule.

**on** Turn quota on. If previous quota session wasn't switched off properly (quota is not running, but quota file indicates it is), initialization procedure will be performed. **-f** option allow to force initialization procedure regardless of the shutdown status. Command **on** doesn't work in case specified quota id is running.

**off** Turn quota off, write usage statistic back to the quota file. Doesn't work if quota file cannot be accessed, also accepts **-f** option (force switching off, even if usage statistic will be lost). This is possible that quota will still be in a stopped state, even if **-f** flag is used.

**setlimit**
Set new quota parameters. Requires at least one quota parameter or flag specified. Applies new parameters immediately if quota with given quota_id is running. Stores new limits and flags in the quota file. Option **-f** specifies to mark quota as dirty, so at the next quota start, disk will be rescanned and usage updated.

**setlimit2**
Set second-level quota parameters. Applies new parameters immediately if quota with given quota_id is running and second-level quota is on. Stores new limits in the quota file.

**reload2**
Reload second-level quota limits from quota file for given quota_id.

**stat** Show usage statistics and update it in quota file. Option **-f** causes to do not read and update quota file, just print statistics from kernel. Option **-t** specifies to show and update user/group based quota statistics for a container. Works on running containers only. The command with **-t** option flushes all quota statistics from kernel to file and thus may be used for backup purposes.

**show** Show usage and limits info from quota file. Option **-t** specifies to show user/group quota information as well.

## Setting quota limits

All these options are required in **init** command, and optionally accepted in **on** and **setlimit** commands.

**−s**, **--sub−quotas 1**|**0**
Enables or disables user/group based quota inside the container. Here **1** means to enable, and **0 -** to disable. By default user/group quota is disabled. This option is accepted by **init**, **on** and **setlimit** commands.

**−u** *user_id*
For **setlimit2** command only. Limits will be applied to the specified *user_id*.

**−g** *group_id*
For **setlimit2** command only. Limits will be applied to the specified *group_id*.

**−u**, **--ugid−limit** *limit*
For **on** and **setlimit** commands only. Specifies maximum number of user and group IDs allowed in the container. If the value is **0**, user/group quota will not be accounted. Default value is *0*. There is one note concerning **setlimit** command. If first-level quota is running, second-level quota is active and not all ugid objects were loaded into kernel by **on** command due to insufficient *ugid_limit* value (this can be checked by issuing **stat -t** command and observing whether ugid limit was exceeded), then **setlimit** with new *limit* value updates it in kernel and file but this change does not take immediate effect. Modification will be applied after quota restart.

**−b**, **--block−softlimit** *bsl*
: Disk quota block soft limit. Soft limit is amount of blocks which excess is allowed in time equal *exptime*. On the expiration of this time soft limit becomes hard limit. Block limits are set in 1k sized blocks.

**−B**, **--block−hardlimit** *bhl*
: Disk quota block hard limit. Hard limit is amount of blocks which excess is not allowed.

**−e**, **--block−exptime** *bet*
: Disk quota expiration time for excess of a block soft limit. Time can be given in two different formats:
1. *dd:hh:mm:ss*
For instance: *30* - 30 seconds; *12:00* - 12 minutes; *20:15:11:00* - 20 days, 15 hours, 11 minutes
2. *xxA*, where *A* - h/H(hour); d/D(day); w/W(week); m/M(month); y/Y(year).
For instance: *7D* - 7 days; *01w* - 1 week; *3m* - 3 months

**−i**, **--inode−softlimit** *isl*
: Disk quota inode soft limit. Similarly to block soft limit.

**−I**, **--inode−hardlimit** *ihl*
: Disk quota inode hard limit.

**−n**, **--inode−exptime** *iet*
: Disk quota expiration time for excess of a inode soft limit.

## Other options

**−p** *path*
: Point of quota accounting for given *quota_id*. This option required for **init** command and can be used also with any another command to override quota path obtained from quota file. For **on** and **setlimit** commands, this option can be used to set and save new quota accounting path for given *quota_id*

**−R**
: Set special relative path to *quota_file*. If this option is specified, quota file location will depends on path to quota accounting dir: if your quota accounting path is /path/to/*somewhere*/ than quota file will be /path/to/quota.*somewhere*. If this option is not specified, quota file location is /var/vzquota/quota.*quota_id*. All commands accept this option.

