



Virtuozzo Hybrid Server 7

User's Guide

April 2, 2025

Virtuozzo International GmbH

Vordergasse 59

8200 Schaffhausen

Switzerland

Tel: + 41 52 632 0411

Fax: + 41 52 672 2010

<https://virtuozzo.com>

Copyright ©2016-2025 Virtuozzo International GmbH. All rights reserved.

This product is protected by United States and international copyright laws. The product's underlying technology, patents, and trademarks are listed at <https://www.virtuozzo.com/legal.html>.

Microsoft, Windows, Windows Server, Windows NT, Windows Vista, and MS-DOS are registered trademarks of Microsoft Corporation.

Apple, Mac, the Mac logo, Mac OS, iPad, iPhone, iPod touch, FaceTime HD camera and iSight are trademarks of Apple Inc., registered in the US and other countries.

Linux is a registered trademark of Linus Torvalds. All other marks and names mentioned herein may be trademarks of their respective owners.

Contents

1. Learning the Basics	1
1.1 Virtuozzo Hybrid Server Overview	1
1.2 Differences between Virtuozzo Hybrid Server and OpenVZ	2
1.3 OS Virtualization Layer	2
1.3.1 Basics of OS Virtualization	2
1.3.2 Virtuozzo Containers	3
1.3.2.1 Virtuozzo Container Hardware	5
1.3.3 Memory and IOPS Deduplication	5
1.3.4 Templates	6
1.4 Hardware Virtualization Layer	7
1.4.1 Hardware Virtualization Basics	7
1.4.2 Virtuozzo Virtual Machines	8
1.4.2.1 Intel Virtualization Technology Support	8
1.4.3 Virtual Machine Hardware	8
1.4.4 Virtual Machine Files	10
1.4.5 Support of Virtual and Real Media	10
1.4.5.1 Supported Types of Hard Disks	11
1.4.5.2 Virtual Hard Disks	11
1.4.5.3 CD/DVD Disc Images	11
1.5 Virtuozzo Hybrid Server Configuration	11
1.6 Resource Management	11
1.7 Understanding Licensing	12
1.8 Physical Server Availability Considerations	12
2. Managing Virtual Machines and Containers	14
2.1 Creating Virtual Machines and Containers	14
2.1.1 Choosing OS EZ Templates for Containers	14

2.1.1.1	Listing OS EZ Templates	14
2.1.1.2	Installing and Caching OS EZ Templates	15
2.1.2	Creating Containers	16
2.1.2.1	Prerequisites for Creating Red Hat Enterprise Linux Containers	17
2.1.3	Creating Virtual Machines	18
2.1.4	Supported Guest Operating Systems	18
2.2	Performing Initial Configuration of Virtual Machines and Containers	19
2.2.1	Using cloud-init for Virtual Machine Guest Initialization	19
2.2.2	Installing Virtio Guest Tools	19
2.2.3	Configuring Network Settings	23
2.2.4	Setting Passwords for Virtual Machines and Containers	23
2.2.5	Setting Startup Parameters	24
2.3	Starting, Stopping, Restarting, and Querying Status of Virtual Machines and Containers	25
2.3.1	Starting Virtual Machines and Containers	25
2.3.2	Stopping Virtual Machines and Containers	25
2.3.3	Restarting Virtual Machines and Containers	25
2.3.4	Checking Status of Virtual Machines and Containers	26
2.4	Listing Virtual Machines and Containers	26
2.5	Cloning Virtual Machines and Containers	27
2.5.1	Creating Linked Clones	28
2.5.2	Configuring Default Directories	28
2.6	Suspending Virtual Machines and Containers	29
2.7	Running Commands in Virtual Machines and Containers	30
2.8	Deleting Virtual Machines and Containers	30
2.9	Viewing Detailed Information about Virtual Machines and Containers	31
2.10	Managing Virtual Machine and Container Backups	32
2.10.1	Creating Virtual Machine and Container Backups	33
2.10.1.1	Improving Backup Performance	35
2.10.2	Listing Virtual Machine and Container Backups	35
2.10.3	Restoring Virtual Machines and Containers from Backups	37
2.10.4	Deleting Virtual Machine and Container Backups	39
2.10.5	Backing Up Entire Servers	40
2.10.6	Attaching Backups to Virtual Machines and Containers	40
2.10.6.1	Attaching Backups to Linux Virtual Machines	41
2.10.6.2	Attaching Backups to Windows Virtual Machines	43

2.10.6.3	Attaching Backups to Linux Containers	43
2.10.7	Detaching Backups from Virtual Machines and Containers	44
2.10.8	Best Practices for Handling Page Cache When Using Third-Party Backup Software	44
2.10.8.1	Performing Backups from the Page Cache Angle	44
2.10.8.2	Limiting Memory Pressure from Node Processes	46
2.11	Managing Templates	48
2.11.1	Creating Templates	48
2.11.2	Listing Templates	49
2.11.3	Deploying Templates	49
2.11.4	Migrating Templates	50
2.11.5	Storing Templates on Virtuozzo Storage	50
2.12	Managing Snapshots	51
2.12.1	Creating Snapshots	52
2.12.1.1	Creating Virtual Machine Snapshots	52
2.12.1.2	Creating Container Snapshots	52
2.12.1.3	Snapshot Branching	53
2.12.1.4	Restrictions and Recommendations	53
2.12.2	Listing Snapshots	54
2.12.3	Reverting to Snapshots	55
2.12.4	Deleting Snapshots	55
2.13	Migrating Virtual Machines and Containers	55
2.13.1	Offline Migration of Virtual Machines and Containers	56
2.13.2	Live Migration of Virtual Machines and Containers	57
2.13.3	Migrating VMs and Containers from Virtuozzo 6 to Virtuozzo Hybrid Server 7	59
2.14	Performing Container-Specific Operations	62
2.14.1	Reinstalling Containers	62
2.14.1.1	Customizing Container Reinstallation	62
2.14.2	Repairing Containers	64
2.14.3	Enabling VPN for Containers	65
2.14.4	Setting Up NFS Server in Containers	65
2.14.5	Mounting NFS Shares on Container Start	66
2.14.6	Managing Container Virtual Disks	66
2.14.6.1	Adding Virtual Disks to Containers	66
2.14.6.2	Configuring Container Virtual Disks	69
2.14.6.3	Deleting Virtual Disks from Containers	69

2.14.7	Restarting Containers	70
2.14.8	Creating SimFS-based Containers	70
2.14.9	Bind-Mounting Host Directories Inside Containers	71
2.14.10	Checking Consistency of Container File System	71
2.15	Performing Virtual Machine-Specific Operations	72
2.15.1	Pausing Virtual Machines	72
2.15.2	Managing Virtual Machine Devices	73
2.15.2.1	Adding New Devices	73
2.15.2.2	Initializing Newly Added Disks	75
2.15.2.3	Configuring Virtual Devices	77
2.15.2.4	Deleting Devices	79
2.15.3	Making Screenshots	80
2.15.4	Assigning Host Devices to Virtual Machines	81
2.15.4.1	Assigning USB Devices to Virtual Machines	81
2.15.4.2	Assigning PCI Devices to Virtual Machines	82
2.15.4.3	Using PCI I/O Virtualization	84
2.15.5	Configuring IP Address Ranges for Host-Only Networks	85
2.15.6	Configuring Virtual Machine Crash Mode	85
2.15.7	Enabling Secure Boot for Virtual Machines	86
2.16	Converting Third-Party Virtual Machines	88
2.17	Converting Containers with almaconvert8	88
2.18	Converting Containers to Virtual Machines with the c2v-convert Tool	90
2.18.1	Operation	92
2.18.1.1	Conversion Time	92
2.18.1.2	Conversion Fallback	92
2.18.1.3	Interactive and Non-interactive Modes	93
2.18.2	Example 1: Converting a Single Container in Interactive Mode	93
2.18.3	Example 2: Converting Multiple Containers in Interactive Batch Mode	94
2.18.3.1	Logging in Batch Mode	96
2.18.3.2	Conversion in Batch Mode Using GNU Screen	96
2.18.4	Example 3: Converting Multiple Containers in Non-Interactive Batch Mode	96
2.18.4.1	Batch File	97
2.18.4.2	Example: Converting Containers in Unattended Mode Using Batch File	97
2.18.5	Troubleshooting	99
2.18.5.1	Running the Tool in Verbose Mode	99

2.18.5.2	Falling Back to Converted Container in Case of Unsuccessful Conversion . . .	100
2.18.5.3	Locale-Related Errors on macOS	100
2.18.5.4	Unmounted Disk After Conversion	101
2.18.6	Usage Data Collection	101
3.	Managing Resources	103
3.1	Managing CPU Resources	103
3.1.1	Configuring CPU Units	104
3.1.2	Configuring CPU Affinity for Virtual Machines and Containers	104
3.1.3	Configuring CPU Limits for Virtual Machines and Containers	105
3.1.3.1	Using <code>-cpulimit</code> to Set CPU Limits	105
3.1.3.2	Using <code>-cpus</code> to Set CPU Limits	106
3.1.3.3	Using <code>-cpulimit</code> and <code>-cpus</code> Simultaneously	106
3.1.3.4	CPU Limit Specifics	106
3.1.4	Binding CPUs to NUMA Nodes	107
3.1.5	Enabling CPU Hotplug for Virtual Machines	108
3.1.6	Configuring CPU Topology for Virtual Machines	109
3.2	Managing Disk Quotas	110
3.3	Managing Virtual Disks	110
3.3.1	Resizing Virtual Disks	110
3.3.1.1	Resizing Virtual Disks Offline	111
3.3.1.2	Resizing Virtual Disks Online	111
3.3.1.3	Checking the Minimum Disk Capacity	111
3.3.2	Compacting Disks	112
3.4	Managing Network Accounting and Bandwidth	112
3.4.1	Network Traffic Parameters	112
3.4.2	Configuring Network Classes	113
3.4.3	Viewing Network Traffic Statistics	115
3.4.4	Configuring Traffic Shaping	116
3.4.4.1	Setting the <code>BANDWIDTH</code> Parameter	117
3.4.4.2	Setting the <code>TOTALRATE</code> Parameter	117
3.4.4.3	Setting the <code>RATEMPU</code> Parameter	118
3.4.4.4	Setting the <code>RATE</code> and <code>RATEBOUND</code> Parameters	118
3.4.4.5	Traffic Shaping Example	119
3.5	Managing Disk I/O Parameters	120
3.5.1	Configuring Priority Levels for Virtual Machines and Containers	120

3.5.2	Limiting Disk I/O Bandwidth	121
3.5.3	Limiting the Number of I/O Operations per Second	122
3.5.3.1	Setting the Direct Access Flag Inside Containers	123
3.5.4	Viewing Disk I/O Statistics	123
3.5.5	Setting I/O Limits for Backup and Migration Operations	124
3.5.6	Improving Disk I/O Performance for Virtual Machines	124
3.6	Managing Container Memory Parameters	125
3.6.1	Configuring Main VSwap Parameters	125
3.6.2	Configuring Container Memory Guarantees	126
3.6.3	Configuring Container Memory Allocation Limit	127
3.6.4	Configuring Container OOM Killer Behavior	127
3.6.5	Tuning VSwap	128
3.7	Managing Virtual Machine Memory Parameters	128
3.7.1	Configuring Virtual Machine Memory Size	129
3.7.2	Configuring Virtual Machine Video Memory Size	129
3.7.3	Enabling Virtual Machine Memory Hotplugging	130
3.7.4	Configuring Virtual Machine Memory Guarantees	130
3.8	Managing Container Resource Configuration	131
3.8.1	Splitting Server into Equal Pieces	132
3.8.2	Applying New Configuration Samples to Containers	132
3.9	Managing Virtual Machine Configuration Samples	133
3.9.1	Creating a Configuration Sample	134
3.9.2	Applying Configuration Samples to Virtual Machines	134
3.9.3	Parameters Applied from Configuration Samples	134
3.10	Monitoring Resources	135
4.	Managing Services and Processes	137
4.1	What Are Services and Processes	137
4.2	Main Operations on Services and Processes	138
4.3	Managing Processes and Services	139
4.3.1	Viewing Active Processes and Services	139
4.3.2	Monitoring Processes in Real Time	141
4.3.3	Determining Container UUIDs by Process IDs	143
5.	Managing Network	144
5.1	Managing Network Adapters on the Hardware Node	144

5.2	Networking Modes in Virtuozzo Hybrid Server	145
5.2.1	Container Network Modes	145
5.2.1.1	Host-Routed Mode for Containers	145
5.2.1.2	Bridged Mode for Containers	147
5.2.2	Virtual Machine Network Modes	149
5.2.2.1	Bridged Mode for Virtual Machines	149
5.2.2.2	Host-Routed Mode for Virtual Machines	151
5.2.3	Differences Between Host-Routed and Bridged Network Modes	152
5.3	Configuring Virtual Machines and Containers in Host-Routed Mode	153
5.3.1	Setting IP Addresses	153
5.3.2	Setting DNS Server Addresses	154
5.3.3	Setting DNS Search Domains	154
5.3.3.1	Switching Virtual Machine Adapters to Host-Routed Mode	154
5.4	Configuring Virtual Machines and Containers in Bridged Mode	155
5.4.1	Managing Virtual Networks	155
5.4.1.1	Creating Virtual Networks	155
5.4.1.2	Creating Network Bridges for Network Adapters	156
5.4.1.3	Configuring Virtual Network Parameters	160
5.4.1.4	Listing Virtual Networks	161
5.4.1.5	Connecting Virtual Networks to Adapters	161
5.4.1.6	Deleting Virtual Networks	162
5.4.2	Managing Virtual Network Adapters in Virtual Environments	163
5.4.2.1	Creating and Deleting Virtual Adapters	163
5.4.2.2	Configuring Virtual Adapter Parameters	163
5.4.2.3	Connecting Virtual Environments to Virtual Networks	165
6.	Managing Licenses	166
6.1	Installing the License	166
6.1.1	Using a Proxy Server to Activate Licenses	166
6.1.2	Installing the License from Product Keys, Activation Codes, or License Files	167
6.2	Updating the License	167
6.2.1	Switching License to a New HWID	168
6.3	Transferring the License to Another Server	168
6.4	Viewing the License	169
6.4.1	License Statuses	170

7. Keeping Your System Up To Date	172
7.1 Updating Overview	173
7.2 Using A Proxy Server to Install Updates	174
7.3 Updating Automatically	174
7.3.1 Enabling Automatic Updates for Upgraded Nodes	176
7.3.2 Changing Node Update Policy	176
7.4 Updating Automatically with Smart Updates	177
7.4.1 Enabling Smart Updates	179
7.4.2 Checking Node Status	181
7.4.3 Setting the Maintenance Window and Schedule	182
7.4.4 Setting the Repositories	183
7.4.5 Excluding RPM Packages and ReadyKernel Patches from Updates	184
7.4.6 Changing Node Update Policies	184
7.4.7 Suspending Updates for All or Specific Nodes	185
7.4.8 Disabling Smart Updates	186
7.5 Updating Manually	186
7.6 Updating Nodes in Virtuozzo Storage Clusters	187
7.7 Updating the Kernel	188
7.8 Updating the Kernel with ReadyKernel	188
7.8.1 Installing ReadyKernel Patches Automatically	188
7.8.2 Managing ReadyKernel Patches Manually	189
7.8.2.1 Downloading, Installing, and Loading ReadyKernel Patches	189
7.8.2.2 Loading and Unloading ReadyKernel Patches	189
7.8.2.3 Installing and Removing ReadyKernel Patches for Specific Kernels	190
7.8.2.4 Downgrading ReadyKernel Patches	190
7.8.3 Disabling Loading of ReadyKernel Patches on Boot	190
7.8.4 Managing ReadyKernel Logs	191
7.9 Updating Software in Virtual Machines	191
7.9.1 Updating the KVM/QEMU Hypervisor in Virtual Machines	191
7.9.1.1 Updating the KVM/QEMU Hypervisor Manually	192
7.9.2 Updating Virtuozzo Guest Tools in Virtual Machines	193
7.10 Updating Containers	194
7.10.1 Updating EZ Templates	195
7.10.2 Updating EZ Template Caches	195
7.10.3 Updating EZ Templates in Existing Containers	196

7.11	Downgrading to Older Versions	197
8.	Managing High Availability Clusters	198
8.1	Prerequisites for High Availability	199
8.2	Enabling and Disabling High Availability for Nodes	200
8.2.1	Disabling High Availability for Specific Virtual Machines and Containers	201
8.2.2	Enabling High Availability for iSCSI Targets	201
8.2.3	Disabling High Availability on Nodes	202
8.3	Configuring Resource Relocation Modes	202
8.4	Configuring Resource Relocation Modes for Nodes Participating in S3 or iSCSI Export	204
8.5	Configuring HA Priority for Virtual Machines and Containers	205
8.6	Managing CPU Pools	206
8.6.1	Adding Nodes to CPU Pools	207
8.6.2	Monitoring CPU Pools	208
8.6.3	Removing Nodes from CPU Pools	209
8.7	Monitoring Cluster Status	209
8.8	Managing Cluster Resources with Scripts	211
8.9	Troubleshooting Shaman Resources	212
8.9.1	Possible Issues	212
8.9.2	Shaman Resource Consistency Best Practices	213
9.	Hardening Virtuozzo Hybrid Server	215
9.1	Update Policy	215
9.2	Audit Policy	215
9.2.1	Storing Logs Remotely	216
9.2.2	Viewing Critical Audit Messages	216
9.3	Mount Policy	216
9.4	Service Policy	217
9.5	Account Policy	218
9.6	Networking Policy	219
10.	Monitoring	220
10.1	Monitoring System Objects via SNMP	220
10.1.1	Enabling SNMP Access	220
10.1.2	Accessing System Objects via SNMP	221
10.1.3	Description of System Objects	223
10.2	Monitoring Nodes and Virtual Environments via Prometheus	225

10.2.1	Installing the Exporters	226
10.2.2	Configuring Prometheus	227
10.2.3	Configuring Grafana	229
10.2.4	Supported Alerts	230
10.3	Monitoring Nodes and Virtual Environments via Zabbix	234
10.3.1	Installing the Zabbix Agent	235
10.3.2	Configuring Zabbix Server	236
10.3.3	Supported Triggers	237
10.3.4	Managing Disk I/O Parameters	238
11.	Customer Experience Program	240
11.1	Participating in Customer Experience Program	240
12.	Advanced Tasks	242
12.1	Configuring Automatic Memory Management Policies	242
12.1.1	Restarting VCMMD	244
12.1.2	Optimizing Virtual Machine Memory Usage with Kernel Same-Page Merging	244
12.1.3	Managing Host Services with VCMMD	245
12.1.3.1	Adjusting the Page Cache Limit for user.slice	248
12.1.4	Managing Virtuozzo Storage Services with VCMMD	248
12.2	Managing Dynamic Mitigation of Intel CPU Vulnerabilities	251
12.3	Creating Customized Containers	252
12.3.1	Using Golden Images	252
12.3.1.1	Disabling Golden Images	253
12.3.2	Using Custom EZ Templates	254
12.3.2.1	Migrating EZ Templates	255
12.3.2.2	EZ Template Configuration Files	256
12.3.3	Creating Customized EZ Template RPMs	257
12.4	Installing Applications to Containers from EZ Templates	257
12.4.1	cPanel	258
12.4.2	ISPssystem ISPmanager	258
12.4.3	Plesk	259
12.4.4	JBMC Software DirectAdmin	259
12.5	Migrating PowerPanel Container from CentOS 7 to VzLinux 7	260
12.5.1	Prerequisites	260
12.5.2	Converting PowerPanel Container	261

12.5.3	Manual Reversion to Backup Container	263
12.6	Creating Virtual Environments with virt-install	264
12.7	Setting Up Docker and Kubernetes in Virtuozzo Containers	266
12.8	Managing Container Virtual Hard Disk Encryption	272
12.8.1	Setting Up Encryption Key Requester	273
12.8.2	Encrypting and Decrypting Container Virtual Hard Disks	273
12.8.3	Encrypting System Swap	275
12.9	Connecting to Virtual Machines and Containers via VNC	276
12.9.1	Securing VNC Connections with SSL	276
12.9.1.1	Using a Certificate Chain to Encrypt VNC Connections	277
12.9.1.2	Securing Previously Enabled VNC Connections	277
12.9.2	Enabling VNC Access to Virtual Machines and Containers	278
12.9.3	Connecting with a VNC Client	278
12.10	Managing iptables Modules	280
12.10.1	Using iptables Modules in Virtuozzo Hybrid Server	280
12.10.2	Using iptables Modules in Containers	281
12.10.2.1	Configuring iptables Modules	281
12.10.2.2	Using conntrack Rules and NAT Tables	281
12.11	Using SCTP in Containers and Virtual Machines	282
12.12	Creating Configuration Files for New Linux Distributions	282
12.13	Aligning Disks and Partitions in Virtual Machines	283
12.13.1	Aligning Partitions	285
12.13.2	Checking Partition Alignment in Existing Virtual Machines	285
12.13.2.1	Linux Virtual Machines	285
12.13.2.2	Windows Virtual Machines	286
12.13.3	Aligning Disks for Linux Virtual Machines	287
12.13.4	Aligning Partitions for Windows Virtual Machines	288
12.13.5	Creating a Template of a Virtual Machine with Aligned Partitions	288
12.14	Uninstalling Virtuozzo Guest Tools from Virtual Machines	289
12.14.1	Uninstalling Guest Tools from Linux Virtual Machines	289
12.14.2	Uninstalling Guest Tools from Windows Virtual Machines	290
12.15	Enabling Legacy VM Debug Mode	292
12.16	Installing Optional Packages	293
12.17	Enabling Nested Virtualization in Virtual Machines	293
12.18	Changing Server-wide Backup Configuration	294

12.19	Customizing the Message of the Day (MOTD)	295
12.19.1	Setting a Custom Static MOTD	295
12.19.2	Setting a Custom Dynamic MOTD	296
12.20	Setting Up RSA Authentication Between Nodes	297
12.21	Switching Ploop I/O Engines	299
12.22	Managing Windows Containers	300
12.22.1	Setting Up the Environment	300
12.22.1.1	Configuring the Virtuozzo Hybrid Server Node	301
12.22.1.2	Configuring Microsoft Windows Nodes	302
12.22.2	Creating Windows Containers	304
12.22.3	Starting, Listing, and Stopping Windows Containers	308
12.22.4	Running Commands in Windows Containers	309
12.22.5	Configuring Windows Container Parameters	309
12.22.6	Migrating Windows Containers	310
12.22.7	Managing Windows Container Backups	311
12.22.8	Troubleshooting Windows Containers	312
12.22.9	Uninstalling the Windows Containers Feature	312
13.	Troubleshooting	314
13.1	General Considerations	314
13.2	Kernel Troubleshooting	316
13.2.1	Using ALT+SYSRQ Keyboard Sequences	316
13.2.2	Saving Kernel Faults (OOPS)	317
13.2.3	Finding a Kernel Function That Caused the D Process State	318
13.3	Container Management Issues	319
13.3.1	Failure to Start a Container	319
13.3.2	Failure to Access a Container from Network	320
13.3.3	Failure to Log In to a Container	320
13.4	Getting Technical Support	321

CHAPTER 1

Learning the Basics

This chapter provides a brief description of Virtuozzo Hybrid Server, virtual machines and containers, their specifications, and underlying technologies.

1.1 Virtuozzo Hybrid Server Overview

Virtuozzo Hybrid Server is a bare-metal virtualization solution that includes container virtualization, KVM-based virtual machines, software-defined storage along with enterprise features and production support. It runs on top of VzLinux, a RHEL-based Linux distribution.

Virtuozzo Hybrid Server provides the best value for cost-conscious organizations enabling them to:

- standardize server hardware platforms
- effectively consolidate server resources
- consolidate and support legacy operating systems and applications
- streamline server and application deployment, maintenance, and management
- simplify software testing and development
- optimize server and application availability

1.2 Differences between Virtuozzo Hybrid Server and OpenVZ

OpenVZ is a free, open-source virtualization solution available under GNU GPL. OpenVZ is the base for Virtuozzo Hybrid Server, the commercial solution that builds on OpenVZ and offers additional benefits to customers.

Compared to OpenVZ, Virtuozzo Hybrid Server has the following extra features and differences:

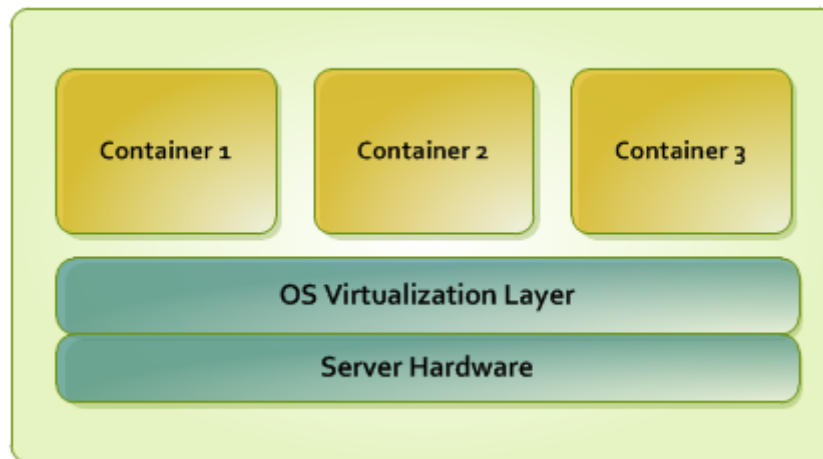
- Container and virtual machine backups
- Software-defined storage
- [ReadyKernel](#)
- Additional memory policies for vcmmd (see *Configuring Automatic Memory Management Policies* on page 242)
- Technical support
- Different installer
- Different default package set

1.3 OS Virtualization Layer

This section provides detailed information on the OS virtualization layer responsible for providing support for Virtuozzo containers.