**−c** *quota_file*
: This option allows to specify a *quota_file* to work with. All commands accept this option. If this option is not specified, default file location depends on whether **-R** option is specified or not (see above).

**−f**
: Force option. Accepted by **drop**, **on**, **off**, **stat**, **setlimit** and **setlimit2** commands. Action of this option differs for different commands and is described above for each command separately.

**−t**
: For **stat** and **show** commands only. Processes user/group quota statistics. Specifies whether to show (update in file) user/group quota information.

**−t**
: For **setlimit2** command. Set second-level quota time grace parameters.

## LIMITATIONS

It is impossible to start or stop quota accounting if the directory given by **-p** option is busy. This is rather limitation of kernel part of disk quota implementation.

## DISPLAY

**vzquota stat** and **vzquota show** display the following information:

**resource**
Either 1k-blocks or inodes.

**usage** Current usage of resource.

**softlimit**
Resource limit. Current usage can exceed this limit up to hard limit during grace time.

**hardlimit**
Resource limit. Current usage can't exceed this limit.

**grace** During this amount of time usage can exceed softlimit. If a soft limit has not been exceeded the grace column is blank. If the grace period has expired, the grace column contain special **none** value.

In case option **−t** is specified, the following information is also displayed:

**User/group quota:**
Status of the 2nd level quota. This can be **on** or **off**, **active** or **inactive**. Values **on**/**off** define the state of the 2nd level quota at the next start of container quota. Values **active**/**inactive** indicate the current state of the 2nd level quota in the kernel.

**Ugids:** Three values are shown: *loaded*, *total* and *limit*. *loaded* is the number of records (uids or gids) in the kernel. *total* is the number of unique records located in the kernel and quota file. *limit* is the current kernel limit of records amount. Note that *loaded* and *total* may be greater then *limit*.

**Ugid limit was exceeded:**
Can be **yes** or **no**. **yes** indicates that vzquota did not loaded all records into the kernel. In this case you should reduce the number of unique records (remove files which belong to unnecessary users) or increase the ugid limit. After that you should restart quota.

**User/group grace times and quotafile flags:**
Grace times and quota file flags (internal parameters of standard linux kernel quota v.3).

## EXIT STATUS

| | |
|---|---|
| **0** | Command executed successfully |
| **1** | System error |
| **2** | Usage error |
| **3** | Virtuozzo syscall error |
| **4** | Quota file error |
| **5** | Quota is already running |
| **6** | Quota is not running |
| **7** | Can not get lock on this quota id |

| | |
|---|---|
| **8** | Directory tree crosses mount points |
| **9** | Quota is running but user/group quota is inactive; this status is returned by **stat -t** command for information purposes and does not indicate a error |
| **10** | Quota is marked as dirty in file; this status is returned by **show** command for information purposes and does not indicate a error |
| **11** | Quota file does not exist |
| **12** | Internal error |
| **13** | Can't obtain mount point |

## COPYRIGHT

## NAME

vzstats − report OpenVZ usage statistics

## SYNOPSIS

**vzstats**
**vzstats --view**|**--status**
**vzstats --enable**|**--disable**
**vzstats --help**

## DESCRIPTION

Utility **vzstats**, installed on an OpenVZ server, periodically collects some usage statistics and sends it to the OpenVZ stats server, **http://stats.openvz.org**, for processing and analysis. Aggregated statistics from all reports are available through the web interface.

Statistics reported include some basic hardware information (CPU, RAM, file system and disk space usage for **/vz**), software information (distribution version, kernel version, OpenVZ software versions), and some OpenVZ-specific information (number of containers running/total, number of containers using **vswap** and **ploop**, used OS templates etc.). Information submission is anonymous; information that can be used to directly identify a particular host (i.e. MAC or IP addresses, hostnames etc.) is not being reported.

For the server to anonymously identify reports from the same client, a unique random ID (UUID v4) is requested from the server during the first **vzstats** run. It is then saved locally and used when sending reports.

If run without any options and unless disabled, **vzstats** runs report scripts found in **REPDIR** (defined in **vzstats.conf**, default is **@REPDIR@**), collects their output and sends it, as a tarball, to stats server.

## OPTIONS

In case any option is provided, no report is being send, instead an action denoted by the option is performed.

The following options are supported:

**--view** Collect but do not send report, giving a user an ability to see what exact information would be send.

**--config**
Output current **vzstats** configuration, including:
• current version number;
• status (enabled or disabled);
• an URL used to submit reports to;
• vzstats UUID;
• report scripts directory.

**--disable**
Disable reporting. This is performed by creating a marker file, **@ETCDIR@/vzstats-disable**. **vzstats** checks this file upon start, and exits if it is present.

**--enable**
Re-enable reporting.

**--help** Show usage.

## FILES

```
@ETCDIR@/vzstats.conf
@ETCDIR@/vzstats-disable
@REPDIR@/*
```

## SEE ALSO

**vzlist**(8), **http://stats.openvz.org/**, **http://openvz.org/vzstats**.

## LICENSE

## NAME

vz.conf − global OpenVZ configuration file

## SYNOPSIS

**/etc/vz/vz.conf**

## DESCRIPTION

This is the global configuration file for OpenVZ. It consists of lines in the form

**PARAMETER**="*value*"

All parameter names and values are case-sensitive. Quotes surrounding *value* are required if value contains spaces, and are optional otherwise. Extra spaces are not allowed. All unrecognized lines will be ignored.

### Global parameters

**VIRTUOZZO=yes|no**

This parameter can be set to **yes** or **no**, and used by the **vz** init script. In case it is not set to **yes**, nothing will be done to boot up OpenVZ on this node.

**LOCKDIR**=*directory*

Set the directory to put lock files to.

**VE0CPUUNITS**=*number*

Value of this parameter sets **cpuunits** for CT0 (host system).

**LOGGING=yes|no**

Enables or disables logging. This parameter can be set to **yes** or **no**, default is **yes**.

**LOGFILE**=*file*

Set location of log file.

**LOG_LEVEL**=*number*

Set the logging level for the log file (does not affect console output). The greater the *number* is, the more information will be logged to the **LOGFILE**. Default is **0**, which means to log normal messages and errors. If set to **-1**, only errors will be logged.

**VERBOSE**=*number*

Set the logging level for console/terminal output (does not affect log file). Default is **0**, which means to log normal messages and errors. Increasing the **number** makes **vzctl**(8) more verbose.

**MODULES_DISABLED=yes|no**

If the value of this parameter is set to **yes**, no attempt to load kernel modules is made by the **vz** initscript. This is helpful on systems which have OpenVZ-specific features compiled into the kernel (i. e. not as modules).

**IPTABLES_MODULES**="*module module ...*"

List of iptables kernel modules to be loaded by **vz** initscript before loading OpenVZ modules (which is required for iptables to work inside containers).

**VZFASTBOOT=yes|no**

If the value of this parameter is set to **yes**, **vz** initscript called with **start** argument will start the containers with uncleanly shutdown quota state without performing quota reinitialization (which is usually a time-consuming process). After all the containers are started, the initscript when restarts those containers with unclean quota in a normal way (to recalculate/fix quotas).

**VE_STOP_MODE**=**suspend**|**stop**

If the value of this parameter is set to **suspend** or not set at all, **vz** initscript called with **stop** argument will try to suspend the running containers, instead of stopping them. Suspended containers when will be restored on **vz start**. This feature usually helps to decrease the reboot time. If a container fails to suspend, it will be stopped anyway.

**VE_PARALLEL**=*number*

A number of containers to be started or stopped simultaneously on node startup or shutdown. If not specified, the number is calculated based on amount of CPU cores. Used by the **vz** initscript.

**TEMPLATE**=*directory*

Value of this parameter is a directory in which all container template data are stored.

**SKIP_SYSCTL_SETUP**=**yes**

If this parameter is not present, **init.d/vz start** sets some **sysctl.conf**(5) parameters required or recommended by OpenVZ (with the help of **vz-postinstall sysctl** script), and then adds this parameter to **vz.conf**.

## Network interface parameters

**VE_ROUTE_SRC_DEV**="*device*"

This parameter specifies the network device name which IP address will be used as the source IP. This is helpful in case more than one network interface is configured on HN and there is a need to specify the source IP address. Default is the first device in the network device list.

**NEIGHBOUR_DEVS**="**all**|**detect**|**list:***dev1* [*dev2* ...]"

Controls on which interfaces to add/remove ARP records for a container IP, also which interfaces to use to query/announce ARP.

If set to **all**, an empty string, or unset, all possible network interfaces (i.e. the ones that are UP and doesn't have NOARP, SLAVE, or LOOPBACK flags) are used. This is the default mode.