1.3.1 Basics of OS Virtualization

The OS virtualization allows you to virtualize physical servers on the operating system (kernel) layer. The diagram below shows the basic architecture of OS virtualization.



The OS virtualization layer ensures isolation and security of resources between different containers. The virtualization layer makes each container appear as a standalone server. Finally, the container itself houses its own applications and workload. OS virtualization is streamlined for the best performance, management, and efficiency. Its main advantages are the following:

- Containers perform at levels consistent with native servers. Containers have no virtualized hardware and use native hardware and software drivers.
- Each container can seamlessly scale up to the resources of an entire physical server.
- OS virtualization technology provides the highest density available from a virtualization solution. You can create and run hundreds of containers on a standard production physical server.
- Containers use a single OS, making it extremely simple to maintain and update across containers. Applications may also be deployed as a single instance.

1.3.2 Virtuozzo Containers

From the point of view of applications and container users, each container is an independent system. This independence is provided by the Virtuozzo Hybrid Server OS virtualization layer. Note that only a negligible part of the CPU resources is spent on virtualization. The main features of the virtualization layer implemented in Virtuozzo Hybrid Server are the following:

- A container looks like a normal Linux system. It has standard startup scripts; software from vendors can run inside containers without any modifications or adjustment.
- A user can change any configuration file and install additional software inside containers.
- Containers are fully isolated from each other (file system, processes, sysctl variables).

- Containers share dynamic libraries, which greatly saves memory.
- Processes belonging to a container are scheduled for execution on all available CPUs. Consequently, containers are not bound to only one CPU and can use all available CPU power.

The two key parts of any container are the contents and configuration. By default, all container files are stored in the `/vz/private/<UUID>` directory on the hardware node, also called private area.

File Name	Description
<code>/vz/private/<UUID></code>	Container private area.
<code>/vz/private/<UUID>/root.hdd/root.hdd</code>	Virtual hard disk with container contents. The maximum size of the virtual hard disk is 50 TB.
<code>/vz/root/<UUID></code>	Container mount point.
<code>ve.conf</code>	Container configuration file: <ul style="list-style-type: none"> • Is symlinked to <code>/etc/vz/conf/<UUID>.conf</code> • Defines container parameters, such as allocated resource limits, IP address and hostname, and so on. • Overrides matching parameters in the global configuration file.

All container files are stored in a single image (`/vz/private/<UUID>/root.hdd/root.hdd`), similar to a virtual machine's hard disk. Such standalone nature:

- Enables easier migrations and backups, because sequential I/O access to container images is faster than to separate container files.
- Removes the need for OS and application templates once a container is created.
- Allows the use of native Linux disk quotas that are journaled and does not require quota recalculation after disasters like server crashes.

Note: Containers that store all data in an image file (container-in-an-image-file layout) can only be placed on `/vz` partitions formatted as `ext4`.

1.3.2.1 Virtuozzo Container Hardware

A container may have the following virtual hardware:

Hardware	Theoretical	Certified
CPU	Up to the total number of threads on the host	Up to 64
RAM	Up to the total amount of physical RAM on the host	Up to 1 TB
Disk drives	Up to 15 hard disk drives mapped to ploop image files and DVD drives mapped to ISO image files, up to 50 TB each	
Network Interfaces	Up to 15	

1.3.3 Memory and IOPS Deduplication

Virtuozzo Hybrid Server provides memory and IOPS deduplication that helps save memory and IOPS on the server and increases the maximum number of running containers per server.

Deduplication is provided by Virtuozzo Hybrid Server File Cache which includes the `pfcached` daemon and a ploop image mounted to a directory on the server. The file cache ploop contains copies of eligible files located inside containers. To be eligible for caching, files in containers must meet certain configurable requirements, e.g., be read in a certain number of containers, be of certain size, be stored in certain directories in containers.

When the kernel gets a request to read a file located in a container ploop, it searches the file cache ploop for a copy of that file by the SHA1 hash stored as file's extended attribute. If successful, the copy in the file cache ploop is read instead of the original file in the container ploop. Otherwise, the original file in the container ploop is read.

To populate the file cache ploop with most requested files, `pfcached` periodically obtains container files read statistics from the kernel, analyzes it, and copies eligible files to the file cache ploop. If the file cache ploop is running out of space, the least recently used files are removed from it.

Virtuozzo Hybrid Server File Cache offers the following benefits:

- Memory deduplication. Only a single file from the file cache ploop needs to be loaded to memory instead of loading multiple identical files located in multiple containers.

- IOPS deduplication. Only a single file from the file cache ploop needs to be read instead of reading multiple identical files located in multiple containers.

If the physical server has storage drives of various performance, e.g., IDE and SSD, the file cache ploop performs better if located on the fastest storage drive on the node, e.g., SSD. In any case:

- If the server memory is not overcommitted, the file cache mostly helps speed up container start during which most files are read. In this case caches residing in memory are not cleaned often, so copies in the file cache ploop, once read during container start, do not need to be reread often during container operation.
- If the server memory is overcommitted, Virtuozzo Hybrid Server File Cache helps speed up both container start and operation. In this case, caches residing in memory may be cleaned often, so files in the file cache ploop need to be reread as often.

Virtuozzo Hybrid Server File Cache can be managed with the `pfcache` utility.

1.3.4 Templates

A template (or a package set) in Virtuozzo Hybrid Server is a set of original application files repackaged for use by Virtuozzo Hybrid Server. Usually, it is just a set of RPM packages for Red Hat like systems. Virtuozzo Hybrid Server provides tools for creating templates, installing, upgrading, adding them to and removing them from a container.

Using templates lets you:

- Share RAM among similar applications running in different containers to save hundreds of megabytes of memory.
- Deploy applications simultaneously in many containers.
- Use different versions of an application in different containers (for example, perform upgrades only in certain containers).

There are two types of templates: OS and application.

- An OS template is an operating system and the standard set of applications to be found right after the installation. Virtuozzo Hybrid Server uses OS templates to create new containers with a preinstalled operating system.
- An application template is a set of repackaged software packages optionally accompanied with configuration scripts. Application templates are used to add extra software to existing containers.

For example, you can create a container on the basis of the redhat OS template and add the MySQL application to it with the help of the `mysql` template.

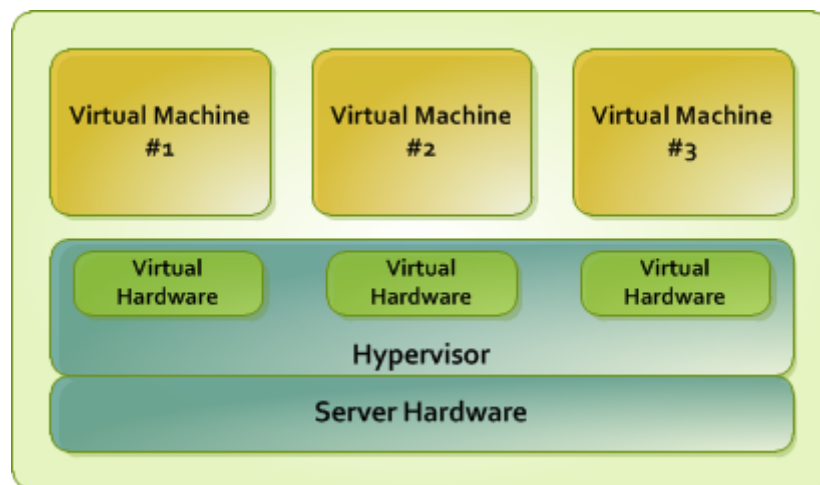
1.4 Hardware Virtualization Layer

This section familiarizes you with the second component of Virtuozzo Hybrid Server—the hardware virtualization layer. This layer provides the necessary environment for creating and managing virtual machines.

1.4.1 Hardware Virtualization Basics

Virtuozzo Hybrid Server is based on the concept of hardware virtualization. Hardware virtualization has a base layer—a hypervisor. This layer is loaded directly on the bare server and acts as an intermediary between the server hardware and virtual machines. To allocate hardware and resources to virtual machines, Virtuozzo Hybrid Server virtualizes all hardware on the server. Once virtualized, hardware and resources can be easily assigned to virtual machines. With its virtual hardware, a virtual machine runs its own complete copies of an operating system and applications.

The following diagram shows the basic architecture of hardware virtualization.



Specifically, Virtuozzo Hybrid Server uses the KVM/QEMU hypervisor and manages virtual machines via the libvirt API.

Hardware virtualization enables you to:

- Create multiple virtual machines with different operating systems on a single physical host.

- Run multiple guest operating systems and their applications simultaneously on a single physical host without rebooting.
- Consolidate and virtualize the computing environment, reduce hardware costs, lower operating expenses, and increase productivity.
- Use open APIs and SDK for integration with in-house and third-party applications.

1.4.2 Virtuozzo Virtual Machines

From the standpoint of applications and virtual machine users, each virtual machine (VM) is an independent system with an independent set of virtual hardware. This independence is provided by the Virtuozzo Hybrid Server hardware virtualization layer. The main features of the virtualization layer are the following:

- A virtual machine resembles and works like a regular computer. It has its own virtual hardware. Software applications can run in virtual machines without any modifications or adjustment.
- Virtual machine configuration can be changed easily (e.g., adding new virtual disks or increasing RAM).
- Virtual machines are fully isolated from each other (file system, processes, sysctl variables) and the Virtuozzo Hybrid Server host.
- A virtual machine can run any supported guest operating system. The guest OS and its applications are isolated inside a virtual machine and share physical hardware resources with other virtual machines.

1.4.2.1 Intel Virtualization Technology Support

Virtuozzo Hybrid Server provides support for Intel virtualization technologies comprising a set of processor enhancements and improving the work of virtualization solutions. Utilizing these technologies, Virtuozzo Hybrid Server can offload some workload to the system hardware, which results in the “near native” performance of guest operating systems.

1.4.3 Virtual Machine Hardware

A Virtuozzo virtual machine works like a usual standalone computer.

By default, virtual machines are created with the following virtual hardware:

- 1 VirtIO SCSI HDD, expanding

- 1 CD-ROM (IDE for Windows and Debian guests, VirtIO SCSI for Linux guests except Debian)
- 1 VirtIO network adapter, bridged
- 32MB video card

Other hardware added to a default VM may depend on the chosen distribution (see [Creating Virtual Machines](#)).

The complete range of virtual hardware a virtual machine can have is provided in the table below.

CPU	Up to 64
RAM	Up to 1 TB
Video adapter	VGA/SVGA video adapter with VBE 3.0
Video RAM	Up to 256 MB of video memory
Floppy disk drive	1.44 MB floppy disk drive mapped to an image file
IDE devices	Up to 4 IDE devices: <ul style="list-style-type: none"> • hard disk drives mapped to QCOW2 image files (up to 16 TB each) • DVD drives mapped to ISO image files
SCSI devices	Up to 15 SCSI devices: <ul style="list-style-type: none"> • hard disk drives mapped to QCOW2 image files (up to 16 TB each) • DVD drives mapped to ISO image files
VirtIO devices	Up to 15 VirtIO hard disk drives mapped to QCOW2 image files (up to 16 TB each)
Network interfaces	Up to 15 VirtIO (default), Intel 82545EM, or Realtek RTL8029 virtual network adapters.
Serial (COM) ports	Up to 4 serial (COM) ports mapped to a socket, a real port, or an output file
Keyboard	Generic USB or PS/2 keyboard
Mouse	Generic USB or PS/2 wheel mouse

1.4.4 Virtual Machine Files

A virtual machine has at least two files: a configuration file (PVS file) and a hard disk image file (HDD file). It can also have additional files: a file for each additional virtual hard disk and output files for virtual ports. By default, the virtual machines files are stored in the `/vz/vmprivate/<UUID>` directory on the Virtuozzo Hybrid Server server.

The list of files related to a virtual machine is given in the table below:

File name	Description
<code>.pvs</code>	Virtual machine configuration file. It defines the hardware and resources configuration of the virtual machine. The configuration file is automatically generated during the virtual machine creation.
<code>.sav</code>	Dump file created when you suspend the virtual machine. This file contains the state of the virtual machine and its applications at the moment the suspend was invoked.
<code>.mem</code>	Memory dump file for the suspended virtual machine. For a running virtual machine, it is a temporary virtual memory file.
<code>.hdd</code>	Hard disk image in QCOW2 format. When you create a virtual machine, you can create it with a new virtual hard disk or use an existing one. A virtual machine can have multiple hard disks.
<code>.iso</code>	CD/DVD disc image. Virtual machines treat ISO images as real CD/DVD discs.
<code>.txt</code>	Output files for serial ports. The output <code>.txt</code> files are generated when a serial port connected to an output file is added to the virtual machine configuration.

1.4.5 Support of Virtual and Real Media

This section lists the types of disks that can be used by Virtuozzo virtual machines and provides the information about basic operations you can perform on these disks.

1.4.5.1 Supported Types of Hard Disks

Virtuozzo virtual machines can only use virtual hard disk image files as hard disks.

1.4.5.2 Virtual Hard Disks

The capacity of a virtual hard disk can be set from 100 MB to 16 TB.

Virtuozzo Hybrid Server uses expanding virtual hard disks. The image file of such a disk is initially small in size (smaller than the set virtual disk size) and grows as data is added to the disk in the guest OS.

1.4.5.3 CD/DVD Disc Images

Virtuozzo Hybrid Server can use only CD/DVD disc images that are supported by the guest OS.

1.5 Virtuozzo Hybrid Server Configuration

Virtuozzo Hybrid Server allows you to configure settings for the physical server in general and for each container in particular. Among these settings are disk and user quotas, network parameters, default file locations, sample configuration files, and other.

Virtuozzo Hybrid Server stores all OS virtualization-related configuration information in the global configuration file `/etc/vz/vz.conf`. It defines container parameters like the default OS templates, disk quotas, logging, and so on.

The configuration file is read when the Virtuozzo Hybrid Server software and/or containers are started. However, many settings can also be changed on the fly by means of Virtuozzo Hybrid Server standard utilities like `pr1ct1`, with or without modifying the corresponding configuration file to keep the changes for the future.

1.6 Resource Management

Virtuozzo Hybrid Server resource management controls the amount of resources available to virtual machines and containers. The controlled resources include such parameters as CPU power, disk space, a set of memory-related parameters. Resource management allows you to:

- Efficiently share available physical server resources among virtual machines and containers.

- Guarantee quality of service in accordance with a service level agreement (SLA).
- Provide performance and resource isolation and protect from denial-of-service attacks.
- Simultaneously assign and control resources for a number of virtual machines and containers.
- Collect usage information for system health monitoring.

Resource management is much more important for Virtuozzo Hybrid Server than for a standalone server since server resource utilization in such a system is considerably higher than that in a typical system.

1.7 Understanding Licensing

To start using Virtuozzo Hybrid Server, you need a Virtuozzo Hybrid Server license. You must install this license on your server after or during Virtuozzo Hybrid Server installation. Every physical server hosting virtual machines and containers must have its own license. Licenses are issued by Virtuozzo Hybrid Server and define a number of parameters in respect of your physical server. The main licensed parameters are listed below:

- The number of physical CPUs which can be installed on the physical server. That is, a dual core or hyperthreading processor is regarded as one CPU.
- The license expiration date. A license can be time-limited or permanent. Virtuozzo Hybrid Server licenses have a start date, and if they are time-limited, can also have an expiration date specified in them. You must set up your system clock correctly. Otherwise, the license validation may fail.
- The number of virtual machines and containers that can simultaneously run on the physical server.
- The platform and architecture with which Virtuozzo Hybrid Server is compatible.

For instructions on how to install, update, view, and transfer licenses, see [Managing Licenses](#) on page 166.

1.8 Physical Server Availability Considerations

The availability of a physical server running Virtuozzo Hybrid Server is more critical than the availability of a typical PC server. Since it runs multiple virtual machines and containers providing a number of critical services, physical server outage might be very costly. It can be as disastrous as the simultaneous outage of a number of servers running critical services.

To increase physical server availability, we suggest that you follow the recommendations below:

- Use a RAID storage for critical virtual machines and containers. Do prefer hardware RAIDs, but software mirroring RAIDs might suit too as a last resort.
- Do not run any software on the server itself. Create special virtual machines and containers where you can host necessary services such as BIND, FTPD, HTTPD, and so on. On the server, you need only the SSH daemon. Preferably, it should accept connections from a pre-defined set of IP addresses only.
- Do not create users on the server itself. You can create as many users as you need in any virtual machine and container. Remember: compromising the server means compromising all virtual machines and containers as well.

CHAPTER 2

Managing Virtual Machines and Containers

This chapter describes how to perform day-to-day operations on virtual machines and containers.

2.1 Creating Virtual Machines and Containers

This section explains how to create new Virtuozzo virtual machines and containers using the `prctl create` command. The options you should pass to this command differ depending on whether you want to create a virtual machine or container.

2.1.1 Choosing OS EZ Templates for Containers

Before creating a container, you need to choose an OS EZ template it will be based on.

2.1.1.1 Listing OS EZ Templates

To find out which OS EZ templates are already installed on the hardware node and cached (i.e. ready to be used), you can use the `vzpkg list` command. For example:

```
# vzpkg list -0
almalinux-8-x86_64          2022-12-27 16:57:59
<...>
```

The timestamp next to an OS EZ template indicates when the template was cached.

Adding the `-o` option to the `vzpkg list` command, you can list only those OS EZ templates which are installed but not cached. You can also add the `--with-summary` option to display brief template descriptions:

```
# vzpkg list -o --with-summary
almalinux-8-x86_64           :AlmaLinux 8 (for AMD64/Intel EM64T) Virtuozzo Template
<...>
```

2.1.1.2 Installing and Caching OS EZ Templates

Some of the supported OS EZ templates may not be preinstalled, so you may need to perform additional steps before you can create containers based on these templates. To list templates available for installation in the official remote repositories, run the following command:

```
# vzpkg list --available
sles-15-x86_64              virtuozzo-os
<...>
```

To prepare a template for container creation, do the following:

1. Install the template package. For example:

```
# vzpkg install template sles-15-x86_64
```

2. Configure additional template parameters if needed. Some of the EZ templates may require specific preparation steps that depend on the operating system. To prepare, for example, a SLES 15 template for container creation, you additionally need to do the following:

- 2.1. Obtain twelve keys from the installed system by copying them in the `etc/zypp/repos.d/` directory:

```
cd /etc/zypp/repos.d/
grep 'https://updates.suse.com/SUSE/Products/SLE-Module-Basesystem/15-SP2/x86_64/\
product?'; *.repo $SLES_KEY_REL
grep 'https://updates.suse.com/SUSE/Updates/SLE-Module-Basesystem/15-SP2/x86_64/\
update?'; *.repo $SLES_KEY_UPD
grep 'https://updates.suse.com/SUSE/Products/SLE-Product-SLES/15-SP2/x86_64/\
product?'; *.repo $SLES_KEY_PRO
grep 'https://updates.suse.com/SUSE/Updates/SLE-Product-SLES/15-SP2/x86_64/\
update?'; *.repo $SLES_KEY_PRU
grep 'https://updates.suse.com/SUSE/Products/SLE-Module-Server-Applications/15-SP2/\
x86_64/product?' *.repo $SLES_SERVER_REPO
grep 'https://updates.suse.com/SUSE/Updates/SLE-Module-Server-Applications/15-SP2/\
x86_64/update?' *.repo $SLES_SERVER_UPDATES
grep 'https://updates.suse.com/SUSE/Products/SLE-Module-Web-Scripting/15-SP2/x86_64/\
product?'; *.repo $SLES_KEY_WEB_PRO
grep 'https://updates.suse.com/SUSE/Updates/SLE-Module-Web-Scripting/15-SP2/x86_64/\
update?'; *.repo $SLES_KEY_WEB_UPD
grep 'https://updates.suse.com/SUSE/Products/SLE-Module-Development-Tools/15-SP2/\
```

```
x86_64/product?'; *.repo $SLES_KEY_DEV_PRO
grep 'https://updates.suse.com/SUSE/Updates/SLE-Module-Development-Tools/15-SP2/\
x86_64/update?'; *.repo $SLES_KEY_DEV_UPD
grep 'https://updates.suse.com/SUSE/Products/SLE-Module-Legacy/15-SP2/x86_64/\
product?'; *.repo $SLES_KEY_LEG_PRO
grep 'https://updates.suse.com/SUSE/Updates/SLE-Module-Legacy/15-SP2/x86_64/\
update?'; *.repo $SLES_KEY_LEG_UPD
```

2.2. Add the obtained keys to the `/etc/vztt/url.map` file:

```
echo "$SLES_KEY_REL <obtained_key_1>" >> /etc/vztt/url.map
echo "$SLES_KEY_UPD <obtained_key_2>" >> /etc/vztt/url.map
echo "$SLES_KEY_PRO <obtained_key_3>" >> /etc/vztt/url.map
echo "$SLES_KEY_PRU <obtained_key_4>" >> /etc/vztt/url.map
echo "$SLES_SERVER_REPO <obtained_key_5>" >> /etc/vztt/url.map
echo "$SLES_SERVER_UPDATES <obtained_key_6>" >>/ etc/vztt/url.map
echo "$SLES_KEY_WEB_PRO <obtained_key_7>" >> /etc/vztt/url.map
echo "$SLES_KEY_WEB_UPD <obtained_key_8>" >> /etc/vztt/url.map
echo "$SLES_KEY_DEV_PRO <obtained_key_9>" >> /etc/vztt/url.map
echo "$SLES_KEY_DEV_UPD <obtained_key_10>" >> /etc/vztt/url.map
echo "$SLES_KEY_LEG_PRO <obtained_key_11>" >> /etc/vztt/url.map
echo "$SLES_KEY_LEG_UPD <obtained_key_12>" >>/ etc/vztt/url.map
```

You may also need to set the SUSE repository credentials (obtained from the [SUSE Customer Center](#)) in the `/etc/vztt/url.map` file as follows:

```
$SLES_LOGIN    <user>
$SLES_PASSWORD <password>
```

2.3. Copy certificates from the installed system to the

`/vz/template/sles/15/x86_64/config/os/default/files/` directory:

```
trust extract --format=openssl-directory --filter=ca-anchors --overwrite /tmp/certs/
scp -r /tmp/certs/* root@<your-node-ip>:/vz/template/sles/15/x86_64/config/os/default/files
```

3. Create the template cache:

```
# vzpkg create cache sles-15-x86_64
```

Now you can create containers based on the prepared template.

2.1.2 Creating Containers

To create a container, use the `prlctl create` command as follows:

```
# prlctl create MyCT --vmtypes ct
```

Virtuozzo Hybrid Server will create a new container with the name `MyCT` using the default parameters from

the global configuration file `/etc/vz/vz.conf`.

If you want to create a container with a guest OS different from the default specified in the global configuration file, add the `--ostemplate` option after the `prlctl create` command. For example:

```
# prlctl create MyCT --vmtype ct --ostemplate centos-6-x86_64
```

All container contents will be stored in this container's private area. To find out where the private area is located, use the `prlctl list` command as follows:

```
# prlctl list MyCT -i | grep "Home"
Home: /vz/private/26bc47f6-353f-444b-bc35-b634a88dbbcc
```

Take note of the following:

- The first time you install an operating system in a container, its cache is created. To create a cache, you need to have an active Internet connection to access repositories that contain packages for the respective operating system. You can also set up a local package repository and use this repository to provide packages for your operating system. A local package repository is also required for some commercial distributions (e.g., for Red Hat Enterprise Linux).
- For information on creating containers with preinstalled applications, see [Using Golden Images](#) on page 252.

2.1.2.1 Prerequisites for Creating Red Hat Enterprise Linux Containers

Creating Red Hat Enterprise Linux containers requires certificates to be present on the host. You will need a working Red Hat Enterprise Linux 7, 8, or 9 installation. You can use the same certificates for these three versions.

Do the following before creating such containers:

1. Find the certificate paths in the repository file, for example, `/etc/yum.repos.d/redhat.repo`:

```
sslcacert = /etc/rhsm/ca/redhat-uep.pem
sslclientkey = /etc/pki/entitlement/4662537897317115958-key.pem
sslclientcert = /etc/pki/entitlement/4662537897317115958.pem
```

2. Copy these certificates to `/etc/rhel/` on the host where you will be creating Red Hat Enterprise Linux containers.
3. Rename the copied certificates as follows:

```
sslclientkey.pem  
sslclientcert.pem
```

2.1.3 Creating Virtual Machines

Creating a new virtual machine means creating a VM configuration based on a distribution you specified. To create VMs, use the `prlctl create` command. For example:

```
# prlctl create MyVM --distribution centos7 --vmtype vm
```

This command creates a configuration for a virtual machine `MyVM`, adjusts it for running the CentOS 7 guest OS, and places all the required files in the `/vz/vmprivate/<UUID>` directory.

Once the virtual machine configuration is ready, you will need to install a supported guest OS in it (e.g., via VNC as described in *Enabling VNC Access to Virtual Machines and Containers* on page 278).

Note: To install a Windows guest in a virtual machine, you will need to manually mount a corresponding image file to VM's virtual floppy drive using the command `prlctl set --device-set fdd0 --image <image_file>`. The images are located in `/usr/share/vz-guest-tools/` and are typically named to match the guest OS name. For Windows Server 2012 (not R2), however, choose `floppy_win8.vfd`.

When choosing a distribution to install, have in mind that Virtuozzo Hybrid Server supports VM guest initialization via cloud-init, so you can perform some of the initial configuration tasks on stopped virtual machines. To be able to use this feature, you can install a “cloud-enabled” distribution instead of a regular one. For more information, see *Using cloud-init for Virtual Machine Guest Initialization* on page 19.

2.1.4 Supported Guest Operating Systems

The list of guest operating systems supported in virtual machines and containers is provided in the [Virtuozzo Hybrid Server Supported Guest Operating Systems Guide](#).

2.2 Performing Initial Configuration of Virtual Machines and Containers

Before you start using a newly created virtual machine or container, you will need to configure it. This section describes the main configuration steps.

2.2.1 Using cloud-init for Virtual Machine Guest Initialization

Virtuozzo Hybrid Server supports VM guest initialization via cloud-init, so you can perform some of the initial configuration tasks described further in this section on stopped virtual machines. The supported tasks are: installing guest tools, setting user names and passwords, and configuring network settings.

The changes resulting from performing the above tasks are not applied to the VM immediately but rather saved as instructions to be carried out when the guest OS with cloud-init is loading. So when you run a corresponding command (e.g., `prlctl set --userpasswd`), the following happens: the bundled image with cloud-init instructions is copied to the VM home path, a CD-ROM device is added to the VM, and the image is mounted to said CD-ROM. However, the changes (e.g., to the user name and password) will only be applied after you install and start loading the guest OS.

As mentioned above, you will need cloud-init installed in a guest OS for the feature to work. For Linux guests, the easiest way to get cloud-init is to install a “cloud-enabled” distribution that already comes with it. You can also install cloud-init manually (e.g., by running `yum install cloud-init` on CentOS 7). For Windows guests, you can create your own distributions with cloud-init or install it manually. The Windows version is available at <https://cloudbase.it/cloudbase-init/>.

2.2.2 Installing Virtuozzo Guest Tools

Virtuozzo guest tools enable you to configure running virtual machines from the physical host. With tools you can:

- Run commands in VMs with the `prlctl exec` command.
- Set passwords for users in VMs with the `prlctl set --userpasswd` command. If the user does not exist, it will be created.
- Obtain and change VM network settings.

Note: We highly recommend installing Virtuozzo guest tools before configuring a network inside a virtual machine. Installing the guest tools for the first time may cause deleting the previous manual network configuration. It also includes the cases of installing the guest tools after the **Configure network settings from guest OS** checkbox in your Virtuozzo Automator is selected or the command line equivalent `prlctl set VM1 --device-set net0 --configure no` is set.

In addition, installing Virtuozzo guest tools schedules weekly automatic “trimming” of file systems in Linux guests by means of the `fstrim` service. It reclaims unused storage space by discarding data blocks unused by VM’s file system. If you disable the service, it will not be re-enabled by future updates of Virtuozzo guest tools. In Windows guests, Storage Optimizer does the same periodic job by default.

Note: If a node is in a Virtuozzo Storage cluster that provides data redundancy by replication, automatic compacting (trimming) of the file system(s) that store data replicas is disabled. In case data redundancy is provided by erasure coding, trimming is enabled.

Important: The first automatic installation of Virtuozzo guest tools to a running SUSE Linux Enterprise Server/Desktop 15 SP6 virtual machine with the `vz-guest-tools-updater` tool requires stopping the VM. If you prefer the first-time installation of the guest tools without downtime, install them manually from our system ISO after ensuring the previously mounted ISO is ejected. Use the following CLI command:

```
# prlctl installtools MyVM
```

Otherwise, via Virtuozzo Automator, go to the **Configure** tab and select **Install Guest Tools**.

With the Virtuozzo guest tools set up, the subsequent run of `vz-guest-tools-updater` for the running VM will update the guest tools online (without downtime) to the latest version.

It is recommended to have cloud-init installed in the guest OS (see the previous section). In this case, you only need to do the following:

1. Mount the guest tools image shipped with Virtuozzo Hybrid Server to the virtual machine’s optical drive. For example:

```
# prlctl installtools MyVM
```

2. Start the virtual machine:

```
# prlctl start MyVM
```

Cloud-init will install the guest tools automatically.

If a VM guest OS does not have cloud-init, you can install Virtuozzo guest tools either automatically or manually.

To install the guest tools automatically without cloud-init:

1. Make sure these requirements are met:
 - The `vz-guest-tools-updater` package is installed on the node.
 - In the `/etc/vz/tools-update.conf` file, the `InstallTools` parameter is set to `true` (default).
 - The virtual machine has the `--tools-autoupdate` parameter set to `on` (default).
2. Stop the virtual machine before installing the guest tools:

```
# prlctl stop MyVM
```

3. Start `vz-guest-tools-updater` for the VM:

```
# vz-guest-tools-updater MyVM
```

4. Start the virtual machine:

```
# prlctl start MyVM
```

Once the VM is launched, the `vz-guest-tools-updater` tool will start installing Virtuozzo guest tools, which can take several minutes.

Important: By default, the updater tool obtains the new guest tools from the official repository. If the repository is not accessible, the updater tool forcibly mounts the guest tools image to VM's optical disk drive even if it is already in use.

To install the guest tools manually without cloud-init:

1. Mount the guest tools image shipped with Virtuozzo Hybrid Server to the virtual machine's optical drive. For example:

```
# prlctl installtools MyVM
```

2. Log in to the virtual machine and do the following:

- Inside a Linux VM, create a mount point for the optical drive with the guest tools image and run the installer:


```
# mount /dev/cdrom /mnt/cdrom
# bash /mnt/cdrom/install
```

- Inside a Windows VM, if autorun is enabled, launch the installer in the AutoPlay window.


CD Drive (D:) Virtuozzo Tools


Choose what to do with this disc.

Install or run program from your media

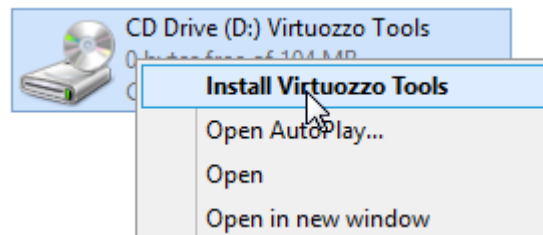
 Install Virtuozzo Tools
Publisher not specified

Other choices

 Open folder to view files
File Explorer

 Take no action

Otherwise right-click the optical drive in Explorer and click **Install Virtuozzo Tools**.



Note the following:

- Virtuozzo guest tools rely on QEMU guest agent which is installed alongside the tools. The agent daemon/service (`qemu-ga`) must be running for the tools to work.
- If you find out that Virtuozzo guest tools are incompatible with some software inside a virtual machine, you can uninstall them from that VM (for details, refer to *Uninstalling Virtuozzo Guest Tools from Virtual Machines* on page 289).

2.2.3 Configuring Network Settings

To make virtual machines and containers accessible from the network, you need to assign valid IP addresses to them and configure DNS servers. The session below illustrates setting these parameters for the virtual machine MyVM and the container MyCT:

- Assigning IPv4 and IPv6 addresses:

```
# prlctl set MyVM --device-set net0 --ipadd 10.0.186.100/24
# prlctl set MyVM --device-set net0 --ipadd 1fe80::20c:29ff:fe01:fb07
# prlctl set MyCT --ipadd 10.0.186.101/24
# prlctl set MyCT --ipadd fe80::20c:29ff:fe01:fb08
```

net0 in the commands above denotes the network card in the virtual machine to assign the IP address to. You can view all network cards of a virtual machine using the `prlctl list <VM_name> -i` command.

- Setting DNS server addresses:

```
# prlctl set MyVM --nameserver 192.168.1.165
# prlctl set MyCT --nameserver 192.168.1.165
```

Note the following:

- You can configure the network settings only for virtual machines that have Virtuozzo guest tools installed.
- To assign network masks to containers operating in the `venet0` network mode, you must set the `USE_VENET_MASK` parameter in the `/etc/vz/vz.conf` configuration file to `yes`.

2.2.4 Setting Passwords for Virtual Machines and Containers

In Virtuozzo Hybrid Server, you can use the `--userpasswd` option of the `prlctl set` command to create new accounts in your virtual machines and containers directly from the hardware node. The created account can then be used to log in to the virtual machine or container. The easiest way of doing it is to run this command:

```
# prlctl set MyCT --userpasswd user1:2wsx123qwe
```

This command creates the `user1` account in the container `MyCT` and sets the `2wsx123qwe` password for it. Now you can log in to the container as `user1` and administer it in the same way you would administer a standalone server: install additional software, add users, set up services, and so on.

The `prlctl set` command can also be used to change passwords for existing accounts in your virtual machines and containers. For example, to change the password for `user1` in the container `MyCT` to `0pi65jh9`,

run this command:

```
# prlctl set MyCT --userpasswd user1:0pi65jh9
```

When setting passwords for virtual machines and containers, keep in mind the following:

- You can manage user accounts only inside virtual machines that have Virtuozzo guest tools installed.
- You should use passwords that meet the minimum length and complexity requirements of the respective operating system. For example, for Windows Server 2008, a password must be more than six characters in length and contain characters from three of the following categories: uppercase characters, lowercase characters, digits, and non-alphabetic characters.
- You should not create accounts with empty passwords for virtual machines and containers running Linux operating systems.
- Ensure the `cloud-init` package is installed before changing a password in stopped SUSE Linux Enterprise Server 15 SP6 virtual machines.

2.2.5 Setting Startup Parameters

The `prlctl set` command allows you to define the `autostart` startup parameter for virtual machines and containers. Setting this parameter to `on` makes your virtual machine or container automatically boot at the physical server startup. For example, to enable the container `MyCT` and the virtual machine `MyVM` to automatically start on your server boot, you can execute the following commands:

- For the container `MyCT`:

```
# prlctl set MyCT --autostart on
```

- For the virtual machine `MyVM`:

```
# prlctl set MyVM --autostart on
```

Notice that the `autostart` parameter will have effect only on the next server startup.

2.3 Starting, Stopping, Restarting, and Querying Status of Virtual Machines and Containers

After a virtual machine or container has been created, it can be managed like a usual computer.

2.3.1 Starting Virtual Machines and Containers

You can start virtual machines and containers with the `prlctl start` command. For example:

- To start the container `MyCT`:

```
# prlctl start MyCT
```

- To start the virtual machine `MyVM`:

```
# prlctl start MyVM
```

2.3.2 Stopping Virtual Machines and Containers

You can stop virtual machines and containers with the `prlctl stop` command. For example:

- To stop the container `MyCT`:

```
# prlctl stop MyCT
```

- To stop the virtual machine `MyVM`:

```
# prlctl stop MyVM
```

2.3.3 Restarting Virtual Machines and Containers

You can restart virtual machines and containers with the `prlctl restart` command. For example:

- To restart the container `MyCT`:

```
# prlctl restart MyCT
```

- To restart the virtual machine `MyVM`:

```
# prlctl restart MyVM
```

Note: Restarting virtual machines requires a guest OS and Virtuozzo guest tools to be installed.

2.3.4 Checking Status of Virtual Machines and Containers

You can check the status of a virtual machine or container with the `prlctl status` command. For example:

- To check the status of the container `MyCT`:

```
# prlctl status MyCT
CT MyCT exists running
```

- To check the status of the virtual machine `MyVM`:

```
# prlctl status MyVM
Vm MyVM exists stopped
```

2.4 Listing Virtual Machines and Containers

To get an overview of the virtual machines and containers existing on the physical server and to get additional information about them—their IP addresses, hostnames, current resource consumption, and so on—use the `prlctl list` command. In the most general case, you can get a list of all virtual machines and containers by issuing the following command:

```
# prlctl list -a
UUID                               STATUS  IP_ADDR  T  NAME
{600adc12-0e39-41b3-bf05-c59b7d26dd73}  running  10.10.1.101  CT  MyCT
{b2de86d9-6539-4ccc-9120-928b33ed31b9}  stopped  10.10.100.1  VM  MyVM
```

The `-a` option shows all—both running and stopped—VMs and containers (only running VMs and containers are shown by default). The default columns include VM and container UUIDs, status, type, IP addresses, and names. The list of columns can be customized with the `-o` option. For example:

```
# prlctl list -a -o name,ctid
NAME                               UUID
MyCT                               {26bc47f6-353f-444b-bc35-b634a88dbbcc}
MyVM                               {b8cb6d99-1af1-453d-a302-2fddd8f86769}
```

Note: To see a list of all columns, run `prlctl list -L`.

2.5 Cloning Virtual Machines and Containers

You can create a copy (clone) of a particular virtual machine or container that will have identical data and resource parameters. Cloning may save time as clones require little reconfiguration compared to setting up new virtual machines or containers.

You can clone stopped virtual machines and stopped and running containers. For example:

```
# prlctl clone MyCT --name MyCT_clone
# prlctl clone MyVM --name MyVM_clone
```

The `--name` option specifies a name for the clone.

When cloning Windows virtual machines, consider changing their security identifiers (SIDs) with the `--changesid` option.

Note: For Windows 2019 VMs, use Microsoft's Sysprep tool instead of the `--changesid` option.

Successfully cloned virtual machines and containers will be shown in the list of virtual environments on the host. For example:

```
# prlctl list -a
UUID          STATUS      IP_ADDR      T  NAME
{62951c2a-...} stopped    10.30.10.101 CT  MyCT
{49b66605-...} stopped    10.30.10.101 CT  MyCT_clone
{7f4904ad-...} stopped    10.30.128.115 VM  MyVM
{2afb2aa2-...} stopped    10.30.128.134 VM  MyVM_clone
```

The example above shows that the cloned container has the same IP address as the original container.

Before starting to use the clones, make sure their IP addresses are unique (for instructions on how to assign IP addresses to VMs and containers, see [Configuring Network Settings](#) on page 23).

Note the following:

- When cloning a virtual machine, the MAC address of a network adapter in the clone changes. Thus, it may require additional reconfiguration for guest OSs that do not automatically obtain the MAC addresses of network adapters.

- A cloned SUSE Linux Enterprise Server 15 SP6 virtual machine gets a NIC name with an index increased by one. For example, if an original NIC name is `eth0`, it will acquire +1 to its existing name. Thus, the name will change to `eth1`. The subsequent clone operation from the cloned VM will cause the same changes to the NIC name of a newly cloned VM, i.e., `eth2`, `eth3`, etc.
- To avoid situations when cloned SUSE Linux Enterprise Server 15 SP6 virtual machines obtain duplicate IP addresses via DHCP, regenerate `machine-id` according to the instructions from [Resolving Duplicate Machine IDs in Cloned Salt Clients](#).

2.5.1 Creating Linked Clones

Starting from Virtuozzo Hybrid Server 7.0.7 (Update 7), you can create linked clones of virtual machines. A linked clone is a copy of a virtual machine that shares virtual disks with the original virtual machine. Linked clones take less time and disk space to deploy as they store only changes to the original disks rather than copy them whole. A linked clone cannot run without access to its parent, so make sure the original virtual machine is available and its disks are not corrupted.

To create a linked clone, add the `--linked` option to the `prlctl clone` command. For example:

```
# prlctl clone MyVM --name MyVM_linked_clone --linked
```

On the host, linked clones are shown as usual virtual machines. For example:

```
# prlctl list -a
UUID          STATUS      IP_ADDR      T  NAME
{7f4904ad-...} stopped    10.30.128.115 VM MyVM
{2e9862f6-...} stopped    10.30.128.135 VM MyVM_linked_clone
```

Note: Migration, backup, restore, and unlink operations are not supported for linked clones.

2.5.2 Configuring Default Directories

When cloning a virtual machine or container, you can also override the following default directories:

- default directory `/vz/vmprivate/<dest_UUID>` storing the files of a cloned virtual machine (where `<dest_UUID>` denotes the UUID of the resulting virtual machine). To store the files of the `MyVM_clone` virtual machine in a custom directory, you can run the following command:

```
# prlctl clone MyVM --name MyVM_clone --dst vz/vmprivate/customVMs
```

In this case all virtual machine files will be placed to the /customVMs directory. Note that the specified directory must exist on the server; otherwise, the command will fail.

- default directory /vz/private/<dest_UUID> defining the container private area (where <dest_UUID> denotes the UUID of the resulting container). To define a custom private area path for the container MyCT_clone, you can execute the following command:

```
# prlctl clone MyCT1 --name MyCT_clone --dst /vz/private/customCTs
```

Note: The default /vz/vmprivate and /vz/private are valid for servers that do not participate in Virtuozzo Hybrid Server storage clusters.

2.6 Suspending Virtual Machines and Containers

Virtuozzo Hybrid Server allows you to suspend a running virtual machine or container on the physical server by saving its current state to a special file. Later on, you can resume the virtual machine or container and get it in the same state it was suspended. Suspending your virtual machines and containers may prove useful, for example, if you need to restart the physical server but do not want to:

- Quit the applications currently running in the virtual machine or container.
- Spend much time on shutting down the guest operating system and then starting it again.

You can use the `prlctl suspend` command to save the current state of a virtual machine or container. For example, you can issue the following command to suspend the container MyCT:

```
# prlctl suspend MyCT
```

At any time, you can resume the container MyCT by executing the following command:

```
# prlctl resume MyCT
```

Once the restoration is complete, any applications that were running in the container MyCT when it was suspended would be running again and the data inside the container would be the same.

2.7 Running Commands in Virtual Machines and Containers

Virtuozzo Hybrid Server allows you to execute arbitrary commands inside virtual machines and containers by running them on the physical server, i.e. without the need to log in to the respective virtual machine or container. For example, this can be useful in these cases:

- If you do not know the virtual machine or container login information, but need to run some diagnosis commands to verify that it is operational.
- If the virtual machine or container has no network access.

In both cases, you can use the `prlctl exec` command to run a command inside the respective virtual machine or container. By default, running `prlctl exec <command>` is equivalent to executing `bash -c <command>` in a Linux VM or container or `cmd /c <command>` in a Windows VM. Add the `--without-shell` option to run commands directly without the shell.

The session below illustrates the situation when you run the stopped SSH daemon inside a Linux virtual machine with the name of `My_Linux`:

```
# prlctl exec My_Linux /etc/init.d/sshd status
openssh-daemon is stopped
# prlctl exec My_Linux /etc/init.d/sshd start
Starting sshd: [ OK ]
# prlctl exec My_Linux /etc/init.d/sshd status
openssh-daemon (pid 26187) is running...
```

Note the following:

- You can use the `prlctl exec` command only inside virtual machines that have Virtuozzo guest tools installed.
- The `prlctl exec` command is executed inside a virtual machine or container from the `/` directory rather than from `/root`.

2.8 Deleting Virtual Machines and Containers

You can delete a virtual machine or container that is not needed anymore using the `prlctl delete` command. Note that you cannot delete a running or mounted virtual machine or container. The example below illustrates deleting the running container `MyCT`:

```
# prlctl delete MyCT
Removing the CT...
Failed to remove the CT: Unable to complete the operation. This operation cannot \
be completed because the virtual machine "{4f27f27f-c056-4a65-abf6-27642b6edd21}" \
is in the "running" state.
# prlctl stop MyCT
Stopping the CT...
The CT has been successfully stopped.
# prlctl delete MyCT
Removing the CT...
The CT has been successfully removed.
```

2.9 Viewing Detailed Information about Virtual Machines and Containers

To view detailed information about a virtual machine or container, you can use the `prlctl list -i` command. For example, the following command lists all information about the virtual machine `MyVM`:

```
# prlctl list -i MyVM
```

The following table describes the main options displayed by `prlctl list -i`.

Option	Description
ID	Virtual machine identifier. Usually, you use this ID, along with the virtual machine name, when performing an operation on the virtual machine.
EnvID	Kernel virtual machine identifier. This is the ID the kernel on the physical server uses to refer to a virtual machine when displaying some information on this virtual machine.
Name	Virtual machine name.
Description	Virtual machine description.
State	Virtual machine state.
OS	Guest operating system installed in a virtual machine.

Continued on next page

Table 2.9.1 -- continued from previous page

Option	Description
Uptime	Time that shows for how long a virtual machine has been running since counter reset. Note: The uptime counter as well as count start date and time can be reset with the <code>prlctl reset-uptime</code> command.
Home	Directory storing virtual machine files.
Guest tools	Shows whether Virtuozzo guest tools are installed in a virtual machine.
Autostart	Shows whether a virtual machine is automatically started when you turn on the physical server.
Boot order	Order in which the virtual machine devices are checked for an operating system.
Hardware	Devices available in a virtual machine.
Offline management	Denotes whether the offline management feature is enabled for the virtual machine, and if yes, lists the available offline services.

Note: The options `prlctl list` displays for containers are similar to those for virtual machines.

2.10 Managing Virtual Machine and Container Backups

Backing up your virtual machines and containers on a regular basis is essential for system reliability. Virtuozzo Hybrid Server allows you to back up and restore virtual machines and containers on the local node with the `prlctl` and `prlsrvctl` utilities.

2.10.1 Creating Virtual Machine and Container Backups

You can create backups of virtual machines and containers with the `prlctl backup` command. The command is executed on the local node where the virtual machines or containers are located. The resulting backups can be stored on either the local node or a remote one (e.g., a dedicated backup server).

By default, an incremental backup is created that contains only the files changed since the previous full or incremental backup. If no previous backups exist, a full backup is created. You can, however, forcibly create a full backup with the `-f` option.

Note: For increased security during backup operations, Virtuozzo Hybrid Server provides connection tunneling between the local and remote nodes. Tunneling increases backup time, so if you want to speed up the process and do not need a secure tunnel between nodes, you can disable connection tunneling with the `--no-tunnel` option. To use it, configure the firewall of the destination node to allow incoming connections on any port on the corresponding network interface.

For example, to create a backup of the virtual machine `MyVM` and store it on the local node, run the following command:

```
# prlctl backup MyVM
...
The VM has been successfully backed up with \
backup ID {746dba2a-3b10-4ced-9dd6-76a2b1c14a69}
```

The backup UUID, like the one shown above, will be required to manage the backup in the future.

To create a backup of the virtual machine `MyVM` and store it on a remote node, specify the remote node's IP address or hostname with the `-s` option, for example:

```
# prlctl backup MyVM -s user:passwd@192.168.0.10
```

The root account is used to log in to the remote node by default, so you will be asked for the root password. You can also provide different credentials and port in the format `[<user>[:<passwd>]@<server>[:<port>]`. Alternatively, you can set up RSA authentication and not use passwords at all (see [Setting Up RSA Authentication Between Nodes](#) on page 297).

By default, backups are placed in the `/vz/vmprivate/backups` directory. To set another default backup directory for the node, use `prlsrvctl`. For example:

```
# prlsrvctl set --backup-path "/vz/mybackupdir"
```

```
# prlsrvctl info | grep "Backup path"
Backup path: /vz/mybackupdir
```

To change the default backup directory for a specific virtual machine or container, use `prlctl`. For example:

```
# prlctl set MyVM --backup-path "/vz/myvmbackupdir"
# prlctl list -i MyVM | grep "Backup path"
Backup path: /vz/myvmbackupdir
```

Now the command `prlctl backup MyVM` will back up `MyVM` to `/vz/myvmbackupdir` by default.

In general, you can use the `--backup-path` option to back up VEs to any existing directory, even if it is different from VE's default backup directory.

You can also combine the `-s` and `--backup-path` options to back up VEs to any existing directory on a remote node. For example:

```
# prlctl backup MyVM -s user:passwd@192.168.0.10 --backup-path "/vz/mybackupdir"
```

If, however, the remote node runs an older version of Virtuozzo Hybrid Server that does not support the `--backup-path` option, the backup will be placed in the default backup directory on the remote node.

Note the following:

- You can back up both running and stopped virtual machines and containers.
- Creating a consistent backup of a running virtual machine requires the Virtuozzo guest tools to be installed in that virtual machine.
- You cannot back up virtual machines with attached physical HDDs, mounted ISO or floppy disk images, etc.
- If you restore a VM or container from backup and back it up again, a full backup will be created. The following backups will be incremental.
- Incremental backups will not be created if you manage backups with more than one solution that uses the Virtuozzo API, e.g., `prlctl backup` and some other.
- Container images are compacted at 02:00 every night by `pcompact` launched by a cron job. This process locks container images and prevents any other processes from opening them for writing. As a result, backup processes on locked images may fail. To avoid this, you can either reschedule backups or the `pcompact` job.

2.10.1.1 Improving Backup Performance

A virtual environment with a write-intensive workload may write to its snapshot blocks faster than they are backed up, so writing is slowed down to the backup speed. Besides, if the backup process stalls, the virtual environment also stalls. To avoid this, the original block can be copied to a temporary intermediate snapshot called a reversed delta and then rewritten without delay.

To switch to backups with reversed delta, run

```
# prlsrvctl set --backup-mode push-with-reversed-delta
```

Temporary snapshots will be saved to the virtual environment's home directory by default. To set a different location, use the command

```
# prlsrvctl set --backup-tmpdir <path>
```

Temporary snapshots will be written with the `O_DIRECT` flag to avoid wasting node's cache, so the directory must allow `O_DIRECT` writes.

To revert to direct write backups, run

```
# prlsrvctl set --backup-mode push
```

Or you can back up a virtual environment with the `--no-reversed-delta` option. For example:

```
# prlctl backup MyVM --no-reversed-delta
```

When direct write backups are created, the amount of page cache available to them is limited to 512 MB to reduce wasting node's cache.