If set to **detect**, the right network interface (i.e. the one which is in the same subnet as a container IP) will be chosen automatically.

If the value starts with **list:** prefix, the space-separated list of interfaces following the prefix is used.

**ERROR_ON_ARPFAIL**=**yes**|**no**

In case the value of this parameter is set to **yes**, vzctl will fail to start a container if there is another host with the same IP present in the network. Any other value makes vzctl to only print the warning (which is the default behavior).

**SKIP_ARPDETECT**=**yes**|**no**

In case the value of this parameter is set to **yes**, vzctl will not use ARP queries when starting a container in order to detect if there is another host with the same IP present in the network. Note that for each IP, vzctl sends ARP query and waits up to 1 second for response. Any other value means to do ARP detection.

**FORCE_ROUTE**=**yes**|**no**
: In case the value of this parameter is set to **yes**, vzctl will attempt to change an existing route to point to a container when starting a container if new route setup fails. Any other value means to only attempt to add new routes (default behavior).

### Defaults for vzctl create

These parameters are defaults for **vzctl create** and can be overwritten by its appropriate command line options.

**DEF_OSTEMPLATE**=*name*
: Default OS template to create a container from. Corresponds to **--ostemplate** option of **vzctl create**.

**CONFIGFILE**=*name*
: Default configuration file (**/etc/vz/conf/ve-***name***.conf-sample**) used to create a new container. Corresponds to **--config** option of **vzctl create**.

**VE_LAYOUT**=**simfs**|**ploop**[**:**{**plain**|**expanded**|**raw**}]
: Default CT filesystem layout for a new container, can either be **ploop** or **simfs**. In case **ploop** is used, one can additionally specify ploop disk image format after a colon. Possible ploop formats are **expanded**, **plain** and **raw**. Default is **expanded**. Using **raw** is not recommended and is not supported.

: Corresponds to **--layout** option of **vzctl create**.

### Defaults for containers

Below parameters are defaults for containers, and can be overwritten by parameters in **ctid.conf**(5) per-container configuration file.

**DISK_QUOTA**=**yes**|**no**
: In case the value of this parameter is set to **no**, all disk quota operations are disabled.

**VE_ROOT**=*directory*
: Value of this parameter is the *directory* which serves as container root mount point. Value must contain literal string **$VEID**, which will be substituted with the actual numeric CT ID.

**VE_PRIVATE**=*directory*
: Value of this parameter is the *directory* in which all the files and directories specific to that container are stored. Value must contain literal string **$VEID**, which will be substituted with the actual numeric CT ID.

**STOP_TIMEOUT**="*number*"
: Time to wait for a container to shut down on **vzctl stop**, before forcibly killing it, in seconds. Hardcoded to 120 if not set.

**NAMESERVER**="*ip* [*ip* ...]"
: Default value for containers nameserver(s). Several name server addresses are divided by spaces. If set to **inherit**, values from host system's **/etc/resolv.conf** are used.

**SEARCHDOMAIN**="*domain* [*domain* ...]"
: Default value for containers search domains. Several search domains are divided by spaces. If set to **inherit**, values from host system's **/etc/resolv.conf** are used.

Most of the other parameters that appear in per-container configuration files **ctid.conf**(5) can be also set here. Still, it is recommended to keep **TEMPLATE**, **VE_PRIVATE** and **VE_ROOT** in this configuration file, and all the other container related parameters in per-container configuration files.

## SEE ALSO

**vzctl**(8), **ctid.conf**(5).

## LICENSE

## NAME

*ctid*.conf − configuration file for an OpenVZ container.

## SYNOPSIS

**/etc/vz/conf/***CTID***.conf**

## DESCRIPTION

This is a configuration file for a container. It is stored as **/etc/vz/conf/***CTID***.conf**, where *CTID* is the numeric ID of the given container.

Configuration file consists of lines in the form

**PARAMETER**="*value*"

All parameter names and values are case-sensitive. Quotes surrounding a *value* are required if value contains spaces, and are optional otherwise. Extra spaces are not allowed. All unrecognized lines will be ignored.

The meaning of most parameters are described in **vzctl**(8), so here only the appropriate **vzctl set** option names are given.