2.10.2 Listing Virtual Machine and Container Backups

You can list backups on the node with the `prlctl backup-list` command. For example:

```
# prlctl backup-list
      ID Backup_ID      Node      Date              Type  Size
{c1dee22f-<...>} {209d54a0-<...>} test.local 09/09/2020 10:19:32   f 411566405
```

If no options are specified, the command lists backups that reside in the default backup directory set for the VE or, if that has not been done, for the node. To list backups residing in a custom directory, add the `--backup-path` option to the command. For example:

```
# prlctl backup-list --backup-path "/vz/mybackupdir"
      ID Backup_ID      Node      Date              Type  Size
```

```
{c1dee22f-<...>} {0d882beb-<...>} test.local 09/09/2020 14:03:25 f 411566405
{eab1e7fd-<...>} {33002958-<...>} test.local 09/09/2020 14:03:41 f 229376
```

To list backups for a specific VE, add the VE name or UUID to the command. For example:

```
# prlctl backup-list MyCT --backup-path "/vz/mybackupdir"
      ID Backup_ID      Node      Date      Type      Size
{c1dee22f-<...>} {0d882beb-<...>} test.local 09/09/2020 14:03:25 f 411566405
```

You can also add the `-s` option to list backups residing on a remote node. For example:

```
# prlctl backup-list MyCT -s user:passwd@192.168.0.10 --backup-path "/vz/myremotebakdir"
      ID Backup_ID      Node      Date      Type      Size
{c1dee22f-<...>} {c91b57a6-<...>} test.remote 09/09/2020 14:03:25 f 411566405
{c1dee22f-<...>} {c91b57a6-<...>}.2 test.remote 09/09/2020 18:43:40 i 250016
```

If, however, the remote node runs an older version of Virtuozzo Hybrid Server that does not support the `--backup-path` option, the command will list backups residing in the default backup directory on the remote node.

In any case, the command shows the following information on the listed backups:

Column	Description
ID	Virtual machine or container UUID.
Backup_ID	Backup UUID. Specify it to perform operations on the backup.
Node	The hostname of the physical server where the backup is stored.
Date	The date and time when the backup archive was created.
Type	The backup type: <ul style="list-style-type: none"> • Full, <code>f</code>. A complete backup. • Incremental, <code>i</code>, with only the files changed since the previous full or incremental backup. This is the default backup type.
Size	The size of the backup image, in bytes.

If required, you can filter the backup list with the `--vmtype ct|vm|all` option that only shows backups of containers, virtual machines, or both. To list only backups created on the local node, use the `--localvms` option.

2.10.3 Restoring Virtual Machines and Containers from Backups

Local or remote backups of virtual machines and containers can be restored with the `pr1ct1 restore` command.

Note: For increased security during restore operations, Virtuozzo Hybrid Server provides connection tunneling between the local and remote nodes. Tunneling increases restore time, so if you want to speed up the process and do not need a secure tunnel between nodes, you can disable connection tunneling with the `--no-tunnel` option. To use it, configure the firewall of the destination node to allow incoming connections on any port on the corresponding network interface.

The following rules and considerations apply:

- Restore commands are run on the node where the backups will be restored.
- Only stopped virtual machines and containers can be restored from backup.
- Virtuozzo 6 backups can be restored to Virtuozzo Hybrid Server 7 nodes (with conversion to Virtuozzo Hybrid Server 7 format).
 - Backups of virtual machines and containers with guests unsupported in Virtuozzo Hybrid Server 7 may not be restored correctly (see [Virtuozzo Hybrid Server Supported Guest Operating Systems Guide](#)).
 - VZFS-based containers must be converted to ploop format and backed up again before they can be restored to Virtuozzo Hybrid Server 7.

Note: If conversion of the restored VM fails, the restored VM is deleted from the destination server and you can try again. If the second attempt also fails, you need to enable a legacy VM debug mode on the destination Virtuozzo Hybrid Server 7 server (see [Enabling Legacy VM Debug Mode](#) on page 292), make another restore attempt, send the problem report, and contact the technical support team. With the debug mode enabled, the migrated VM will not be deleted from the destination Virtuozzo Hybrid Server 7 server after conversion failure. It will remain stopped or running with disabled network to let the technical support team study the memory dump and find out the reason for failure.

For example, to restore a backup of the virtual machine `MyVM` with the UUID `a53f1184-333e-41cf-b410-2ec8ffea67d4`, run either command:

```
# prlctl restore MyVM
# prlctl restore a53f1184-333e-41cf-b410-2ec8ffea67d4
```

If multiple backups of a VE exist, the latest one will be restored. To restore a particular backup, specify its UUID with the `-t` option. For example:

```
# prlctl restore -t 24a3011c-8667-4870-9e11-278f1398eab0
```

By default, the command looks for backups in the default backup directory set for the VE or, if that has not been done, for the node. To restore a VE from a backup residing in a custom directory, use the `--backup-path` option. For example:

```
# prlctl restore MyVM --backup-path "/vz/myvmbackupdir"
```

Add the `-s` option if the backup is on a remote node. For example:

```
# prlctl restore MyVM -s user:passwd@192.168.0.10 --backup-path "/vz/myvmbackupdir"
```

If, however, the remote node runs an older version of Virtuozzo Hybrid Server that does not support the `--backup-path` option, the command will restore VE's latest backup from the default backup directory on the remote node.

If backups are stored remotely and the VE to be restored does not exist on the node, you can restore it by specifying its UUID as well as backup server's IP address or hostname. For example:

```
# prlctl restore -t 24a3011c-8667-4870-9e11-278f1398eab0 -s 192.168.0.10
```

If the VM or container exists on the node (e.g., has been restored from a remote backup once), you can restore it from the latest backup by its name or UUID.

If necessary, you can transfer remote backups to the local node and restore them locally. To do this:

1. Find out the backup directories on the source and destination servers.

If the VE has its own backup directory, locate it with `prlctl list`. For example:

```
# prlctl list -i MyVM | grep "Backup path"
Backup path: /vz/myvmbackupdir
```

Otherwise, locate node's default backup directory. For example:

```
# prlsrvctl info | grep "Backup path"
Backup path: /vz/mybackupdir
```

2. Copy backups to a directory on the local node. Or, if you keep backups on a network storage, mount it to a local directory.

- Restore the backups with the `prlctl restore -t` command as shown in the example above. Specify the local directory where you have copied the backups with `--backup-path`.

To reduce downtime of a restored virtual machine, you can restore it live. In this case, the restored virtual machine is started right after launching the restore process. To do it, use the `prlctl restore --live` command. For example:

```
# prlctl restore MyVM --live
```

Note: Online restore of VMs with EFI BIOS is not supported.

2.10.4 Deleting Virtual Machine and Container Backups

You can delete backups with the `prlctl backup-delete` command.

To delete all backups of a VM or container, specify only the VM or container name or UUID:

```
# prlctl backup-delete MyVM
```

To remove a specific VM or container backup, provide the VM or container name or UUID as well as the backup ID (you can find out these IDs with the `prlctl backup-list` command). For example:

```
# prlctl backup-list MyVM
      ID  Backup_ID      Node      Date              Type  Size
{c1dee22f-<...>} {209d54a0-<...>} test.local 2011-05-30 10:19:32  f 411566405
# prlctl backup-delete MyVM -t 209d54a0-e3b8-4a03-9ca8-d4cc7a2a27ca --keep-chain
```

With the `--keep-chain` option specified, the remaining backup chain will be preserved after specific backups are deleted from it.

To delete backups stored on a remote node, use the `-s` option. For example:

```
# prlctl backup-delete MyVM -s user:passwd@192.168.0.10
```

To delete backups stored in a custom directory, use the `--backup-path` option. For example:

```
# prlctl backup-delete MyVM --backup-path "/vz/myvmbackupdir"
```

You can also combine these two options to remove backups from a custom directory on a remote node.

2.10.5 Backing Up Entire Servers

In addition to backing up single virtual machines and containers, you can create backups of all virtual environments on the node with the `prlsrvctl backup` command. For example:

```
# prlsrvctl backup -f
Backing up the CT MyCT
...
The CT has been successfully backed up with backup id {b14ec76d-c0e2-432f-859a- \
4727c0042065}
Backing up the VM MyVM
...
The VM has been successfully backed up with backup id {746dba2a-3b10-4ced-9dd6- \
76a2blc14a69}.
```

Note the following:

- You can back up both running and stopped virtual machines and containers.
- Creating consistent backups of running virtual machines requires the Virtuozzo guest tools to be installed in those virtual machines.
- You cannot back up virtual machines with attached physical HDDs, mounted ISO or floppy disk images, etc.

2.10.6 Attaching Backups to Virtual Machines and Containers

To read the contents of a virtual machine or container backup, you can attach it to a virtual machine or container as a virtual hard disk.

Note the following:

- Only local backups can be attached.
- The attached backup is writable so that the file system can process its journal on mount. However, all changes will be discarded when the backup is detached. The amount of data that can be written to the attached backup is limited to 256MB.
- Attached backups are not preserved during clone, backup, and snapshot operations.
- Use the `nouuid` mount option to attach backups to VMs with XFS partitions (e.g., VzLinux 8.4). Doing so will help avoid XFS UUID collisions.

2.10.6.1 Attaching Backups to Linux Virtual Machines

1. Make sure that the `prl_backup` and `kpartx` utilities are installed in the virtual machine the backup will be attached to. The `prl_backup` utility is provided by Virtuozzo guest tools.
2. Obtain the ID and file name of the backup to attach. You can do this with the `prlctl backup-list` command. For example:

```
# prlctl backup-list vm2 -f
...
Backup_ID: {0fcd6696-c9dc-4827-9fbd-6ee3abe017fa}
...
Name: hddisk.hdd.qcow2
```

3. Attach the backup as an HDD to the Linux VM you will access the backup from. You can do this with the `prlctl set --backup-add` command. For example:

```
# prlctl set vm1 --backup-add {0fcd6696-c9dc-4827-9fbd-6ee3abe017fa} \
--disk hddisk.hdd.qcow2
Creating hdd1 (+) sata:2 real='backup:/// {0fcd6696-c9dc-4827-9fbd-6ee3abe017fa}/ \
hddisk.hdd.qcow2' backup='{0fcd6696-c9dc-4827-9fbd-6ee3abe017fa}' \
disk='hddisk.hdd.qcow2'
```

If the backup contains multiple disks and you need to connect them all, omit the `--disk` option.

4. Obtain the name of the newly attached device, which is disabled at the moment, using the `prl_backup list` command. For example:

```
# prlctl exec vm1 prl_backup list
...
List of disabled attached backups:
[1] /dev/sdc
```

5. Enable the backup with the `prl_backup enable` command. For example:

```
# prlctl exec vm1 prl_backup enable /dev/sdc
```

6. Optionally, make sure the backup is now enabled, using the `prl_backup list -e` command. For example:

```
# prlctl exec vm1 prl_backup list -e
List of enabled attached backups:
[1] /dev/sdc (/dev/mapper/backup1)
NAME          TYPE  SIZE  FSTYPE  UUID
MOUNTPOINT
backup1 (dm-3) dm    64G
```

```
|-backup1p1 (dm-4) part 500M ext4 1ac82165-113d-40ee-8ae2-8a72f62d95bf
|-backup1p2 (dm-5) part 63.5G LVM2_mem Zw9QiY-BiU5-o8dn-ScTK-v0Zx-KujW-wbgmS3
```

Now you can mount the required backup part as a file system.

Use `mount` to mount the `ext4` part. For example:

```
# prlctl exec vm1 mount /dev/mapper/backup1p1 /mnt/backup1p1
```

You can now access the backup part contents at `/mnt/backup1p1`.

To avoid an XFS UUID collision, mount the XFS partitions with the `nouuid` mount option:

```
prlctl exec vm1 mount -o nouuid /dev/mapper/backup1p1 /mnt/backup1p1
```

Do the following to mount the `LVM2_member` part:

1. Assign the new volume group a new name so it can coexist with other volume groups. You can do this with the `vgimportclone` command. For example:

```
# prlctl exec vm1 vgimportclone -n backup1p2 /dev/mapper/backup1p2
...
Volume group "VolGroup" successfully renamed to "backup1p2"
...
Found volume group "backup1p2" using metadata type lvm2
...
```

Note: If the guest operating system is RockyLinux 9.3, use an additional option `--importdevices` when running the `vgimportclone` command. For example:

```
prlctl exec vm1 vgimportclone --importdevices -n r1-backup /dev/mapper/backup1p2
```

2. Obtain the list of mountable logical volumes with the `lvs` command. For example:

```
# prlctl exec vm1 lvs | grep backup1p2
lv_home backup1p2 -wi-----11.54g
lv_root backup1p2 -wi----- 50.00g
lv_swap backup1p2 -wi-----1.97g
```

3. Activate the required logical volume with the `lvchange -ay` command. For example:

```
# prlctl exec vm1 lvchange -ay /dev/backup1p2/lv_root
```

4. Mount the logical volume as a file system. For example:

```
# prlctl exec vm1 mount /dev/backup1p2/lv_root /mnt/backup1p2
```

You can now access the backup part contents at `/mnt/backup1p2`.

2.10.6.2 Attaching Backups to Windows Virtual Machines

1. Obtain the backup ID and file name. For example:

```
# prlctl backup-list vm2 -f
...
Backup_ID: {cff742a9-f942-41c5-9ac2-ace3b4eba783}
...
Name: hddisk.hdd.qcow2
```

2. Attach the required backup as an HDD to the Windows virtual machine you will access the backup from. For example:

```
# prlctl set vm1 --backup-add {cff742a9-f942-41c5-9ac2-ace3b4eba783} \
--disk hddisk.hdd.qcow2
Creating hdd1 (+) sata:2 real='backup:///{cff742a9-f942-41c5-9ac2-ace3b4eba783}/ \
hddisk.hdd.qcow2' backup='{cff742a9-f942-41c5-9ac2-ace3b4eba783}' \
disk='hddisk.hdd.qcow2'
```

The attached backup will appear as a ready-to-use disk in the Windows virtual machine.

2.10.6.3 Attaching Backups to Linux Containers

1. Obtain the backup ID and file name with the `prlctl backup-list -f` command. For example:

```
# prlctl backup-list 102 -f
...
Backup_ID: {d70441dd-f077-44a0-8191-27704d4d8fdb}
...
Name: root.hdd.qcow2c
...
```

2. Attach the backup as an HDD to the Linux container you will access the backup from. You can do this with the `prlctl set --backup-add` command. For example:

```
# prlctl set MyCT --backup-add {d70441dd-f077-44a0-8191-27704d4d8fdb} \
--disk root.hdd.qcow2c
Creating hdd1 (+) sata:0 real='backup:///{d70441dd-f077-44a0-8191-27704d4d8fdb}/ \
root.hdd.qcow2c' backup='{d70441dd-f077-44a0-8191-27704d4d8fdb}' \
disk='root.hdd.qcow2c'
```

3. Using the backup ID, identify the ploop device corresponding to the backup. For example:

```
# ploop list | grep {d70441dd-f077-44a0-8191-27704d4d8fdb}
ploop28261 /buse/{8417a267-0919-4c8f-a31d-68671358d6a8}_ \
{d70441dd-f077-44a0-8191-27704d4d8fdb}_root.hdd.qcow2c/content
```

4. Mount the logical volume as a file system. For example:

```
# prlctl exec MyCT mount /dev/ploop28261p1 /mnt/backup1
```

You can now access the backup contents at `/mnt/backup1`.

2.10.7 Detaching Backups from Virtual Machines and Containers

Note: Before detaching a backup from a running virtual machine, do the following:

1. (Linux VMs) Disable the backup device with the `prl_backup disable` command run in the guest OS.
2. (Linux and Windows VMs) Disconnect the corresponding virtual disk by running `prlctl set --device-set hdd<N> --disconnect` command on the server.

- To detach all virtual disks from all backups attached to a virtual machine or container, use the `prlctl set --backup-del all` command. For example:

```
# prlctl set vm1 --backup-del all
```

- To detach all virtual disks from a specific backup attached to a virtual machine or container, use the `prlctl set --backup-del <backup_ID>` command. For example:

```
# prlctl set vm1 --backup-del {e13561bb-5676-49bd-a935-ae0145eb0229}
```

- To detach a specific virtual disk from any of the backups attached to a virtual machine or container, delete the disk with the `prlctl set --device-del hdd<N>` command. For example:

```
# prlctl set vm1 --device-del hdd1
```

2.10.8 Best Practices for Handling Page Cache When Using Third-Party Backup Software

2.10.8.1 Performing Backups from the Page Cache Angle

Backup software by its nature must, first, read all files from a disk or directory to back up and, second, write that data somewhere. If no precautions are taken, both steps generate huge amounts of page cache, as both file reads and writes send their data to the page cache.

As mentioned before, because RAM size is normally much smaller than disk size, backup forces the memory

management subsystem to drop the existing page cache used by processes and replace it with the page cache generated by backup. At that, files read and written during backup are unlikely be read by other processes anytime soon. As a result, node's RAM fills up with useless page cache, while the useful page cache is dropped.

Thus when writing backup software, one needs to make sure to keep the useful page cache on the node. That is, drop page cache generated by backup. This can be done in two ways:

1. Have backup software read and write files in the `O_DIRECT` mode.

Always consider this when creating new backup tools. If you are writing a simple script, this may sometimes be possible as well.

- If you need to copy a large file, e.g., a snapshot, use `dd` with the `O_DIRECT` flag instead of `cp`:

```
# dd if=/path/to/src_file of=/path/to/dst_file bs=8M iflag=nocache oflag=direct
```

- If you need to create an archive and write it somewhere, most often you can avoid page cache generation for the write part if the archiver can write to `stdout`. Just write the archive to disk with `dd`:

```
# archive_and_write_to_stdout | dd of=/path/to/dst_file bs=8M oflag=direct
```

It may not be that easy with the read part, however. For example, `tar` found in the mainstream tree still does not support the `--o_direct` option, although patches exist that implement this functionality. It is like that with `rsync` too. There are patches, pull requests, and even custom builds that add `fadvise` (`FADV_DONTNEED`) support to it, but none of those have been accepted to the `rsync` mainstream tree.

One interesting tool is `nocache`. It tries to minimize the effect of applications on the Linux file system cache. It intercepts `open` and `close` system calls and calls `posix_fadvise` with the `POSIX_FADV_DONTNEED` parameter. For example:

```
# nocache tar cz -0 /path/to/backup_dir | dd of=/path/to/dst_file bs=8M oflag=direct
```

or

```
# nocache rsync -aiv --rsync-path='nocache rsync' /src/ host:/dest/
```

The `--rsync-path` option makes the remote side use a `nocache` wrapper as well.

The `nocache` tool provides a workaround for small and medium-sized files (note that by default it only handles files up to 1MB in size). However by the time `tar`, `rsync`, or another utility closes a large source file and `nocache` calls `fadvise()` to drop it, it is too late, as the file contents have already reached node's page cache, replacing useful data in it. More details are provided in [this post](#).

2. Run backup software in a cgroup with a limited page cache (Virtuozzo Hybrid Server 7).

If your backup tools do not support the `O_DIRECT` file access mode, you can use a feature of Virtuozzo Hybrid Server 7: limiting of page cache for particular memory cgroups (see *Limiting Memory Pressure from Node Processes* on page 46).

2.10.8.2 Limiting Memory Pressure from Node Processes

To limit memory pressure on the hardware node resulting from fast page cache generation by node processes (like backups), one needs to do the following:

1. Create a separate memory cgroup.
2. Place the desired process into the cgroup.
3. Disable `tcache` for the cgroup.
4. Limit page cache for the cgroup (a feature of Virtuozzo Hybrid Server 7).
5. After the process that generates page cache fast ends, remove the memory cgroup.

The easiest way is to use the script `nocache.sh` that performs all these steps.

Usage:

```
# nocache.sh
[-limit LIMIT]
[-pcgroup PARENT_CGROUP]
<command>
```

Where:

- `-limit LIMIT` is the page cache limit in megabytes to apply to the process. The default is 256MB.
- `-pcgroup PARENT_CGROUP` is the parent memory cgroup, inside which the cgroup with the limited page cache will be created. The current memory cgroup is used by default.
- `<command>` is the command with options (or script) that generates page cache.

Examples:

```
# nocache.sh backup_script.sh
```

```
# nocache.sh -limit 128 -pcgroup /machine.slice backup_script.sh
```

If you cannot use the script for some reason, you can perform the same steps manually:

1. Create a separate memory cgroup.

```
# mkdir /sys/fs/cgroup/memory/user.slice/limited_pagecache
```

Note: You can replace `limited_pagecache` with any unique string.

2. Place the desired process with PID `$PID` into the cgroup.

```
# echo $PID >> /sys/fs/cgroup/memory/user.slice/limited_pagecache/tasks
```

Note: You can write a Bash wrapper for the command that creates the page-cache-heavy process. Place the process PID into the cgroup as follows and the children processes will inherit the parent's cgroup:

```
# echo $$ >> /sys/fs/cgroup/memory/user.slice/limited_pagecache/tasks
```

3. Disable `tcache` for the cgroup.

```
# echo 1 > /sys/fs/cgroup/memory/user.slice/limited_pagecache/memory.disable_cleanache
```

4. Limit page cache for the cgroup to `$LIMIT` bytes (a feature of Virtuozzo Hybrid Server 7):

```
# echo $LIMIT > /sys/fs/cgroup/memory/user.slice/limited_pagecache/memory.cache.limit_in_bytes
```

Note: The limit depends on the application. For example, 256MB is enough for `tar + gzip`.

5. After the process that generates page cache fast ends, remove the memory cgroup.

```
# rmdir /sys/fs/cgroup/memory/user.slice/limited_pagecache
```

Note that a cgroup with processes cannot be removed. If on step 2 you put the process PID into the cgroup, first remove it from the cgroup with:

```
# echo $$ >> /sys/fs/cgroup/memory/user.slice/tasks
```

Frequently asked questions:

- **Q:** May my application fail to work due to memory allocation issues if run with `nocache.sh`?

A: No. The wrapper does not limit the size of anonymous memory allowed for the application, so memory allocation will not fail.

- **Q:** May `nocache.sh` affect the performance of my application?

A: It depends on the application. Backup-like programs that read each file only once will not slow down. Applications that read same files two or more times may slow down.

2.11 Managing Templates

A template in Virtuozzo Hybrid Server is a pre-configured virtual machine or container that can be easily and quickly deployed into a fully functional virtual machine or container. Like any normal virtual machine or container, a template contains hardware (virtual disks, peripheral devices) and the operating system. It can also have additional software installed. In fact, the only main difference between a virtual machine or container and a template is that the latter cannot be started.

You can perform the following operations on templates:

- Create a new template.
- List existing templates.
- Create a virtual machine or container from a template.
- Migrate templates between Virtuozzo Hybrid Server servers.
- Store templates on Virtuozzo Storage.

These operations are described in the following subsections in detail.

Note: In addition, see *Using Custom EZ Templates* on page 254 for details on how to manually create and use custom container templates.

2.11.1 Creating Templates

In Virtuozzo Hybrid Server, you can create a template using the `pr1ct1 clone` utility. Making a template may prove useful if you need to create several virtual machines or containers with the same configuration. In this case, your steps can be as follows:

1. Create a virtual machine or container with the required configuration.
2. Make a template on the basis of the created virtual machine or container.

- Use the template to create as many virtual machines or containers as necessary.

For example, to create a template of the virtual machine MyVM, run the following command:

```
# prlctl clone MyVM --name template1 --template
```

This command clones the virtual machine and saves it as the `template1` template. After the template has been successfully created, you can use it for creating new virtual machines.

Note: When creating Microsoft Windows VM templates, use the Sysprep tool with the `/mode:vm` parameter.

2.11.2 Listing Templates

Sometimes, you may need to get an overview of the templates available on your hardware node. For example, this may be necessary if you plan to create a virtual machine or container from a specific template, but do not remember its exact name. In this case, you can use the `prlctl list` command to list all templates on the hardware node and find the one you need:

```
# prlctl list -t
UUID                                DIST      T  NAME
{017bdfd0-b546-4309-90d0-147ce55773f2} centos    VM  centos_tmpl
{92cd331e-0572-46ac-8586-f19b8d029c4d} centos    CT  ct201_tmpl
{fc40e38e-8da4-4b26-bb18-6098ec85f7b4} debian    VM  deb_tmpl
{0dea5912-b114-45a9-bd1a-dc065c1b8e9f} ubuntu    VM  ubuntu_tmpl
{479e66aa-332c-4e3e-975e-b8b6bfc9d2e0} win-2012  VM  w12en_tmpl
```

In this example, 5 templates exist on the server. The information on these templates is presented in the form of a table with the following columns (from left to right): the template ID, the operating system contained in the template, the template type (for a container or virtual machine) and the template name.

2.11.3 Deploying Templates

To deploy a virtual machine or container from a template, use the `--ostemplate` option of the `prlctl create` command. For example, to deploy the virtual machine `MyVMtemplate1` from the template `template1`, run the following:

```
# prlctl create MyVMtemplate1 --ostemplate template1
```

To check that the virtual machine has been successfully created, use the `prlctl list -a` command:

```
# prctl list -a
STATUS      IP_ADDR      NAME
running     10.12.12.101 MyVM
stopped     10.12.12.34  MyVMtemplate1
```

The template itself is left intact and can be used for creating other virtual machines:

```
# prctl list -t
{4ad11c28-9f0e-4086-84ea-9c0487644026} win-2008 template1
{64bd8fea-6047-45bb-a144-7d4bba49c849} rhel      template2
```

2.11.4 Migrating Templates

Migrating virtual machine and container templates between Virtuozzo Hybrid Server servers is similar to migrating virtual machines and containers offline.

- To migrate (move) the template `template1` to the remote server `remoteserver.com`, on the local server run:

```
# prctl migrate template1 remoteserver.com
```

- To migrate (move) the template `template1` from the remote server `remoteserver.com`, on the local server run:

```
# prctl migrate remoteserver.com/template1 localhost
```

The root account is used to log in to the remote server by default, so you will be asked for the root password. You can also provide different credentials (and port) in the format `[<user>[:<passwd>]@]<server>[:<port>]`.

Once migration is complete, the original template is removed from the source server (unless `--clone` is added).