### Miscellaneous parameters

**NAME**="*vename*"
: Corresponds to the **--name** option.

**DESCRIPTION**="*string*"
: Corresponds to the **--description** option.

**ONBOOT**="**yes**|**no**"
: Specifies whether this container will be started during system boot. Corresponds to the **--onboot** option.

**BOOTORDER**="*number*"
: Specifies the CT boot order priority. Corresponds to the **--bootorder** option.

**OSTEMPLATE**="*tmpl_name*"
: Corresponds to the **--ostemplate** option.

**VE_ROOT**="*directory*"
: Corresponds to the **--root** option.

**VE_PRIVATE**="*directory*"
: Corresponds to the **--private** option.

**MOUNT_OPTS**="*option*[,*option...*]"
: Corresponds to **--mount_opts** option.

**DISABLED**="**yes**|**no**"
: Corresponds to the **--disabled** option.

**ORIGIN_SAMPLE**="*name*"
: Name of container sample configuration which the container is based on.

**STOP_TIMEOUT**="*number*"
: Corresponds to the **--stop-timeout** option.

**Resource management parameters**

**NUMPROC**
Corresponds to the **--numproc** option.

**NUMFILE**
Corresponds to the **--numfile** option.

**NUMFLOCK**
Corresponds to the **--numflock** option.

**NUMPTY**
Corresponds to the **--numpty** option.

**NUMSIGINFO**
Corresponds to the **--numsiginfo** option.

**NUMTCPSOCK**
Corresponds to the **--numtcpsock** option.

**NUMOTHERSOCK**
Corresponds to the **--numothersock** option.

**PRIVVMPAGES**
Corresponds to the **--privvmpages** option.

**VMMGUARPAGES**
Corresponds to the **--vmguarpages** option.

**OOMGUARPAGES**
Corresponds to the **--oomguarpages** option.

**LOCKEDPAGES**
Corresponds to the **--lockedpages** option.

**SHMPAGES**
Corresponds to the **--shmpages** option.

**KMEMSIZE**
Corresponds to the **--kmemsize** option.

**TCPSNDBUF**
Corresponds to the **--tcpsndbuf** option.

**TCPRCVBUF**
Corresponds to the **--tcprcvbuf** option.

**OTHERSOCKBUF**
Corresponds to the **--othersockbuf** option.

**DGRAMRCVBUF**
Corresponds to the **--dgramrcvbuf** option.

**DCACHESIZE**
Corresponds to the **--dcachesize** option.

**NUMIPTENT**
Corresponds to the **--numiptent** option.

**PHYSPAGES**
Corresponds to the **--physpages** or **--ram** option.

**SWAPPAGES**
Corresponds to the **--swappages** or **--swap** option.

**VM_OVERCOMMIT**
Corresponds to the **--vm_overcommit** option.

**CPUUNITS**
Corresponds to the **--cpuunits** option.

**DISK_QUOTA**="**yes**|**no**"
Corresponds to the **--diskquota** option. If this is set to **no**, disk quota is not set up for this CT.

**DISKSPACE**="*softlimit*[:*hardlimit*]"
Corresponds to the **--diskspace** option.

**DISKINODES**="*softlimit*[:*hardlimit*]"
Corresponds to the **--diskinodes** option.

**QUOTATIME**="*seconds*"
Corresponds to the **--quotatime** option.

**QUOTAUGIDLIMIT**="*num*"
Corresponds to the **--quotaugidlimit** option.

**CAPABILITY**="*capname*:**on**|**off** [...]"
Corresponds to the **--capability** option.

### Network related parameters

**IP_ADDRESS**="*address* [*address* ...]"
Specifies the *address* the container will be assigned. Several addresses are divided by spaces. Corresponds to the **--ipadd** option.

**HOSTNAME**="*name*"
Corresponds to the **--hostname** option.

**NAMESERVER**="*ip* [*ip* ...]"
Corresponds to the **--nameserver** option. Several name server addresses are divided by spaces. If set to **inherit**, values from host system's **/etc/resolv.conf** are used.

**SEARCHDOMAIN**="*domain* [*domain* ...]"
Corresponds to the **--searchdomain** option. Several search domains are divided by spaces. If set to **inherit**, values from host system's **/etc/resolv.conf** are used.

## SEE ALSO

**vzctl**(8), **vzcfgvalidate**(8), **vz.conf**(5), **http://wiki.openvz.org/UBC**.

## LICENSE