2.11.5 Storing Templates on Virtuozzo Storage

Starting from Virtuozzo Hybrid Server 7.0.7 (Update 7), you can store container and virtual machine templates in shared directories of Virtuozzo Storage clusters. These templates will be available to any server participating in the cluster.

To place a template on Virtuozzo Storage, do as follows on the cluster node where the source container or VM is located:

1. Create a template. For example:


```
# prlctl clone MyVM --name template1 --template
```

2. Move this template to the `vmtemplates` directory located on Virtuozzo Storage:

- for Virtuozzo Storage with CLI management, the path is `/vstorage/<cluster_name>/vmtemplates`, e.g.:

```
# prlctl move template1 --dst /vstorage/vstor1/vmtemplates
```

- for Virtuozzo Storage with GUI management, the path is `/mnt/vstorage/vmtemplates`, e.g.:

```
# prlctl move template1 --dst /mnt/vstorage/vmtemplates
```

Within five minutes, the template will be autodetected by the `prlctl` utility. You can check template availability by listing templates on another Virtuozzo Storage server. For example:

```
# prlctl list -t | grep template1
{4ad11c28-9f0e-4086-84ea-9c0487644026} win-2008 template1
```

Once the template is available throughout the cluster, you can start creating containers or VMs based on it on any cluster node as described in [Deploying Templates](#) on page 49.

2.12 Managing Snapshots

In Virtuozzo Hybrid Server, you can save the current state of a virtual machine or container by creating a snapshot. You can then continue working in your virtual machine or container and return to the saved state any time you wish. Snapshots may be useful in the following cases:

- Configuring applications with a lot of settings. You may wish to check how settings work before applying them to the application. So, before you start experimenting, you create a snapshot.
- Participating in large-scale development projects. You may wish to mark development milestones by creating a snapshot for each. If anything goes wrong, you can easily revert to the previous milestone and resume the development.

In Virtuozzo Hybrid Server, you can create, list, revert to, and delete snapshots. These operations are described in the following subsections.

2.12.1 Creating Snapshots

To create a snapshot of a virtual machine or container, use the `prctl snapshot` command.

2.12.1.1 Creating Virtual Machine Snapshots

To create a snapshot of the virtual machine `MyVM`, do the following:

```
# prctl snapshot MyVM
...
The snapshot with ID {12w32198-3e30-936e-a0bbc104bd20} has been successfully created.
```

A newly created snapshot is saved to the `/vz/vmprivate/<UUID>/snapshots/<snapshot_ID>/config.pvs` file, where `<UUID>` is the corresponding virtual machine ID and `<snapshot_ID>` is a unique snapshot ID. In the above example, the snapshot with the ID `{12w32198-3e30-936e-a0bbc104bd20}` is saved to the file `/vz/vmprivate/d35d28e5-11f7-4b3f-9065-8fef6178bc5b/snapshots/{12w32198-3e30-936e-a0bbc104bd20}/config.pvs`.

```
# ls /vz/vmprivate/d35d28e5-11f7-4b3f-9065-8fef6178bc5b/snapshots/
{12w32198-3e30-936e-a0bbc104bd20}/config.pvs
```

Snapshot IDs are needed to switch to and delete snapshots.

When creating a snapshot, you can also set its name and description:

```
# prctl snapshot MyVM -n Clean_System -d "This snapshot was created right after \
installing Windows XP."
...
The snapshot with ID {0i8798uy-1eo0-786d-nn9ic106b9ik} has been successfully created.
```

You can view the set name and description in the `/vz/vmprivate/d35d28e5-11f7-4b3f-9065-8fef6178bc5b/snapshots/{12w32198-3e30-936e-a0bbc104bd20}/snapshot.xml` file.

2.12.1.2 Creating Container Snapshots

To create a snapshot of the container `MyCT`, do the following:

```
# prctl snapshot MyCT
...
The snapshot with ID {08ddd014-7d57-4b19-9a82-15940f38e7f0} has been successfully \
created.
```

A newly created snapshot is saved to the `/vz/private/<UUID>/dump/<snapshot_ID>.ve.conf` file, where `<UUID>` is the container UUID and `<snapshot_ID>` is a snapshot ID. In the example above, the snapshot with the ID

{08ddd014-7d57-4b19-9a82-15940f38e7f0} is saved to the file

/vz/private/26bc47f6-353f-444b-bc35-b634a88dbbcc/dump/{08ddd014-7d57-4b19-9a82-15940f38e7f0}.ve.conf.

```
# ls /vz/private/26bc47f6-353f-444b-bc35-b634a88dbbcc/dump
{08ddd014-7d57-4b19-9a82-15940f38e7f0}.ve.conf
```

Snapshot IDs are needed to switch to and delete snapshots.

When creating a snapshot, you can also set its name and description:

```
# prlctl snapshot MyCT -n Clean_System -d "This snapshot was created right after \
installing Windows XP."
...
The snapshot with ID {e78bb2b8-7a99-4c8b-ab9a-491a47648c44} has been successfully \
created.
```

The set name and description are stored in the /vz/private/<UUID>/Snapshots.xml file.

2.12.1.3 Snapshot Branching

Snapshot branches can be useful for working with, testing or comparing similar configurations. A snapshot branch is created when you do the following:

1. Create several snapshots.
2. Revert to one of the snapshots.
3. Make changes to the virtual machine or container.
4. Create a snapshot.

In this case, the newly created snapshot will start a new branch based on the snapshot from Step 2.

2.12.1.4 Restrictions and Recommendations

- If a virtual machine or snapshot name or description contains spaces, enclose it in quotation marks (e.g., "Windows XP") in the `prlctl` command.
- Before creating a snapshot, it is recommended that you finish any installations, downloads, and stop writing to external devices. You should also complete or cancel any transactions performed via the virtual machine in external databases.
- You cannot create snapshots for containers with enabled NFS server feature.

2.12.2 Listing Snapshots

To list all snapshots of a particular virtual machine or container, use the `prlctl snapshot-list` command. For example, to check all current snapshots of the virtual machine `MyVM`, run this command:

```
# prlctl snapshot-list MyVM
PARENT_SNAPSHOT_ID          SNAPSHOT_ID
                             {989f3415-3e30-4494-936e-a0bbc104bd20}
{989f3415-3e30-4494-936e-a0bbc104bd20}  *{063615fa-f2a0-4c14-92d4-4c935df15840}
```

This command shows that the virtual machine `MyVM` has two snapshots. The snapshot with ID `{063615fa-f2a0-4c14-92d4-4c935df15840}` is based on the snapshot with ID `{989f3415-3e30-4494-936e-a0bbc104bd20}`, i.e. the former is a child of the latter. The asterisk marks the current snapshot.

To view the relationships between snapshots, use the `-t` option:

```
# prlctl snapshot-list MyVM -t
_{989f3415-3e30-4494-936e-a0bbc104bd20}_{063615fa-f2a0-4c14-92d4-4c935df15840}\
*{712305b0-3742-4ecc-9ef1-9f1e345d0ab8}
```

The command output shows you that currently two branches exist for the virtual machine `MyVM`. The snapshot with ID `{989f3415-3e30-4494-936e-a0bbc104bd20}` is the base for these branches.

To get detailed information on a particular snapshot, use the `-i` option with the snapshot ID:

```
# prlctl snapshot-list MyVM -i {063615fa-f2a0-4c14-92d4-4c935df15840}
ID: {063615fa-f2a0-4c14-92d4-4c935df15840}
Name: Clean_System
Date: 2012-07-22 22:39:06
Current: yes
State: poweroff
Description: <![CDATA[This snapshot was created right after installing Windows 7]]>
```

The `prlctl snapshot-list` command with the `-i` option displays the following information about snapshots:

Field	Description
ID	ID assigned to the snapshot.
Name	Name assigned to the snapshot.
Date	Date and time when the snapshot was created.
Current	Denotes that this is the current snapshot of the virtual machine.
State	State the virtual machine was in at the time you took the snapshot.
Description	The description set for the snapshot.

2.12.3 Reverting to Snapshots

To revert to a snapshot, use the `prlctl snapshot-switch` command. When you revert to a snapshot, the current state of the virtual machine or container is discarded, and all changes made to the system since the previous snapshot are lost. So, before reverting, you may want to save the current state by creating a new snapshot (see *Creating Snapshots* on page 52).

The `prlctl snapshot-switch` command requires the virtual machine or container name and the snapshot ID as arguments, for example:

```
# prlctl snapshot-switch MyVM --id {cedbc4eb-dee7-42e2-9674-89d1d7331a2d}
```

In this example, you revert to the snapshot `{cedbc4eb-dee7-42e2-9674-89d1d7331a2d}` for the virtual machine `MyVM`.

2.12.4 Deleting Snapshots

To delete unneeded snapshots of virtual machines or containers, use the `prlctl snapshot-delete` command. For example:

```
# prlctl snapshot-delete MyVM --id {903c12ea-f6e6-437a-a2f0-a1d02eed4f7e}
```

When you delete a parent snapshot, child snapshots are not deleted, and the information from the former is merged into the latter.

2.13 Migrating Virtual Machines and Containers

To facilitate hardware upgrades and load balancing between multiple hosts, Virtuozzo Hybrid Server enables you to migrate virtual machines and containers either between Virtuozzo Hybrid Server 7 servers or from Virtuozzo 6 to Virtuozzo Hybrid Server 7 servers. Virtual machines and containers stored in a Virtuozzo Storage cluster are migrated between cluster nodes without data copying, which significantly reduces migration time.

Before migration, make sure the following requirements are met:

- The destination server has enough hard disk space to store the resulting virtual machine or container

and enough memory and CPU power to run it.

- A stable network connection is provided between the source and destination servers.
- A direct SSH connection on port 22 is allowed between the source and destination servers.
- The source and destination servers must have exactly the same names for bridged networks used by virtual machines or containers.

Note the general restrictions that are applied to all types of migration:

- Migration of virtual machines with snapshots is not supported (any existing snapshots must be deleted). For more information, see *Managing Snapshots* on page 51.
- Migration of virtual machines with linked clones or linked clones of VMs is not supported. For more information, see *Creating Linked Clones* on page 28.
- Migration of containers to virtual machines is not supported.
- The `pmigrate` utility is not supported.

Virtuozzo Hybrid Server allows you to perform two types of migration between Virtuozzo Hybrid Server servers:

- Offline migration for stopped and suspended containers and virtual machines.
- Online (live) migration for running containers and running and paused virtual machines. Containers and virtual machines may be located on Virtuozzo Storage or local storage.

Both types are described in the sections below.

2.13.1 Offline Migration of Virtual Machines and Containers

Offline migration implies copying all files of a virtual machine or container from one server to another over the network.

You can migrate VMs and containers both to and from a remote server. For example, to move a VM to a remote server, run this command on the local server:

```
# prlctl migrate MyVM remoteserver.com
```

To move a VM from a remote server, run this command the local server:

```
# prlctl migrate remoteserver.com/MyVM localhost
```

The root account is used to log in to the remote server by default, so you will be asked for the root password. You can also provide different credentials (and port) in the format [`<user>[:<passwd>]@<server>[:<port>]`]. Alternatively, you can set up RSA authentication and not use passwords at all (see [Setting Up RSA Authentication Between Nodes](#) on page 297).

If you want to place the migrated virtual environment in a custom directory on the destination server, specify the full path to that directory in the `--dst=<custom_path>` option. The resulting path to the migrated virtual environment files will be `<custom_path>/<VE_UUID>`.

By default, once migration is complete:

- the original virtual machine is removed from the source server,
- the `.migrated` suffix is added to the names of the original container's private area and configuration file on the source server.

Adding the `--clone` option to the `prlctl migrate` command enables you to skip the two default actions above. That is, to keep the original VM and not to add the `.migrated` suffix to container's private area and configuration file. The clone will have a different UUID, MAC address, SID (for Windows-based VMs only; if the `--changesid` option is specified), and offline management disabled.

Note: For Windows 2019 VMs, use Microsoft's Sysprep tool instead of the `--changesid` option.

Migration implies transferring large amounts of data between servers which can take considerable time. To reduce the amount of data to be transferred, Virtuozzo Hybrid Server has compression enabled by default. Compression consumes additional server resources and can be disabled if necessary with the `--no-compression` option.

2.13.2 Live Migration of Virtual Machines and Containers

The process of migrating virtual machines and containers live is as follows:

1. Once you start the migration, Virtuozzo Hybrid Server checks whether the destination server meets all the migration requirements and the virtual machine or container can be migrated to it.
2. All virtual memory and disks of the virtual machine or container are migrated to the destination server.
3. The virtual machine or container on the source server is suspended.
4. The changed memory pages and virtual disk blocks, if any, are migrated to the destination server.

5. The virtual machine or container is resumed on the destination server.

The virtual machine or container continues running during steps 1 and 2 and is not available to the user during steps 3-5. But since the amount of memory pages and virtual disk blocks changed during step 2 is small, the downtime is almost imperceptible.

After migration, the relocated virtual machine or container may not be accessible over the network for several minutes due to network equipment reconfiguration (for example, as switches are updating their dynamic VLAN membership tables).

When performing live migration, take into account the following requirements and restrictions:

- The system time on the source and destination servers should be synchronized, for example, by means of NTP (<http://www.ntp.org>). The reason is that certain processes running in virtual machines and containers may rely on system time being steady and might behave unpredictably when resumed on a destination server where time is different.

Note: In Virtuozzo Hybrid Server 7, time synchronization via NTP is enabled by default using the `chronyd` service. If you want to use `ntpd` or `ntpd`, stop and disable `chronyd` first.

- The network must support data transfer rates of at least 1 Gbps.
- The source and destination servers must belong to the same subnetwork.
- The CPUs on the source and destination servers must be manufactured by the same vendor. The CPU features of the destination server must be the same or exceed those of the source server. If only some of the CPU features are common between the servers, you can create a CPU pool to guarantee live migration between them. For more information, see *Managing CPU Pools* on page 206.

Alternatively, you can mask individual CPU features with the `prlsrvctl set --cpu-features-mask` command (refer to the `prlsrvctl man` page).

Note: Both CPU pools and masking of individual CPU features only work on Intel Ivy Bridge and newer CPUs. They do not work on AMD processors.

- Container live migration is only possible between physical nodes having the `cpuid_override` feature in `/proc/vz` (this condition does not apply to the nested installations where containers run inside a virtual machine). For example:


```
# ls -l /proc/vz | grep cpuid_override
-rw-r--r-- 1 root root 0 Feb  3 15:38 cpuid_override
```

- Virtual machine and container disks can be located on local disks and Virtuozzo Storage.
- Live migration is not supported for virtual machines and containers with open `prlctl enter` sessions and containers with IPsec connections.
- Containers with NFS clients inside can be migrated only if the following conditions are met:
 - Block and character device files shared over NFS are not opened.
 - Remote file locking and over-mounted NFS file systems are not used simultaneously.

Note: Migration of local file locks is supported only for NFS version 3 as it has native support of such locks.

- Live migration is not supported for containers with enabled NFS server feature.

The command that launches online migration does not differ from that used for offline migration (see [Offline Migration of Virtual Machines and Containers](#) on page 56). For example, to move a running VM to a remote server, run this command on the local server:

```
# prlctl migrate MyVM remoteserver.com
```

For increased security during live migration, Virtuozzo Hybrid Server provides connection tunneling between the source and destination servers. Tunneling increases migration time. If you do not need a secure tunnel between servers, you can speed up VM live migration by disabling connection tunneling with the `--no-tunnel` option. Tunnelless container migration is not supported. To use the option, configure the firewall of the destination server to allow incoming connections on any port on the corresponding network interface.

2.13.3 Migrating VMs and Containers from Virtuozzo 6 to Virtuozzo Hybrid Server 7

You can migrate older Virtuozzo 6 virtual machines (online) and containers (offline only) to Virtuozzo Hybrid Server 7 servers. During migration, such VMs and containers will be converted in the Virtuozzo Hybrid Server 7 format. In particular, VM devices will be replaced by those supported in Virtuozzo Hybrid Server 7 (for a list of VM hardware supported in Virtuozzo Hybrid Server 7, see [Virtual Machine Hardware](#) on page 8).

Before migrating Virtuozzo 6 virtual machines, take into account the following requirements and restrictions:

- The hardware node must be running the latest version of Virtuozzo 6 (or at least 6.0.11-3466).
- The recommended and highest-supported destination hardware node version is Virtuozzo Hybrid Server 7.5 Update 4.
- The VM must have a guest OS installed.
- The VM must not have snapshots.
- The VM must be running.
- The container must be stopped.
- Windows VMs may have Virtuozzo guest tools, and Linux VMs must have installed kernel headers and kernel-devel packages.
- The system time must be synchronized on the source and destination servers, for example, by means of NTP (<http://www.ntp.org>). The reason is that certain processes running in virtual machines may rely on system time being steady and might behave unpredictably when resumed on a destination server where time is different.
- The network must support data transfer rates of at least 1Gbps.
- The source and destination servers must belong to the same subnetwork.
- The CPUs on the source and destination servers must be manufactured by the same vendor, and the CPU capabilities of the destination server must be the same or exceed those on the source server.
- VM disks can be located on local disks, shared NFS and GFS2 storages, and iSCSI raw devices.
- Live migration is not supported for both VMs and containers. However, you can use hot VM migration, which involves the following:
 - Sending disk data in the background while the VM runs and stopping the VM to finish the migration and conversion on the destination server.
 - Restarting the VM with a guest OS on the destination server.

To migrate a running VM or a stopped or suspended container from Virtuozzo 6 to Virtuozzo Hybrid Server 7, run the following command on the Virtuozzo 6 server.

```
# prlctl migrate <VM_or_CT_name> root@<VZ7_server_IP_address_or_hostname>
```

Note: If you need to migrate a container from a Virtuozzo 6 server with local storage to a Virtuozzo Storage cluster based on Virtuozzo Hybrid Server 7, use the following command: `prlctl migrate <CT_name> root@<VZ7_server_IP_address_or_hostname>/<CT_UUID>`.

Migration from Virtuozzo 6 to Virtuozzo Hybrid Server 7 implies VM and container downtime that depends on network bandwidth, virtual machine RAM size, and server load. To reduce downtime, it is recommended to at least perform migration when the server load is minimal.

During migration, a virtual machine is copied to the destination server and converted to the Virtuozzo Hybrid Server 7 format. After a successful migration, the original Virtuozzo 6 VM is unregistered and the `.migrated` suffix is added to its directory name. If conversion fails, the migrated VM is deleted from the destination server and you can try again. If the second attempt also fails, you need to enable a legacy VM debug mode on the destination Virtuozzo Hybrid Server 7 server (see [Enabling Legacy VM Debug Mode](#) on page 292), make another migration attempt, send the problem report, and contact the technical support team. With the debug mode enabled, the migrated VM will not be deleted from the destination Virtuozzo Hybrid Server 7 server after conversion failure. It will remain stopped or running with disabled network to let the technical support team study the memory dump and find out the reason for failure.

Note the following:

- Network settings configured from inside Windows VMs currently cannot be preserved during migration of such VMs.
- VNC is enabled in VMs before migration so that the technical support team can locate the issue if migration fails.
- Parallels guest tools, if such exist, are automatically removed from VMs during conversion, and then Virtuozzo guest tools are installed.
- Containers with templates unsupported in Virtuozzo Hybrid Server 7 (i.e. not listed in [Virtuozzo Hybrid Server Supported Guest Operating Systems Guide](#)) cannot be reinstalled with the `prlctl reinstall` command after migration to Virtuozzo Hybrid Server 7.

2.14 Performing Container-Specific Operations

This section describes operations specific to Virtuozzo Hybrid Server containers.

2.14.1 Reinstalling Containers

Reinstalling a container may help if any required container files have been inadvertently modified, replaced, or deleted, resulting in container malfunction. You can reinstall a container by using the `prlctl reinstall` command that creates a new container private area from scratch according to its configuration file and relevant OS and application templates. For example:

```
# prlctl reinstall MyCT
```

To keep the personal data from the old container, the utility also copies the old private area contents to the `/vz/root/<UUID>/old` directory of the new private area (unless the `--no-backup` option is given). This directory may be deleted after you copy the personal data where you need.

The `prlctl reinstall` command retains user credentials base, unless the `--resetpwb` option is specified.

2.14.1.1 Customizing Container Reinstallation

The default reinstallation, as performed by the `prlctl reinstall` command, creates a new private area for the broken container as if it were created by the `prlctl create` command and copies the private area of the broken container to the `/old` directory in the new private area so that no file is lost. There is also a possibility of deleting the old private area altogether without copying or mounting it inside the new private area, which is done by means of the `--no-backup` option. This way of reinstalling corrupted containers might in certain cases not correspond exactly to your particular needs. It happens when you are accustomed to creating new containers in some other way than just using the `prlctl create` command. For example, you may install additional software licenses into new containers, or anything else. In this case you would naturally like to perform reinstallation in such a way so that the broken container is reverted to its original state as determined by you, and not by the default behavior of the `prlctl create` command.

To customize reinstallation, you should write your own scripts determining what should be done with the container when it is being reinstalled, and what should be configured inside the container after it has been reinstalled. These scripts should be named `vps.reinstall` and `vps.configure`, respectively, and should be located in the `/etc/vz/conf` directory on the hardware node. To facilitate your task of creating customized scripts, the container software is shipped with sample scripts that you may use as the basis of your own

scripts.

When the `prlctl reinstall <UUID>` command is called, it searches for the `vps.reinstall` and `vps.configure` scripts and launches them consecutively. When the `vps.reinstall` script is launched, the following parameters are passed to it:

Option	Description
<code>--veid</code>	Container UUID.
<code>--ve_private_tmp</code>	The path to the container temporary private area. This path designates where a new private area is temporarily created for the container. If the script runs successfully, this private area is mounted to the path of the original private area after the script has finished.
<code>--ve_private</code>	The path to the container original private area.

You may use these parameters within your `vps.reinstall` script.

If the `vps.reinstall` script finishes successfully, the container is started, and the `vps.configure` script is called. At this moment the old private area is mounted to the `/old` directory inside the new one irrespective of the `--no-backup` option. This is done in order to let you use the necessary files from the old private area in your script, which is to be run inside the running container. For example, you might want to copy some files from there to regular container directories.

After the `vps.configure` script finishes, the old private area is either dismounted and deleted or remains mounted depending on whether the `--no-backup` option was provided.

If you do not want to run these reinstallation scripts and want to stick to the default `prlctl reinstall` behavior, you may do either of the following:

- Remove the `vps.reinstall` and `vps.configure` scripts from the `/etc/vz/conf` directory, or at least rename them.
- Modify the last line of the `vps.reinstall` script so that it would read `exit 128` instead of `exit 0`.

The exit code 128 tells the utility not to run the scripts and to reinstall the container with the default behavior.

2.14.2 Repairing Containers

Note: The repair mode does not work for virtual machines.

If a container malfunctions, starting it in the repair mode may help you fix it. Do the following:

1. Start the broken (original) container as follows:

```
# prlctl start <original_CT> --repair
```

A temporary container will be created with the same name, parameters, and user accounts as the original container. The temporary container will start and the original container's root will be mounted to /repair in it.

Note: Virtuozzo PowerPanel may not detect this operation.

2. Invite the container owner to log in to the temporary container as they would log in to the original one. The container owner can now save the critical data from /repair or try to fix the container, with or without your help.

Important: Warn the container owner to never save the critical data to the temporary container. It will be automatically destroyed when stopped.

3. When the critical data has been saved or container has been fixed, stop the temporary container:

```
# prlctl stop <original_CT>
```

The original container's root will be unmounted and the temporary container will be destroyed.

4. If the original container has been fixed, start it so the owner can log in to it as usual. If the container owner saved the critical data and the original container cannot be fixed, reinstall it and invite the owner to upload the saved data back.

2.14.3 Enabling VPN for Containers

Virtual Private Network (VPN) is a technology which allows you to establish a secure network connection even over an insecure public network. Setting up a VPN for a separate container is possible via the TUN/TAP device. To allow a particular container to use this device, do the following:

1. Make sure the `tun.o` module is already loaded before Virtuozzo Hybrid Server is started:

```
# lsmod | grep 'tun'
```

2. Allow the container to use the TUN/TAP device:

```
# vzctl set MyCT --devnodes net/tun:rw --save
```

Configuring the VPN properly is a common Linux administration task, which is out of the scope of this guide. Some popular Linux software for setting up a VPN over the TUN/TAP driver includes [Virtual TUNnel](#) and [OpenVPN](#).

2.14.4 Setting Up NFS Server in Containers

To set up an NFS server in a container, do the following:

1. Make sure the `rpcbind`, `nfsd`, and `nfslock` services are installed in the container.
2. Enable the NFS server feature for the container by running the `prctl set --features nfsd:on` command on the hardware node. For example:

```
# prctl set MyCT --features nfsd:on
```

If the container is running, stop it first. After enabling the feature, restart the container.

Note:

- You cannot create snapshots of containers with the enabled NFS server feature.
 - When performing a live migration of containers with the enabled NFS server feature, the NFS service is stopped before dump and restarted after restore.
 - Custom services that depend on the NFS server feature (i.e., that stop with NFS but are not restarted when the NFS server is up again) will be stopped after migration.
-

3. Start the rpcbind service in the container.

```
# service rpcbind start
Starting rpcbind: [ OK ]
```

4. Start the nfs and nfslock services in the container.

```
# service nfs start
Starting NFS services: [ OK ]
Starting NFS quotas: [ OK ]
Starting NFS mountd: [ OK ]
Starting NFS daemon: [ OK ]
# service nfslock start
Starting NFS statd: [ OK ]
```

You can now set up NFS shares in the configured container.

2.14.5 Mounting NFS Shares on Container Start

If you configured an NFS share in the `/etc/fstab` file of a CentOS or RHEL-based container, and you need this NFS share to be mounted on container start, enable autostart for the `netfs` service with the `chkconfig netfs on` command.

2.14.6 Managing Container Virtual Disks

Note: You can manage virtual disks of both stopped and running containers.

Virtuozzo Hybrid Server allows you to perform the following operations on container virtual disks:

- Add new virtual disks to containers.
- Configure the virtual disk properties.
- Remove virtual disks from containers.

2.14.6.1 Adding Virtual Disks to Containers

New containers are created with a single virtual hard disk, but you can add more disks as follows:

- Attach a new or existing image file that emulates a hard disk drive.

- Attach a physical hard disk of the host server.

2.14.6.1.1 Using Image Files

You can either attach an existing image to the container or create a new one and keep it at a custom location, e.g., on a regular disk or in a Virtuozzo Storage cluster. Thus you can create more flexible containers with the operating system on a fast SSD and user data on redundant Virtuozzo Storage.

To create a new image file and add it to a container as a virtual hard disk, use the `prlctl set --device-add hdd` command. For example:

```
# prlctl set MyCT --device-add hdd --size 100G --mnt /userdisk
```

Note: If you omit the `--mnt` option, the disk will be added unmounted.

This command adds to the configuration of the container `MyCT` a virtual hard disk with the following parameters:

- Name: `hdd<N>` where `<N>` is the next available disk index.
- Default image location: `/vz/private/<CT_UUID>/harddisk<N>.hdd` where `<N>` is the next available disk index.
- Size: 102400 MB.
- Mount point inside the container `MyCT`: `/userdisk`. A corresponding entry is also added to container's `/etc/fstab` file.

To attach an existing image file to a container as a virtual hard disk, specify the path to the image file with the `--image` option. For example:

```
# prlctl set MyCT --device-add hdd --image /hdd/MyCT.hdd --size 100G --mnt /userdisk
```

2.14.6.1.2 Attaching Physical Hard Disks

You can attach to a container any physical block device available on the physical server, whether it is a local hard disk or an external device connected via Fibre Channel or iSCSI.

Note: A physical block device must be formatted and have only one file system before it can be attached to

a container.

You will need to specify the path to the device, which you can find out the with the `prlsrvctl info` command. For example:

```
# prlsrvctl info
...
Hardware info:
  hdd  WDC WD1002FAEX-0 ATA (/dev/sda2)  '/dev/disk/by-id/lvm-pv-uuid-RDYrbU-<...>'
  hdd  WDC WD1002FAEX-0 ATA (/dev/sda)   '/dev/disk/by-id/wwn-0x50014ee25a3df4dc'
  cdrom PIONEER DVD-RW DVR-220          '/dev/sr0'
  net   eth0                             'eth0'
  serial /dev/ttyS0                          '/dev/ttyS0'
...
```

All physical devices available on the physical server are listed in the **Hardware info** section.

Once you know the path to the physical block device, you can attach it to a container with the `prlctl set --device-add hdd --device` command. For example:

```
# prlctl set MyCT --device-add hdd --device '/dev/disk/by-id/wwn-0x50014ee25a3df4dc' \
--mnt /userdisk
```

Note: If you omit the `--mnt` option, the disk will be added unmounted.

This command adds to the configuration of the container `MyCT` a virtual hard disk with the following parameters:

- Name: `hdd<N>` where `<N>` is the next available disk index.
- Path to the device: `/dev/disk/by-id/wwn-0x50014ee25a3df4dc` where `wwn-0x50014ee25a3df4dc` is a storage device unique identifier.
- Mount point inside the container: `/userdisk`. A corresponding entry is also added to container's `/etc/fstab` file.

Note:

1. Before migrating containers with external hard drives, make sure that corresponding physical disks exist on the destination server and are available by the same name (for this purpose, use persistent naming, for example, via `/dev/disk/by-id/`),

2. During container backup operations, physical disks connected to the container are not backed up.
3. If you use multipath from a system to a device, it is recommended to use the `user_friendly_names no` feature so that multipath devices have names persistent across all nodes in a cluster.

2.14.6.2 Configuring Container Virtual Disks

To configure the parameters of a virtual disk attached to a container, use the `prlctl set --device-set` command.

You will need to specify the disk name, which you can find out the with the `prlctl list -i` command. For example:

```
# prlctl list -i MyCT | grep "hdd"
hdd0 (+) scsi:0 image='/vz/private/9fd3eee7-70fe-43e3-9295-1ab29fe6dba5/root.hdd' type='expanded' \
10240Mb mnt=/ subtype=virtio-scsi
hdd1 (+) scsi:1 real='/dev/disk/by-id/wwn-0x50014ee25a3df4dc' mnt=/userdisk subtype=virtio-scsi
```

Once you know the virtual device name, you can configure its properties. For example, to change the type of the virtual disk `hdd0` in the container `MyCT` from SCSI to IDE, execute:

```
# prlctl set MyCT --device-set hdd0 --iface ide
```

To check that the virtual disk type has been changed, use the `prlctl list -i` command. For example:

```
# prlctl list -i MyCT | grep "hdd"
hdd0 (+) ide:0 image='/vz/private/9fd3eee7-70fe-43e3-9295-1ab29fe6dba5/root.hdd' type='expanded' \
10240Mb mnt=/
```

2.14.6.3 Deleting Virtual Disks from Containers

You can delete a virtual hard disk from a container with the `prlctl set --device-del` command.

You will need to specify the disk name, which you can find out the with the `prlctl list -i` command. For example:

```
# prlctl list -i MyCT | grep "hdd"
hdd0 (+) scsi:0 image='/vz/private/9fd3eee7-70fe-43e3-9295-1ab29fe6dba5/root.hdd' type='expanded' \
10240Mb mnt=/ subtype=virtio-scsi
hdd1 (+) scsi:1 real='/dev/disk/by-id/wwn-0x50014ee25a3df4dc' mnt=/userdisk subtype=virtio-scsi
```

Once you know the virtual device name, you can remove it from your container. For example, to remove the virtual disk `hdd1` from the container `MyCT`, execute:

```
# prlctl set MyCT --device-del hdd1
```

2.14.7 Restarting Containers

You can restart containers from the inside using typical Linux commands, e.g., `reboot` or `shutdown -r`. Restarting is handled by the `vzeventd` daemon.

If necessary, you can keep containers from starting again after the `reboot` command has been executed from the inside, as follows:

- To disable restarting for a specific container, add the `ALLOWREBOOT="no"` line to the container configuration file (`/etc/vz/conf/<UUID>.conf`).
- To disable restarting globally for all containers on the server, add the `ALLOWREBOOT="no"` line to the global configuration file (`/etc/vz/vz.conf`).
- To disable restarting globally except for specific containers, add the `ALLOWREBOOT="no"` line to the global configuration file (`/etc/vz/vz.conf`) and explicitly specify `ALLOWREBOOT="yes"` in the configuration files of the respective containers.

As a result, a container with the `ALLOWREBOOT` option set to `"no"` retains the mounted status after the `reboot` command has been executed inside the container.

2.14.8 Creating SimFS-based Containers

In Virtuozzo Hybrid Server 7, the `simfs` layout is based on bindmounts. When a `simfs`-based container is started, its private area is bindmounted to the root container area.

To create a `simfs` container:

1. Set `VEFSTYPE=simfs` in `/etc/vz/vz.conf`.
2. Run `prlctl create <CT_name>`.

The limitations of `simfs` in Virtuozzo Hybrid Server 7 are:

1. No support for first- or second-level quotas.
2. No support for live migration of `simfs`-based containers.

2.14.9 Bind-Mounting Host Directories Inside Containers

You can bind-mount a host directory inside a container using `vzctl`:

```
# vzctl set <CT> --bindmount_add <host_dir>:<CT_dir> --save
```

For example, to bind-mount a host directory `/vz/host_dir` to a container directory `/home/ct_dir`, run

```
# vzctl set MyCT --bindmount_add /vz/host_dir:/home/ct_dir --save
```

Such a bind mount is permanent and will be saved after a container restart and during container backup.

To remove a bind mount, use `--bindmount_del`:

```
# vzctl set MyCT --bindmount_del /home/ct_dir
```

Note the following:

- Containers with bind mounts cannot be migrated.
- If the container is stopped, the host directory will be bind-mounted or unmounted on container start. If the container is running, the host directory will be bind-mounted or unmounted immediately, no container restart will be required.
- If the specified container directory does not exist, it will be created on bind-mount. It will not, however, be deleted on unmount.
- If the specified container directory is not empty on bind-mount, it will be replaced by the host directory until it is unmounted.
- The bind-mounted directory will obey file system permissions. A user with insufficient permissions inside the container will not be able to access it.

2.14.10 Checking Consistency of Container File System

You can check the consistency of a container's file system from the host. The container can be stopped or running to avoid downtime. Use the command

```
# ploop fscheck <CT_home>/root.hdd/DiskDescriptor.xml
```

Where `<CT_home>` is the container's directory.

For example, for a container `MyCT`:

```
# prlctl list -i MyCT | grep Home
Home: /vz/private/788aff6e-cb2f-4dec-9d61-ee3105fc90d7
# ploop fscheck /vz/private/788aff6e-cb2f-4dec-9d61-ee3105fc90d7/root.hdd/DiskDescriptor.xml
<...>
Running: fsck.ext4 -f -n /dev/ploop58356p1
<...>
No error found on /dev/ploop58356p1
```

2.15 Performing Virtual Machine-Specific Operations

This section focuses on operations specific to virtual machines.

2.15.1 Pausing Virtual Machines

Pausing a running virtual machine releases the resources, such as RAM and CPU, currently used by this virtual machine. The released resources can then be used by the hardware node or other running virtual machines and containers.

To pause a virtual machine, you can use the `prlctl pause` command. For example, the following command pauses the virtual machine `MyVM`:

```
# prlctl pause MyVM
```

You can check that the virtual machine has been successfully paused by using the `prlctl list -a` command:

```
# prlctl list -a
STATUS IP_ADDR      NAME
running 10.10.10.101   MyCT
paused  10.10.10.201    MyVM
```

The command output shows that the virtual machine `MyVM` is paused at the moment. To continue running this virtual machine, execute this command:

```
# prlctl start MyVM
```

2.15.2 Managing Virtual Machine Devices

Virtuozzo Hybrid Server allows you to manage the following virtual machine devices:

- hard disk drives
- CD/DVD-ROM drives
- floppy disk drives
- network adapters
- serial ports
- USB controllers

When managing devices, keep in mind the following limits for VM devices:

IDE Devices	4 IDE devices (virtual disks or CD/DVD-ROM drives)
SCSI Devices	15 SCSI devices (virtual disks or CD/DVD-ROM drives)
VirtIO Devices	15 VirtIO virtual disks
Floppy disk drive	1 floppy disk drive
Network Interfaces	15 virtual NICs
Serial (COM) Ports	4 serial (COM) ports
USB controller	1 USB controller

The main operations you can perform on VM devices are:

- Add a new device to the virtual machine.
- Configure device properties.
- Remove a device from the virtual machine.

2.15.2.1 Adding New Devices

This section provides information on adding new devices to your virtual machines. You can add new virtual devices to your virtual machine using the `prlctl set` command. The options responsible for adding particular devices are listed in the following table:

Option	Description
hdd	<p>Adds a new hard disk drive to a virtual machine. You can either connect an existing image to the virtual machine or create a new one.</p> <hr/> <p>Note: When detaching and adding back an existing disk, the UUID of the device changes. Thus, it may require additional reconfiguration for guest OSs sensitive to the disk UUID modification.</p> <hr/> <p>SCSI and VirtIO hard disks can be added to both running and stopped VMs; IDE disks can only be added to stopped VMs.</p>
cdrom	Adds a new CD/DVD-ROM drive to the virtual machine that must be stopped first.
net	Adds a new network adapter to a stopped virtual machine.
fdd	Adds a new floppy disk drive to a stopped virtual machine.
serial	Adds a new serial port to a stopped virtual machine.
usb	Adds a new USB controller to a stopped virtual machine.

For example, you can execute the following command to add a new virtual disk to the virtual machine MyVM:

```
# prlctl set MyVM --device-add hdd
Creating hdd1 () scsi:1 image='/vz/vmprivate/d35d28e5-11f7-4b3f-9065-8fef6178bc5b/ \
harddisk1.hdd' type='expanded' 65536Mb subtype=virtio-scsi
...
```

This command creates a new virtual disk with the following default parameters:

- Name: hdd1
- Disk type: SCSI
- Disk subtype: VirtIO SCSI
- Image file name and location: /vz/vmprivate/d35d28e5-11f7-4b3f-9065-8fef6178bc5b/harddisk1.hdd
- Disk format: expanded
- Disk capacity: 65536 MB

You can change some of these parameters with `prlctl`. For example, to create an 84 GB IDE virtual disk, run

```
# prlctl set MyVM --device-add hdd --size 84000 --iface ide
Creating hdd2 () ide:0 image='/vz/vmprivate/d35d28e5-11f7-4b3f-9065-8fef6178bc5b/ \
harddisk2.hdd' type='expanded' 84000Mb
```


...

The virtual disk has been added to your virtual machine. However, before you can use it, you need to initialize it as described in the next subsection.

2.15.2.1.1 Adding Hyper-V SCSI Devices to Windows Virtual Machines

Virtuozzo Hybrid Server supports Hyper-V paravirtualized device emulation of SCSI hard disks and CD/DVD-ROM drives. The emulation allows to use these devices with native Windows drivers. It is recommended for all the supported Windows guests (see [Virtuozzo Hybrid Server Supported Guest Operating Systems Guide](#)).

To add, for example, a SCSI CD/DVD-ROM and HDD emulated via Hyper-V to a VM, run the following commands:

```
# prlctl set vm1 --device-add cdrom --image <path_to_image> --iface scsi --subtype hyperv
# prlctl set vm1 --device-add hdd --iface scsi --subtype hyperv
```

2.15.2.2 Initializing Newly Added Disks

After you added a new blank virtual hard disk to the virtual machine configuration, it will be invisible to the operating system installed inside the virtual machine until the moment you initialize it.

2.15.2.2.1 Initializing the New Virtual Hard Disk in Windows

To initialize a new virtual hard disk in a Windows guest OS, you will need the Disk Management utility available. For example, in Windows Server 2012 you can access this utility by clicking Start > Control Panel > System and Security > Administrative Tools > Computer Management > Storage > Disk Management.

When you open the Disk Management utility, it automatically detects that a new hard disk was added to the configuration and launches the Initialize Disk wizard:

1. In the Select disks section, select the newly added disk.
2. Choose the partition style for the selected disk: MBR (Master Boot Record) or GPD (GUID Partition Table).
3. Click OK.

The added disk will appear as a new disk in the Disk Management utility window, but its memory space will

be unallocated. To allocate the disk memory, right-click this disk name in the Disk Management utility window and select New Volume. The New Volume Wizard window will appear. Follow the steps of the wizard and create a new volume in the newly added disk.

After that your disk will become visible in My Computer and you will be able to use it as a data disk inside your virtual machine.

2.15.2.2.2 Initializing the New Virtual Hard Disk in Linux

Initializing a new virtual hard disk in a Linux guest OS comprises two steps: (1) allocating the virtual hard disk space and (2) mounting this disk in the guest OS.

To allocate the space, you need to create a new partition on this virtual hard disk using the fdisk utility:

Note: You need the root privileges to use fdisk.

1. Launch a terminal window.
2. To list the IDE disk devices present in your virtual machine configuration, enter:

```
fdisk /dev/hd*
```

Note: If you added a SCSI disk to the virtual machine configuration, use the `fdisk /dev/sd*` command instead.

3. By default, the second virtual hard disk appears as `/dev/hdc` in your Linux virtual machine. To work with this device, enter:

```
fdisk /dev/hdc
```

Note: If this is a SCSI disk, use the `fdisk /dev/sdc` command instead.

4. To get detailed information about the disk, enter:

```
p
```

5. To create a new partition, enter:

```
n
```

6. To create the primary partition, enter:

```
p
```

7. Specify the partition number. By default, it is 1.
8. Specify the first cylinder. If you want to create a single partition on this hard disk, use the default value.
9. Specify the last cylinder. If you want to create a single partition on this hard disk, use the default value.
10. To create a partition with the specified settings, enter:

```
w
```

When you allocated the space on the newly added virtual hard disk, you should format it by entering the following command in the terminal:

```
# mkfs -t <file system> /dev/hdc1
```

Note: <file system> stands for the file system you want to use on this disk. It is recommended to use ext4.

When the added virtual hard disk is formatted, you can mount it in the guest OS.

1. To create a mount point for the new virtual hard disk, enter:

```
# mkdir /mnt/hdc1
```

Note: You can specify a different mount point.

2. To mount the new virtual hard disk to the specified mount point, enter:

```
mount /dev/hdc1 /mnt/hdc1
```

When you mounted the virtual hard disk, you can use its space in your virtual machine.

2.15.2.3 Configuring Virtual Devices

You can configure the parameters of an existing virtual device in your virtual machine using the `--device-set` option of the `prlctl set` command.

To change the virtual device parameters, do the following:

1. List the VM information to find out the name of the device you wish to configure. For example, to obtain the list of virtual devices in the virtual machine MyVM, run:

```
# prlctl list -i MyVM
...
Hardware:
  cpu cpus=2 VT-x accl=high mode=32 ioprio=4 iolimit='0'
  memory 1024Mb
  video 32Mb 3d acceleration=off vertical sync=yes
  hdd0 (+) scsi:0 image='/vz/vmprivate/d35d28e5-11f7-4b3f-9065-8fef6178bc5b/ \
  harddisk.hdd' type='expanded' subtype=virtio-scsi
  hdd1 (+) scsi:1 image='/vz/vmprivate/d35d28e5-11f7-4b3f-9065-8fef6178bc5b/ \
  harddisk1.hdd' type='expanded' subtype=virtio-scsi
  cdrom0 (+) scsi:1 image='' subtype=virtio-scsi
  usb (+)
  net0 (+) dev='vme426f6594' network='Bridged' mac=001C426F6594 card=virtio
...
```

All virtual devices currently available to the virtual machine are listed under Hardware. In our case the virtual machine MyVM has the following devices: 2 CPUs, main memory, video memory, a floppy disk drive, 2 hard disk drives, a CD/DVD-ROM drive, a USB controller, and a network card.

2. Configure the properties of the virtual device. For example, to configure the current type of the virtual disk hdd1 in the virtual machine MyVM from SCSI to IDE, execute:

```
# prlctl set MyVM --device-set hdd1 --iface ide
```

To check that the virtual disk type has been successfully changed, use the `prlctl list -i` command:

```
# prlctl list -i MyVM | grep hdd1
hdd1 (+) ide:0 image='/vz/vmprivate/d35d28e5-11f7-4b3f-9065-8fef6178bc5b/ \
harddisk1.hdd' type='expanded'
```

2.15.2.3.1 Connecting and Disconnecting Virtual Devices

In Virtuozzo Hybrid Server, you can connect or disconnect certain devices when a virtual machine is running. These devices include:

- CD/DVD-ROM drives
- Floppy disk drives
- Network adapters
- Printer ports
- Serial ports

Usually, all virtual devices are automatically connected to a virtual machine when you create them. To disconnect a device from the virtual machine, you can use the `prlctl set` command. For example, the following command disconnects the CD/DVD-ROM drive `cdrom0` from the virtual machine `MyVM`:

```
# prlctl set MyVM --device-disconnect cdrom0
```

To connect the CD/DVD-ROM drive back, you can run the following command:

```
# prlctl set MyVM --device-connect cdrom0
```

2.15.2.4 Deleting Devices

You can delete a virtual device that you do not need any more in your virtual machine using the `--device-del` option of the `prlctl set` command. The options responsible for removing particular devices are listed in the following table:

Option	Description
<code>hdd</code>	Deletes the specified hard disk drive from a stopped virtual machine.
<code>cdrom</code>	Deletes the specified CD/DVD-ROM drive from a running or stopped virtual machine.
<code>net</code>	Deletes the specified network adapter from a stopped virtual machine.
<code>fdd</code>	Deletes the floppy disk drive from a running or stopped virtual machine.
<code>serial</code>	Deletes the specified serial port from a stopped virtual machine.
<code>usb</code>	Deletes the USB controller from a running or stopped virtual machine.

To remove a virtual device, do the following:

1. List the VM information to find out the name of the device you wish to delete. For example, to obtain the list of virtual devices in the virtual machine `MyVM`, run:

```
# prlctl list --info MyVM
...
Hardware:
  cpu cpus=2 VT-x accl=high mode=32 ioprio=4 iolimit='0'
  memory 1024Mb
  video 32Mb 3d acceleration=off vertical sync=yes
  hdd0 (+) scsi:0 image='/vz/vmprivate/d35d28e5-11f7-4b3f-9065-8fef6178bc5b/ \
  harddisk.hdd' type='expanded' subtype=virtio-scsi
  hdd1 (+) scsi:1 image='/vz/vmprivate/d35d28e5-11f7-4b3f-9065-8fef6178bc5b/ \
  harddisk1.hdd' type='expanded' subtype=virtio-scsi
  cdrom0 (+) scsi:1 image='' subtype=virtio-scsi
  usb (+)
```

```
net0 (+) dev='vme426f6594' network='Bridged' mac=001C426F6594 card=virtio
...
```

2. Remove the virtual device from your virtual machine. For example, to remove the virtual disk hdd1 from the virtual machine MyVM, execute:

```
# prlctl set MyVM --device-del hdd1
```

If you do not want to permanently delete a virtual device, you can temporarily disconnect it from the virtual machine using the `--disable` option.

2.15.3 Making Screenshots

When a virtual machine stops responding to requests, you can check its state by capturing an image (or screenshot) of its screen with the `prlctl capture` command. A screenshot is saved in PNG format.

Note: You can take screenshots of running virtual machines only.

To take a screenshot of the MyVM virtual machine screen and save it to the `/usr/screenshots/image1.png` file:

1. Make sure that the virtual machine is running:

```
# prlctl list
UUID                                STATUS  IP_ADDR      T  NAME
{b2de86d9-6539-4ccc-9120-928b33ed31b9}  running  10.10.100.1  VM  MyVM
```

2. Take the virtual machine screenshot:

```
# prlctl capture MyVM --file /usr/screenshots/image1.png
```

If the `--file` option is not specified, a screenshot is dumped to the command output.

3. Check that the `image1.png` file has been successfully created:

```
# ls /usr/screenshots/
image1.png
```

2.15.4 Assigning Host Devices to Virtual Machines

Starting from Virtuozzo Hybrid Server 7 Update 11, you can assign a host device to a virtual machine so that the device is automatically connected to the VM when you connect the device to the hardware node or start the VM. This method, also known as passthrough, allows host devices to appear and behave as if they were physically attached to the guest operating system. Virtuozzo Hybrid Server supports assigning USB, PCI devices (physical NICs), and SR-IOV devices (virtual NICs). Ways to do this are described in the following sections.

When assigning host devices to VMs, keep in mind that:

- After a VM is migrated to another server, all its assignments are lost.
- All assignments are preserved if you restore a VM to its original location. Otherwise they are lost.

2.15.4.1 Assigning USB Devices to Virtual Machines

Warning: This feature is experimental and tested only with some USB devices.

To assign a USB device to a VM, specify two parameters:

1. The USB device ID that you can get by running the `prlsrvctl usb list` command. For example:

```
# prlsrvctl usb list
Sony - Storage Media      '3-2|054c|05ba|high|--|CB070B7B49A7C06786|3'
```

2. The virtual machine UUID that you can obtain by running the `prlctl list -a` command. For example:

```
# prlctl list -a
UUID                                STATUS      IP_ADDR      T  NAME
{d8d516c9-dba3-dc4b-9941-d6fad3767035}  stopped    10.10.100.1  VM MyVM
```

Once you know the necessary parameters, do the following:

1. Make sure the VM is stopped. For example:

```
# prlctl list -i MyVM | grep State
State: stopped
```

2. Assign the USB device to the VM using the `prlsrvctl usb set` command. For example:

```
# prlsvctl usb set '3-2|054c|05ba|high|--|CB070B7B49A7C06786|3' \
{d8d516c9-dba3-dc4b-9941-d6fad3767035}
```

This command assigns the USB device Sony - Storage Media with ID

'3-2|054c|05ba|high|--|CB070B7B49A7C06786|3' to the virtual machine MyVM with UUID

{d8d516c9-dba3-dc4b-9941-d6fad3767035}. When running the command, remember to put the USB device ID in single quotes and the VM UUID in curly brackets.

To check that the USB device has been successfully assigned to the VM, use the `prlsvctl usb list` command. For example:

```
# prlsvctl usb list
Sony - Storage Media      '3-2|054c|05ba|high|--|CB070B7B49A7C06786|3' \
{d8d516c9-dba3-dc4b-9941-d6fad3767035}
```

The command output shows that the USB device Sony - Storage Media will be automatically connected to the VM with UUID {d8d516c9-dba3-dc4b-9941-d6fad3767035} every time you start this VM and connect the device to the hardware node.

To remove the assignment, use the `prlsvctl usb del` command. For example:

```
# prlsvctl usb del '3-2|054c|05ba|high|--|CB070B7B49A7C06786|3'
```

2.15.4.2 Assigning PCI Devices to Virtual Machines

Assigning PCI devices to VMs is available only on server platforms that support Intel Virtualization Technology for Directed I/O (VT-d) or AMD I/O Virtualization Technology (IOMMU). This feature must be supported by both the motherboard chipset and CPU. For a list of IOMMU-supporting hardware, see [this Wikipedia article](#).

Before assigning PCI devices, prepare your system as follows:

1. Enable the CPU virtualization extensions (VT-d or AMD IOMMU) in the host BIOS.
2. Enable the IOMMU support in the kernel:
 - 2.1. In the `/etc/default/grub` file, locate the `GRUB_CMDLINE_LINUX` line and add within the quotes either `intel_iommu=on iommu=pt` or `amd_iommu=pt`, depending on the platform. For example, on an Intel-based system, the line will be:

```
GRUB_CMDLINE_LINUX="crashkernel=auto rd.lvm.lv=virtuozzo/root quiet \
intel_iommu=on iommu=pt"
```

- 2.2. Regenerate the grub config file:

- On a BIOS-based system, run:

```
# grub2-mkconfig -o /etc/grub2.cfg
```

- On a UEFI-based system, run:

```
# grub2-mkconfig -o /etc/grub2-efi.cfg
```

3. Reboot the system to apply the changes.

To check that IOMMU is enabled, run one of the following commands:

- On an Intel-based system:

```
# dmesg | grep "Virtualization Technology for Directed I/O"
[ 0.902214] DMAR: Intel(R) Virtualization Technology for Directed I/O
```

- On an AMD-based system:

```
# dmesg | grep AMD-Vi
AMD-Vi: Enabling IOMMU at 0000:00:00.2 cap 0x40
AMD-Vi: Lazy IO/TLB flushing enabled
AMD-Vi: Initialized for Passthrough Mode
```

Now you can proceed to assign the PCI device to a VM. Do the following:

1. Find the ID of the desired device in the output of the `prlsrvctl info | grep pci` command. For example:

```
# prlsrvctl info | grep pci
pci X540 Ethernet Controller '8:0:1:8086.1521' assignment=host
<...>
```

2. Make sure the VM that you will assign the PCI device to is stopped. For example:

```
# prlctl list -i MyVM | grep State
State: stopped
```

3. Assign the PCI device to the VM. For example, to assign the network adapter X540 Ethernet Controller with the ID '8:0:1:8086.1521' to the virtual machine MyVM, run:

```
# prlctl set MyVM --device-add pci --device '8:0:1:8086.1521'
```

To check that the PCI device has been successfully assigned to the VM, run the following command:

```
# prlctl list -i MyVM | grep pci
pci0 (+) 'Intel Corporation Ethernet Controller 10-Gigabit X540-AT2' '8:0:1'
```

Now the PCI device X540 Ethernet Controller will automatically connect to the MyVM virtual machine every time this VM is started and the device is connected to the hardware node.

To remove the assignment, use the `prlctl set --device-del` command:

```
# prlctl set --device-del pci0
```

2.15.4.3 Using PCI I/O Virtualization

Single Root I/O Virtualization (SR-IOV) is a specification that allows a single physical PCI device to appear as multiple virtual devices and thus be shared by multiple virtual machines. Virtuozzo Hybrid Server supports assigning Virtual Functions of SR-IOV-capable PCI devices, namely network adapters, to virtual machines.

Before creating virtual PCI devices, make sure the following requirements are met:

1. The system is prepared to support IOMMU as described in *Assigning PCI Devices to Virtual Machines* on page 82.
2. SR-IOV is enabled in the system BIOS.

To create virtual network adapters, or Virtual Functions of a network adapter, you need to specify their number in the `/sys/class/net/<device_name>/device/sriov_numvfs` file. For example, to create two Virtual Functions of the Ethernet adapter `enp2s0`, run:

```
# echo 2 > /sys/class/net/enp2s0/device/sriov_numvfs
```

To ensure that the intended number of Virtual Functions is persistent across server reboots, create a udev rule similar to the following:

```
# vi /etc/udev/rules.d/enp2s0.rules
ACTION=="add", SUBSYSTEM=="net", ENV{ID_NET_DRIVER}=="ixgbe",
ATTR{device/sriov_numvfs}="2"
```

where `enp2s0` is the name of the network adapter, `ixgbe` is the name of the network driver in use, and `2` is the number of Virtual Functions to be created at boot time.

The maximum number of Virtual Functions supported by a network device can be queried by reading the `/sys/class/net/<device_name>/device/sriov_totalvfs` file. For example, for the Ethernet adapter `enp2s0`, run:

```
# cat /sys/class/net/enp2s0/device/sriov_totalvfs
63
```

To check that Virtual Functions have been successfully created, execute:

```
# prlsrvctl info | grep Ethernet
pci X540 Ethernet Controller '8:0:1:8086.1521' assignment=host
pci X540 Ethernet Controller Virtual Function '9:10:4:8086.1520' assignment=host
pci X540 Ethernet Controller Virtual Function '9:10:0:8086.1520' assignment=host
```

After creating virtual network adapters, you can assign them to VMs as described in *Assigning PCI Devices to*

Virtual Machines on page 82. For them to work properly, install appropriate network drivers inside the VMs.

2.15.5 Configuring IP Address Ranges for Host-Only Networks

All virtual machines connected to networks of the host-only type receive their IP addresses from the DHCP server. This DHCP server is set up during the Virtuozzo Hybrid Server installation and includes by default IP addresses from 10.37.130.1 to 10.37.130.254. You can redefine the default IP address range for host-only networks and make virtual machines get their IP addresses from different IP address ranges. For example, you can run the following command to set the start and end IP addresses for the `Host-Only` network (this network is automatically created during the Virtuozzo Hybrid Server installation) to 10.10.11.1 and 10.10.11.254, respectively:

```
# prlsrvctl net set Host-Only --ip-scope-start 10.10.11.1 --ip-scope-end 10.10.11.254
```

You can also specify a custom IP address range directly when creating a new network of the host-only type. Assuming that you want to create a network with the `Host-Only2` name and define for this network the IP addresses range from 10.10.10.1 to 10.10.10.254, you can execute the following command:

```
# prlsrvctl net add Host-Only2 -t host-only --ip-scope-start 10.10.10.1 --ip-scope-end 10.10.10.254
```

When working with IP address ranges, pay attention to the following:

- The start and end IP addresses of an IP address range must belong to the same subnetwork.
- IP address ranges can be defined for each network of the host-only type separately. For example, you can set the IP address range from 10.10.11.1 to 10.10.11.254 for the `Host-Only` network and from 10.10.10.1 to 10.10.10.254 for the `Host-Only2` network.

2.15.6 Configuring Virtual Machine Crash Mode

In Virtuozzo Hybrid Server 7, you can configure a virtual machine behavior after the guest OS crash: restart or pause. By default, when a virtual machine fails, a crash dump is created and sent in the problem report to the Virtuozzo technical support team, and the virtual machine is restarted with the same configuration.

To address the problem yourself, you can switch the virtual machine crash mode to pause using the `prlctl set` command. For example:

```
# prlctl set MyVM --on-crash pause
```

The virtual machine resources will be preserved to allow analysis and its crash dump will be sent in the problem report.

As crash dumps can take up significant disk space and lead to the whole server malfunction, the crash mode is automatically switched to pause, if a virtual machine fails more than three times within twenty-four hours since the last crash.

If you want to skip creating the crash dump and sending the problem report, add `:no-report` to the command. For example:

```
# prlctl set MyVM --on-crash restart:no-report
```

2.15.7 Enabling Secure Boot for Virtual Machines

Secure Boot is a UEFI feature that provides security in the pre-boot environment. It works by ensuring that only trusted software components signed by the Original Equipment Manufacturer (OEM) are loaded during the boot process. Secure boot can only be used if the system boots using UEFI, instead of BIOS.

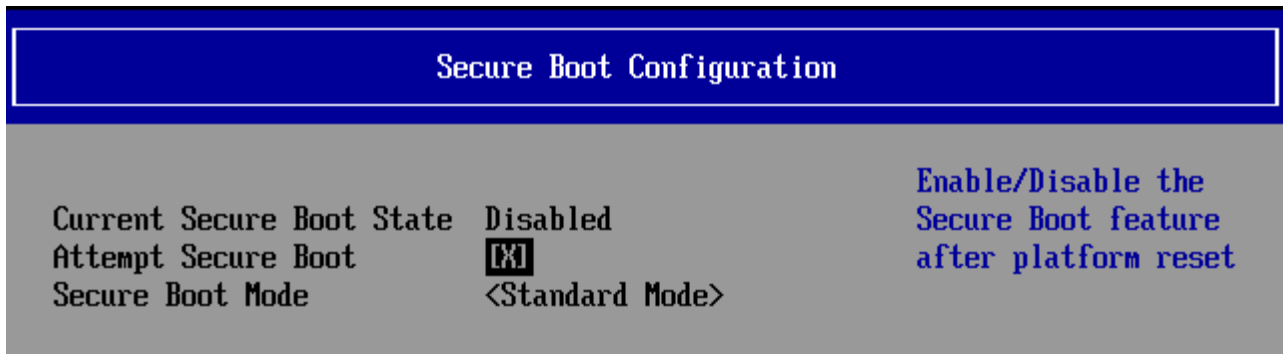
Secure Boot is supported in Virtuozzo Hybrid Server 7 virtual machines running the following operating systems: Windows Server 2012 and newer, CentOS 7, Ubuntu 14.04 LTS and newer.

If you want to enable Secure Boot in your virtual machine, make sure it boots with the EFI firmware before installing a guest OS inside it. To configure the EFI boot, use the `prlctl set --efi-boot` command. For example:

```
# prlctl set MyVM --efi-boot on
```

To enable Secure Boot inside a guest OS, do the following:

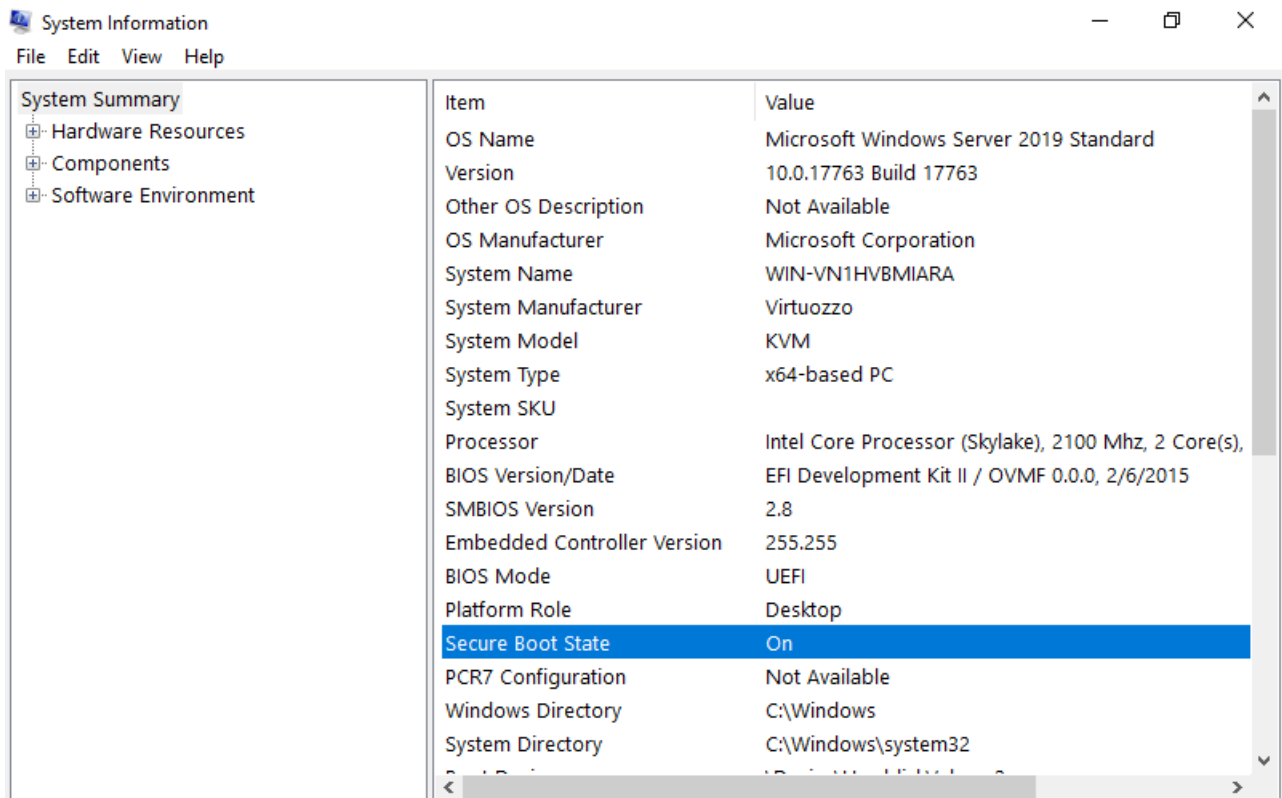
1. While the VM is booting up, press **F2** in the VNC console to open the EFI setup.
2. On the EFI setup screen, select **Device Manager > Secure Boot Configuration**.
3. Locate the **Attempt Secure Boot** option and press **Spacebar** to toggle the checkbox.



4. Press **F10** to save the changes and reset the platform.

You can check that Secure Boot is successfully enabled as follows:

- In Windows VMs, check the **Secure Boot State** field in the **System Information** window:



- In Linux VMs, use `mokutil`:

```
# mokutil --sb-state
SecureBoot enabled
```

2.16 Converting Third-Party Virtual Machines

Virtual machines created for third-party hypervisors can be converted into Virtuozzo Hybrid Server VMs using the `virt-v2v` tool.

In particular, see the [Migrate a VMware virtual machine from VMware vCenter to Virtuozzo 7](#) article for instructions on how to convert Linux-based VMware VMs to Virtuozzo Hybrid Server ones.

As Virtuozzo guest tools must be installed for the smooth functioning of virtual machines, ensure the manual network configuration retains the settings that existed before the conversion of VMware virtual machines to Virtuozzo Hybrid Server ones.

2.17 Converting Containers with `almaconvert8`

Virtuozzo Hybrid Server introduces the option of converting the CentOS 7, CentOS 8, VzLinux 7, and VzLinux 8 operating systems running on a container to AlmaLinux 8.

To use this option, download and install the `vzdeploy8` package:

```
yum install https://repo.virtuozzo.com/vzlinux/vzdeploy/vzdeploy8.rpm
```

Use the `almaconvert8 list` command to list all the containers available for converting. For example:

```
# almaconvert8 list
...
MyCT-centos8
MyCT-centos7
MyCT-vzlinux7
MyCT-4-vzlinux8
```

Execute the `almaconvert8 list-cpanel` command to list containers with cPanel installed inside, available for converting to AlmaLinux 8.

Run the `almaconvert8 convert` command to convert a container from one operating system to AlmaLinux 8. For example, to convert the `MyCT` container to AlmaLinux 8, use the following:

```
# almaconvert8 convert MyCT
```

If you need to convert several containers simultaneously, run the following:

```
# almaconvert8 convert --parallel 7
```

7 is the number of containers that undergo the conversion concurrently.

Find below the list of options you can use when running the `almaconvert8 convert` command.

```
almaconvert8 convert [-h] [--dry-run] [-q] [-v] [--log LOG]
                    [--parallel [parallel]] [--elevate] [--strict]
                    CT [CT1 CT2 ... CTn]
```

Name	Description
CT	The UUID, CTID, or name of the container to convert. If there are several containers to convert, delimit them by spaces.
-h, --help	Provide the list of flags used with <code>almaconvert8 convert</code> .
--dry-run	Check that the conversion is possible.
-q, --quiet	Suppress the output.
-v, --verbose	Provide detailed processing information.
--log LOG	Dump all the information and messages to the specified log file.
--parallel [parallel]	The number of simultaneous conversions to execute.
--elevate	Call the elevate upgrade scripts and wait for them to complete. To use this option for <code>vzdeploy8-1.0.63-1</code> or later, ensure you are running at least Virtuozzo Hybrid Server 7.5 Update 6 or later versions.
--elevate_continue	Call the elevate script with the <code>--continue</code> parameter. It does not have an effect without the <code>--elevate</code> key.
--strict	Block the conversion when a precheck message appears.

Note: The `--elevate` option is an experimental feature for third-party scripts and is optional. Use it only when you have a complete understanding of the way this option is used. If you need to check the version of the tool, run the following command:

```
# almaconvert8 version
```

2.18 Converting Containers to Virtual Machines with the c2v-convert Tool

The `c2v-convert` tool converts a Virtuozzo Hybrid Server 7 system container to a virtual machine. This tool is tested to work with containers based on the following operating systems:

- AlmaLinux 9
- AlmaLinux 8
- CentOS 9
- CentOS 8
- CentOS 7
- Debian 12
- Debian 11
- Debian 10
- RockyLinux 8
- Ubuntu 24.04
- Ubuntu 22.04
- Ubuntu 20.04

Note: An Internet connection is required for `c2v-convert` to access guest OS vendor repositories.

To install `c2v-convert`, use the following command:

```
# yum install c2v-convert
```

Find below a list of options you can use when running the `c2v-convert` command.

```
c2v-convert [-h] [--version] [-q] [-y] [-v] [-b] [--timeout TIMEOUT]
            [--swap SWAP] [--target-state {keep,start,stop}] [--log LOG]
            [--temp TEMP] [-f FROM_FILE] [ct [ct ...]]
```


Name	Description
ct	The name of the container to be converted to a virtual machine.
-h, --help	Provides a list of flags used with the <code>c2v-convert</code> command.
--version	Specifies the program's version number.
-q, --quiet	Skips the confirmation prompt and suppresses the output. Useful when calling from a script.
-y, --yes	The system automatically selects Y for all requests. The <code>-y</code> option assumes <code>Ask me later</code> for data collection by CEP.
-v, --verbose	The verbose mode provides detailed processing information.
-b, --batch_log	Creates a batch log file that provides log records for the containers subject to conversion.
--timeout TIMEOUT	The operation timeout in minutes [5..60]. By default, it is 30 minutes.
--swap SWAP	The swap size in the target virtual machine in MB. The default value is 2,048 MB.
--target-state {keep, start, stop}	<p>This option specifies the desired target state of virtual machine(s) after conversion:</p> <ol style="list-style-type: none"> 1. <code>keep</code>—The resulting virtual machine(s) will keep the state of the converted container. If the container started before the conversion, the resulting virtual machine will be started. If the container stopped before the conversion, its state will remain <code>stop</code>. 2. <code>stop</code>—The resulting virtual machine(s) will be stopped, regardless of the container's state prior to the conversion. 3. <code>start</code>—The resulting virtual machine(s) will be started, regardless of the container's state before the conversion. <p>By default, the value of this option is <code>keep</code>.</p>
--log LOG	The path to the directory to store log files.
--temp TEMP	The path to the directory to store temporary files. By default, it is <code>/vz/tmp</code> .
-f FROM_FILE, --from-file FROM_FILE	The path to the batch file.

2.18.1 Operation

The container-to-virtual-machine conversion consists of 11 stages:

1. Checking the configuration of the container to be converted.
2. Checking disk consistency and snapshot presence. (If a container has snapshots, it may take longer to complete the conversion.)
3. Installing `kernel` and `grub`. (The performance of this stage depends on the Internet speed.)
4. Creating a target virtual machine in a temporary directory.
5. Converting the copies of the container's disks to a virtual machine format.
6. Connecting the disks to the target virtual machine.
7. Configuring the target virtual machine's network. (The network in the container is disconnected to avoid potential conflicts.)
8. Installing Guest Tools on the target virtual machine.
9. Executing the `virt-v2v` scripts in the target virtual machine.
10. Starting the target virtual machine.
11. The cleanup stage.

2.18.1.1 Conversion Time

The conversion time will depend on the node CPU and storage performance. Typically, a single container conversion completes within less than 5 minutes.

2.18.1.2 Conversion Fallback

The resulting virtual machine acquires the name of the original container.

The source system container's name will include the `c2v_` prefix to identify the converted containers easily. The source system container is left in the stopped state without IP configuration as a fallback.

2.18.1.3 Interactive and Non-interactive Modes

You can convert containers in interactive and non-interactive modes:

- Interactive mode encompasses default settings and is used without any flags. Use this mode to test the conversion process or if you have a few containers to convert.
- Non-interactive mode is used when you have many containers to convert and want the system to do it automatically. For example, you can specify options, such as `-y` and `-v`, etc., to instruct that the system automatically selects `y` for all requests and provides detailed processing information.

2.18.2 Example 1: Converting a Single Container in Interactive Mode

Converting a single container in the interactive mode is a good way to test the conversion process for the container types you have in your infrastructure. For example, you may have 40 containers based on Debian 9 and 120 containers based on CentOS 7, with half of CentOS 7 containers having MariaDB installed. You can set up new Debian 9, CentOS 7, and CentOS 7 containers with MariaDB containers and test the conversion. Alternatively, you can create three clones of each type and test the conversion using these clones.

To convert a single container with the name `centos7` to a virtual machine in the interactive mode, run `c2v-convert`:

```
# c2v-convert centos7
```

Here is an example output:

```
[root@vhs75]# c2v-convert centos7
The conversion process will restart the container multiple times and may take up to 30 minutes to c
Proceed? (y/n)
y
2024-10-18 17:17:27,863 INFO: Starting conversion for CT centos7.
2024-10-18 17:17:27,863 INFO Stage [1/11]: Preconversion checks for container
2024-10-18 17:17:36,259 INFO Stage [2/11]: Checking disks consistency
2024-10-18 17:17:53,707 INFO Stage [3/11]: Installing kernel and grub, preparing CT for conversion
2024-10-18 17:18:45,773 INFO Stage [4/11]: Creating the VM for migration
2024-10-18 17:18:47,297 INFO Stage [5/11]: Converting disks in container
2024-10-18 17:19:03,679 INFO Stage [6/11]: Connecting disks to VM
2024-10-18 17:19:03,679 INFO Stage [6/11]: No extra disks to add to fstab
2024-10-18 17:19:08,602 INFO Stage [7/11]: Configuring network
2024-10-18 17:19:08,855 INFO Stage [8/11]: Installing Guest Tools and Agent
2024-10-18 17:19:09,097 INFO Stage [9/11]: Run v2v migration
2024-10-18 17:19:47,525 INFO Stage [10/11]: Start VM after v2v migration
2024-10-18 17:20:17,973 INFO Stage [11/11]: Cleanup and correct VENV names
2024-10-18 17:20:18,444 INFO: Conversion complete.
```

```
The source container is stopped and available as fallback.
If the resulting VM experiences boot issues, use prlctl start C2V_centos7.
```

```
The resulting VM UUID: 1731524066
The resulting VM name: centos7
[root@vhs75]#
```

The new virtual machine acquires the name of the original container, and the system assigns the C2V_ prefix to the container's name. After the conversion is complete, run `prlctl list -a` to confirm that the new virtual machine centos7 is present in the system:

Run `prlctl list -a` to view a list of available virtual environments:

```
[root@vhs75]# prlctl list -a | grep centos7
{a1fa15c6-8d74-4335-befc-1dfab51798b1}  stopped      -          CT C2V_centos7
{e734f5e2-f136-4cdd-a644-e58430956c6f}  stopped      -          VM centos7
[root@vhs75]#
```

Execute `prlctl list -i` to receive information on the new virtual machine centos7:

```
# prlctl list -i centos7
```

Use the `prlctl start` command to start the new virtual machine:

```
# prlctl start centos7
```

2.18.3 Example 2: Converting Multiple Containers in Interactive Batch

Mode

You can specify multiple containers when calling `c2v-convert`. In this case, the tool will run in batch mode and convert the containers sequentially. Use the interactive batch mode if you want to oversee the conversion process of multiple containers.

To convert two containers `centos7_1` and `centos7_2` to virtual machines in the interactive batch mode, run the following:

```
# c2v-convert centos7_1 centos7_2
```

In this mode, you will be prompted to confirm the conversion of both containers. The tool will convert them one after another in the order you specified them.

In the following example, we will convert two system containers to virtual machines and will not request a batch log file. Here is an example output of `c2v-convert` for two containers:

```

[root@vhs75]# c2v-convert centos7_1 centos7_2
2024-10-18 17:29:30,248 INFO: Starting batch conversion of: centos7_1, centos7_2
2024-10-18 17:29:30,249 INFO: Validating the list of CTs
2024-10-18 17:29:30,441 INFO: Container list successfully validated
The conversion process will restart the containers multiple times and may take up to 60 minutes to
Proceed? (y/n)
y
2024-10-18 17:29:32,547 INFO Container [1/2]: Starting conversion for CT centos7_1.
2024-10-18 17:29:32,547 INFO Container [1/2] Stage [1/11]: Preconversion checks for container
2024-10-18 17:29:41,311 INFO Container [1/2] Stage [2/11]: Checking disks consistency
2024-10-18 17:29:58,587 INFO Container [1/2] Stage [3/11]: Installing kernel and grub, preparing CT
2024-10-18 17:30:24,825 INFO Container [1/2] Stage [4/11]: Creating the VM for migration
2024-10-18 17:30:26,177 INFO Container [1/2] Stage [5/11]: Converting disks in container
2024-10-18 17:30:41,100 INFO Container [1/2] Stage [6/11]: Connecting disks to VM
2024-10-18 17:30:41,100 INFO Container [1/2] Stage [6/11]: No extra disks to add to fstab
2024-10-18 17:30:43,671 INFO Container [1/2] Stage [7/11]: Configuring network
2024-10-18 17:30:43,925 INFO Container [1/2] Stage [8/11]: Installing Guest Tools and Agent
2024-10-18 17:30:44,174 INFO Container [1/2] Stage [9/11]: Run v2v migration
2024-10-18 17:31:21,901 INFO Container [1/2] Stage [10/11]: Start VM after v2v migration
2024-10-18 17:31:52,343 INFO Container [1/2] Stage [11/11]: Cleanup and correct VENV names
2024-10-18 17:31:52,821 INFO Container [1/2]: Conversion complete.

The source container is stopped and available as fallback.
If the resulting VM experiences boot issues, use prlctl start C2V_centos7_1.

The resulting VM UUID: 1744630373
The resulting VM name: centos7_1
2024-10-18 17:31:52,850 INFO Container [2/2]: Starting conversion for CT centos7_2.
2024-10-18 17:31:52,850 INFO Container [2/2] Stage [1/11]: Preconversion checks for container
2024-10-18 17:32:00,045 INFO Container [2/2] Stage [2/11]: Checking disks consistency
2024-10-18 17:32:17,236 INFO Container [2/2] Stage [3/11]: Installing kernel and grub, preparing CT
2024-10-18 17:33:06,333 INFO Container [2/2] Stage [4/11]: Creating the VM for migration
2024-10-18 17:33:07,662 INFO Container [2/2] Stage [5/11]: Converting disks in container
2024-10-18 17:33:21,350 INFO Container [2/2] Stage [6/11]: Connecting disks to VM
2024-10-18 17:33:21,351 INFO Container [2/2] Stage [6/11]: No extra disks to add to fstab
2024-10-18 17:33:25,016 INFO Container [2/2] Stage [7/11]: Configuring network
2024-10-18 17:33:25,277 INFO Container [2/2] Stage [8/11]: Installing Guest Tools and Agent
2024-10-18 17:33:25,506 INFO Container [2/2] Stage [9/11]: Run v2v migration
2024-10-18 17:34:03,603 INFO Container [2/2] Stage [10/11]: Start VM after v2v migration
2024-10-18 17:34:34,054 INFO Container [2/2] Stage [11/11]: Cleanup and correct VENV names
2024-10-18 17:34:34,527 INFO Container [2/2]: Conversion complete.

The source container is stopped and available as fallback.
If the resulting VM experiences boot issues, use prlctl start C2V_centos7_2.

The resulting VM UUID: 1920612944
The resulting VM name: centos7_2
2024-10-18 17:34:34,556 INFO: Containers successfully converted: centos7_1 centos7_2
[root@vhs75]#

```

Now, the output shows not only the conversion stage but also the currently processed and total number of containers to convert: Container [1/2] and Container [2/2]. It allows you to see how many containers have

already been processed at a glance.

2.18.3.1 Logging in Batch Mode

If you specified more than one container to convert, `c2v-convert` will create a separate log file for each container in the working directory you started the conversion process. You may add the `-b` option to create a separate batch log file that will contain brief log information on each container and/or `--log` to specify the name of the folder to store log files with detailed information on each container:

```
# c2v-convert -b --log /tmp centos7_1 centos7_2
```

2.18.3.2 Conversion in Batch Mode Using GNU Screen

When converting containers in batch mode, we recommend using the GNU screen terminal program. It creates a virtual terminal persistent across SSH sessions and ensures a strong connection to the target host. With Screen, the program will continue working in the screen session in case of any SSH session disruption.

To start a screen session, run the following in your terminal:

```
$ screen
```

To detach from the screen session, execute `Ctrl-a+d`.

To reattach or resume your screen session, run as follows:

```
$ screen -r
```

2.18.4 Example 3: Converting Multiple Containers in Non-Interactive Batch Mode

The non-interactive batch mode is useful when converting many containers in a pre-defined maintenance window. Use the `-y` option to instruct the system to select `y` for all requests automatically. As with the interactive batch mode, we recommend starting `c2v-convert` in the non-interactive mode in the GNU screen.

2.18.4.1 Batch File

To simplify calling `c2v-convert` on large numbers of containers, we recommend preparing a batch file with container names or CTIDs that you want to convert during a maintenance window. The format of the batch file is as follows:

```
centos7-ct1
centos7-ct2
centos7-ct3
centos7-ct4
centos7-ct5
ubuntu20-1
ubuntu20-2
ubuntu20-3
ubuntu20-4
ubuntu20-5
```

It is possible to generate such a file based on the `prlctl list` output. For example, to create a batch file with the names of all containers apart from containers with Virtuozzo Automator, Virtuozzo Storage UI, and PowerPanel, use the following command:

```
[root@vhs75]# prlctl list --vmtype ct -aHo name | egrep -v "va-mn|pp|vstorage-ui" > cts-to-convert.txt
[root@vhs75]# cat cts-to-convert.txt
100500
1113
200701
3000
3011
ubuntu20_ct
nfs-server
prometheus
1237612346
[root@vhs75]#
```

You can use different batch files for containers based on a specific template, end-user, or any other criteria you select. For example, if you confirmed a maintenance window with an end user owning 20 containers.

2.18.4.2 Example: Converting Containers in Unattended Mode Using Batch File

In this example, we want to convert all containers containing `centos7` and `dev` in their names. We aligned the maintenance window with the users of said containers, and there are enough of them to justify running conversion in the quiet and unattended mode. We will create a batch file based on the `prlctl list` output and our filter and a batch log file to review once the conversion process ends.

We will be running the conversion in a screen session.

2.18.4.2.1 Create Batch File

To create a batch log file, we will use `prlctl list` and `grep`:

```
[root@vhs75]# prlctl list -aHo name | grep dev.*centos7 > cts-to-convert.txt
[root@vhs75]# cat cts-to-convert.txt
dev-112-centos7-spam
dev-11412-centos7-mail
dev-13412-centos7-proxy
dev-6759-centos7-proxy
dev-76224-centos7-mail
dev-8571-centos7-mail
[root@vhs75]#
```

2.18.4.2.2 Execute Conversion

Let's open a screen session:

```
[root@vhs75]# screen
```

Next, we will call `c2v-convert` by specifying the following:

- The `-q` flag to run the conversion in quiet mode, i.e., without confirmation prompts and output.
- The `-b` flag to create a high-level batch log.
- `-f` to point the tool to the file with container names.

Here is the command:

```
[root@vhs75]# c2v-convert -q -b -f ./cts-to-convert.txt
...
```

Because we are using quiet mode, nothing will be printed on the screen. Let's detach from the screen session (`Ctrl-a+d`) and look at the batch log file:

```
[root@vhs75]# screen
[detached from 81468.pts-1.vhs75]
[root@vhs75]# ll
total 48
-rw-r--r-- 1 root root 1314 Oct 18 18:06 20241018-18-05-batch-conversion.log
-rw-r--r-- 1 root root 33818 Oct 18 18:06 20241018-18-05-dev-112-centos7-spam-conversion.log
-rw-r--r-- 1 root root 136 Oct 18 17:57 cts-to-convert.txt
[root@vhs75]# tail -f 20241018-18-05-batch-conversion.log
2024-10-18 18:07:56,297 INFO Container [1/6]: Conversion complete.
```

The source container is stopped and available as fallback.
If the resulting VM experiences boot issues, use `prlctl start C2V_dev-112-centos7-spam`.


```

The resulting VM UUID: 1839266756
The resulting VM name: dev-112-centos7-spam
2024-10-18 18:07:56,327 INFO Container [2/6]: Starting conversion for CT dev-11412-centos7-mail.
2024-10-18 18:07:56,327 INFO Container [2/6] Stage [1/11]: Preconversion checks for container
2024-10-18 18:08:03,546 INFO Container [2/6] Stage [2/11]: Checking disks consistency

```

As we can see, the conversion process is underway. Running the conversion in quiet mode allows you to call it from external automation systems or scripts.

2.18.5 Troubleshooting

2.18.5.1 Running the Tool in Verbose Mode

In case the conversion process fails without a clear error message, rerun the conversion with the `--verbose` option:

```
# c2v-convert -y --verbose myct
```

A verbose output provides much more information about the steps performed, which is useful for troubleshooting. Here is an example of `c2v-convert` running in the verbose mode:

```

[root@vhs75 ~]# c2v-convert -y --verbose myct
2024-11-01 12:42:38,432 DEBUG: Starting conversion with Namespace(batch_log=False, ct=['myct'], fro
2024-11-01 12:42:38,433 DEBUG: Command "modprobe nbd max_part=8" started.
2024-11-01 12:42:38,435 DEBUG: Command "modprobe nbd max_part=8" output:

2024-11-01 12:42:38,435 DEBUG: Command "modprobe nbd max_part=8" finished.
2024-11-01 12:42:38,435 DEBUG: Command "mkdir -p /vz/tmp/c2v" started.
2024-11-01 12:42:38,437 DEBUG: Command "mkdir -p /vz/tmp/c2v" output:

2024-11-01 12:42:38,437 DEBUG: Command "mkdir -p /vz/tmp/c2v" finished.
2024-11-01 12:42:38,437 DEBUG: Command "prlctl list myct" started.
2024-11-01 12:42:38,469 DEBUG: Command "prlctl list myct" output:
UUID                                STATUS      IP_ADDR      T  NAME
{653ae3d9-c475-4922-b7dc-99ee75bc2db9}  running    14.11.182.2  CT  myct

2024-11-01 12:42:38,469 DEBUG: Command "prlctl list myct" finished.
2024-11-01 12:42:38,534 DEBUG: Checking CT myct status
2024-11-01 12:42:38,566 INFO: Starting conversion for CT myct.
2024-11-01 12:42:38,567 INFO Stage [1/11]: Preconversion checks for container
2024-11-01 12:42:38,567 DEBUG Stage [1/11]: Checking CT myct status
2024-11-01 12:42:38,599 DEBUG Stage [1/11]: Command "prlctl snapshot-list myct" started.
2024-11-01 12:42:38,634 DEBUG Stage [1/11]: Command "prlctl snapshot-list myct" output:
PARENT_SNAPSHOT_ID                  SNAPSHOT_ID

```

```

2024-11-01 12:42:38,634 DEBUG Stage [1/11]: Command "prlctl snapshot-list myct" finished.
2024-11-01 12:42:38,702 DEBUG Stage [1/11]: CT RAW HW list myct:
    cpu sockets=1 cpus=unlimited cores=unlimited VT-x hotplug accl=high mode=64 cpuunits=1000 i
    memory 512Mb hotplug
    video 0Mb 3d acceleration=off vertical sync=yes
    memory_guarantee auto
    hdd0 (+) scsi:0 image='/vz/private/653ae3d9-c475-4922-b7dc-99ee75bc2db9/root.hdd' type='exp
    venet0 (+) type='routed' ips='14.11.182.2/255.255.255.0 '
2024-11-01 12:42:38,702 DEBUG Stage [1/11]: Detected following HW list: {'disk': {'hdd0': {'scsi':

```

2.18.5.2 Falling Back to Converted Container in Case of Unsuccessful Conversion

If the conversion is a success, but the resulting virtual machine does not work as expected, you can:

1. Stop the converted virtual machine centos7:

```
# prlctl stop centos7
```

2. Assign the transferred IP address back to the converted container C2V_centos7:

```
# prlctl set C2V_centos7 --device-add <venetX|netX> --ipadd <addr>
```

3. Start the container:

```
# prlctl start C2V_centos7
```

2.18.5.3 Locale-Related Errors on macOS

When running the `c2v-convert` utility on Virtuozzo Hybrid Server, macOS users may encounter the locale-related error message while logging into Virtuozzo Hybrid Server:

```
warning: setlocale: LC_CTYPE: cannot change locale (UTF-8): No such file or directory
```

During conversion, it may cause errors as follows:

```
...AttributeError: 'NoneType' object has no attribute 'stdout'
```

or

```
...UnicodeEncodeError: 'ascii' codec can't encode character '\u201c' in position 679: ordinal not i
```

To avoid this issue, add the following to the `/etc/environment` file on Virtuozzo Hybrid Server and log into it again:

```
LANG=en_US.UTF-8
LC_ALL=en_US.UTF-8
```

2.18.5.4 Unmounted Disk After Conversion

If a container to be converted has additional disks, add information about the mount point of these disks in the configuration file unless it has been added before:

```
# prlctl set MyCT --device-set hdd1 --mnt /userdisk
```

Otherwise, the disk will be unmounted after conversion and will need to be mounted manually.

2.18.6 Usage Data Collection

The `c2v-convert` tool collects anonymous statistics about conversion attempts if the host where the tool is used participates in the [Customer Experience Program \(CEP\)](#). It helps Virtuozzo proactively identify issues in the tool's operation. The system will not gather personal data, such as hostnames, IP addresses, names, etc. The hosts already enrolled in CEP will send conversion statistics by default. If the host with the tool used on it does not participate in CEP, no statistics will be collected.

Below is a list of data gathered by CEP:

- Unique identifier based on the CTID of the container subject to conversion
- Conversion attempt date
- Conversion result: success or failure. For failed conversions, the step where the conversion failed and the reason
- The container OS template of the attempted conversion
- The container App template of the attempted conversion
- The number and type of NICs attached to the container
- The number and size of disks attached to the container
- The execution time of the conversion

Here is an example of the collected data:

```
Saving following information for CEP: {'ct_id': '1000', 'date': '2024-12-05 17:51:44.048897',
'success': True, 'reason': None, 'fail_step': 0, 'ct_os': 'centos7', 'ct_app': 'directadmin',
```

```
'nics': [{'name': 'venet0', 'type': 'routed'}, {'name': 'net0', 'network': 'Bridged'}],  
'disks': [{'name': 'hdd0', 'size': '10240'}, {'name': 'hdd1', 'size': '65536'}],  
'execution_time': 207}
```

CHAPTER 3

Managing Resources

The main goal of resource control in Virtuozzo Hybrid Server is to provide service-level management or quality of service for virtual machines and containers.

Correct resource management prevents virtual machines or containers from overusing host resources (accidentally or maliciously) and abusing other virtual environments on the host. Resource management also enables better service quality for specific virtual machines and containers if necessary.

3.1 Managing CPU Resources

You can manage the following CPU resource parameters for virtual machines and containers:

- CPU units for virtual machines and containers
- CPU affinity for virtual machines and containers
- CPU limits for virtual machines and containers
- NUMA nodes for virtual machines and containers
- CPU hotplug for virtual machines
- CPU topology for virtual machines

Detailed information on these parameters is given in the following sections.

3.1.1 Configuring CPU Units

CPU units define how much CPU time one virtual machine or container can receive in comparison with the other virtual machines and containers on the hardware node if all the CPUs of the hardware node are fully used. For example, if the container `MyCT` and the virtual machine `MyVM` are set to receive 1000 CPU units each and the container `MyCT2` is configured to get 2000 CPU units, the container `MyCT2` will get twice as much CPU time as the container `MyCT` or the virtual machine `MyVM` if all the CPUs of the Node are completely loaded.

By default, each virtual machine and container on the Node gets 1000 CPU units. You can configure the default setting using the `prlctl set` command. For example, you can run the following commands to allocate 2000 CPU units to the container `MyCT` and the virtual machine `MyVM`:

```
# prlctl set MyCT --cpuunits 2000
# prlctl set MyVM --cpuunits 2000
```

3.1.2 Configuring CPU Affinity for Virtual Machines and Containers

If your physical server has several CPUs installed, you can bind a virtual machine or container to specific CPUs so that only these CPUs are used to handle the processes running in the virtual machine or container. The feature of binding certain processes to certain CPUs is known as CPU affinity.

By default, any newly created virtual machine or container can consume the CPU time of all processors installed on the physical server. To bind a virtual machine or container to specific CPUs, you can use the `--cpumask` option of the `prlctl set` command. Assuming that your physical server has 8 CPUs, you can make the processes in the virtual machine `MyVM` and the container `MyCT` run on CPUs 0, 1, 3, 4, 5, and 6 by running the following commands:

```
# prlctl set MyVM --cpumask 0,1,3,4-6
# prlctl set MyCT --cpumask 0,1,3,4-6
```

You can specify the CPU affinity mask—that is, the processors to bind to virtual machines and containers—as separate CPU index numbers (0,1,3) or as CPU ranges (4-6). If you are setting the CPU affinity mask for a running virtual machine or container, the changes are applied on the fly.

To undo the changes made to the virtual machine `MyVM` and the container `MyCT` and set their processes to run on all available CPUs on the server, run these commands:

```
# prlctl set MyVM --cpumask all
# prlctl set MyCT --cpumask all
```

3.1.3 Configuring CPU Limits for Virtual Machines and Containers

A CPU limit indicates the maximum CPU power a virtual machine or container may get for its running processes. The container is not allowed to exceed the specified limit even if the server has enough free CPU power. By default, the CPU limit parameter is disabled for all newly created virtual machines and containers. This means that any application in any virtual machine or container can use all the free CPU power of the server.

Note: You can change which virtual machine threads—both service and activity or only activity—are limited by the parameters described below. To do this, enter the `prlsrvctl set --vm-cpulimit-type <full|guest>` command and restart running virtual machines for the changes to take effect.

To set a CPU limit for a virtual machine or container, you can use one of these options: `--cpulimit`, `--cpus`. Both options are described below in detail.

3.1.3.1 Using `--cpulimit` to Set CPU Limits

As a rule, you set a CPU limit for a virtual machine or container by using the `prlctl set --cpulimit` command. In the following example, the container `MyCT` is set to receive no more than 25% of the server CPU time even if the CPUs on the server are not fully loaded:

```
# prlctl set MyCT --cpulimit 25
```

This command sets the CPU limit for the container `MyCT` to 25% of the total CPU power of the server. The total CPU power of a server in per cent is calculated by multiplying the number of logical CPU cores installed on the server by 100%. So if a server has 2 logical CPU cores, 2 GHz each, the total CPU power will equal 200% and the limit for the container `MyCT` will be set to 500 MHz.

For example, on a hardware node with 2 logical CPU cores, 3 GHz each, the container `MyCT` will be able to get 25% of 6 GHz, that is, 750 MHz. To ensure that the container `MyCT` always has the same CPU limit on all servers, irrespective of their total CPU power, you can set the CPU limits in megahertz (MHz). For example, to make the container `MyCT` consume no more than 500 MHz on any hardware node, run the following command:

```
# prlctl set MyCT --cpulimit 500m
```

Note: For more information on setting CPU limits for virtual machines and containers, see also *CPU Limit Specifics* on page 106.

3.1.3.2 Using `--cpus` to Set CPU Limits

Another way of setting a CPU limit for a virtual machine or container is to use the `prctl set --cpus` command. In this case, you can specify how many logical CPU cores per CPU socket the virtual machine or container may use. For example, as containers have only one CPU socket (for details, see *Configuring CPU Topology for Virtual Machines* on page 109), you can allow the container `MyCT` to use only 2 cores by running this command:

```
# prctl set MyCT --cpus 2
```

To make sure that the CPU limit has been successfully set, you check `/proc/cpuinfo` in the container. For example:

```
# prctl exec MyCT cat /proc/cpuinfo | grep "cpu cores"
cpu cores      : 2
```

3.1.3.3 Using `--cpulimit` and `--cpus` Simultaneously

If you use both `--cpulimit` and `--cpus` to set the CPU limit for a virtual machine or container, the smallest limit applies. For example, running the following commands on a server with 4 CPUs, 2 GHz each, will set the limit for the container `MyCT` to 2 GHz:

```
# prctl set MyCT --cpus 2
# prctl set MyCT --cpulimit 2000m
```

3.1.3.4 CPU Limit Specifics

Internally, Virtuozzo Hybrid Server sets the CPU limit for virtual machines and containers in percent. On multi-core systems, each logical CPU core is considered to have the CPU power of 100%. So if a server has 4 CPU cores, the total CPU power of the server equals 400%.

You can also set a CPU limit in megahertz (MHz). If you specify the limit in MHz, Virtuozzo Hybrid Server uses the following formula to convert the CPU power of the server from MHz into percent: $CPULIMIT_% = 100\% * CPULIMIT_MHz / CPUFREQ$, where

- CPULIMIT_% is the total CPU power of the server in percent.
- CPULIMIT_MHz is the total CPU power of the server in megahertz.
- CPUFREQ is the CPU frequency of one core on the server.

When setting CPU limits, note the following:

- Make sure that the CPU limit you plan to set for a virtual machine or container does not exceed the total CPU power of the server. So if a server has 4 CPUs, 1000 MHz each, do not set the CPU limit to more than 4000 MHz.
- The processes running in a virtual machine or container are scheduled for execution on all server CPUs in equal shares. For example, if a server has 4 CPUs, 1000 MHz each, and you set the CPU limit for a virtual machine or container to 2000 MHz, the virtual machine or container will consume 500 MHz from each CPU.
- All running virtual machines and containers on a server cannot simultaneously consume more CPU power than is physically available on the node. In other words, if the total CPU power of the server is 4000 MHz, the running virtual machines and containers on this server will not be able to consume more than 4000 MHz, irrespective of their CPU limits. It is, however, perfectly normal that the overall CPU limit of all virtual machines and containers exceeds the Node total CPU power because most of the time virtual machines and containers consume only part of the CPU power assigned to them.

3.1.4 Binding CPUs to NUMA Nodes

On systems with a NUMA (Non-Uniform Memory Access) architecture, you can configure virtual machines and containers to use CPUs from specific NUMA nodes only. Consider the following example:

- Your physical server has 8 CPUs installed.
- The CPUs are divided into 2 NUMA nodes: NUMA node 0 and NUMA node 1. Each NUMA node has 4 CPUs.
- You want the processes in the container MyCT to be executed on the processors from NUMA node 1.

To set the container MyCT to use the processors from NUMA node 1, run the following command:

```
# prctl set MyCT --nodemask 1
```

To check that the container MyCT is now bound to NUMA node 1, use this command:

```
# prlctl list -i MyCT | grep nodemask
cpu cpus=unlimited VT-x hotplug accl=high mode=32 cpuunits=1000 ioprio=4 nodemask=1
```

To unbind the container MyCT from NUMA node 1, execute this command:

```
# prlctl set MyCT --nodemask all
```

Now the container MyCT should be able to use all CPUs on the server again.

Note: For more information on NUMA, visit <http://lse.sourceforge.net/numa>.

3.1.5 Enabling CPU Hotplug for Virtual Machines

If a guest operating system supports the CPU hotplug functionality, you can enable this functionality for the virtual machine. Once the CPU hotplug functionality is turned on, you can increase the number of CPUs available to your virtual machines even if they are running.

Currently, the following systems come with the CPU hotplug support:

- Linux operating systems based on the RHEL 5 kernel and higher (Red Hat Linux Enterprise 5, CentOS 5, and so on)
- x64 version of Windows Server 2008 R2 (Datacenter Edition)
- x64 version of Windows Server 2012 (Standard and Datacenter Edition)
- x64 version of Windows Server 2008 (Standard Edition)
- x64 version of Windows Server 2008 (Enterprise Edition)
- x64 version of Windows Server 2008 (Datacenter Edition)

By default, the CPU hotplug support is disabled for all newly created virtual machines. To enable this functionality, you can use the `--cpu-hotplug` option of the `prlctl set` command. For example, to enable the CPU hotplug support in the virtual machine MyVM that runs one of the supported operating systems, stop the virtual machine MyVM and run this command:

```
# prlctl set MyVM --cpu-hotplug on
set cpu hotplug: 1
The VM has been successfully configured.
```

Once the functionality is enabled, you can increase the number of CPUs in the virtual machine MyVM even it is

running. Assuming that your physical server has 4 CPUs installed and the processes in the virtual machine MyVM are set to be executed on two CPUs, you can run the following command to assign 3 CPUs to the virtual machine:

```
# prlctl set MyVM --cpus 3
set cpus(4): 3
The VM has been successfully configured.
```

To disable the CPU hotplug support in the virtual machine MyVM, use this command:

```
# prlctl set MyVM --cpu-hotplug off
set cpu hotplug: 0
The VM has been successfully configured.
```

The changes will come into effect on the next virtual machine start.

3.1.6 Configuring CPU Topology for Virtual Machines

In the current version of Virtuozzo Hybrid Server, you can specify CPU topology for virtual machines, i.e. the number of CPU sockets and CPU cores per socket. This can be efficient, for example, if you use a guest operating system that supports a specific number of CPU sockets (e.g., limited by a license). In such a case, you can set the VM to use as many CPU cores as allowed by the guest OS and achieve near-native guest OS performance. The overall number of CPU cores available to a virtual machine is calculated by multiplying the number of CPU sockets by the number of CPU cores per socket. It can be no greater than the number of CPU cores on the host physical server.

By default, a virtual machine is created with one CPU socket and two CPU cores.

Note: In turn, a container has only one CPU socket, and this parameter cannot be changed.

You can change the number of CPU sockets and CPU cores per socket of a virtual machine using the `--cpu-sockets` and `--cpus` options of the `prlctl set` command. For example, if your physical server has 4 CPU cores, you can allow the virtual machine MyVM to use 2 CPU sockets and 2 CPU cores per socket by running the following command:

```
# prlctl set MyVM --cpu-sockets 2 --cpus 2
```

If a virtual machine is running, the changes will take effect after it is restarted.

To check the current CPU topology of a VM, run the following command:

```
# prlctl list MyVM -i | grep "cpu"  
cpu sockets=2 cpus=2 cores=2 <...>
```

3.2 Managing Disk Quotas

You can limit disk space that individual users and groups in a container can use with standard Linux tools from the quota package.

Before you can set disk quotas in a container, you will need to enable them for this container as follows:

1. Set QUOTAUGIDLIMIT to 1 in container configuration file (/etc/vz/conf/<UUID>.conf) or run the command `prlctl set <UUID> --quotaugidlimit 1`.
2. Restart the container.

3.3 Managing Virtual Disks

In Virtuozzo Hybrid Server, you can resize virtual disks and compact them (reduce their size on the physical hard drive). These operations are described in the following subsections in detail.

3.3.1 Resizing Virtual Disks

Before resizing a virtual disk, note the following requirements and restrictions:

- The virtual machine that uses the virtual disk to be resized must not have any snapshots.
- The virtual disk size shown inside the virtual machine or container may differ from the size the virtual disk occupies on server's physical disk.
- In case disk size is increased, the added disk space is added as unallocated. You can use standard tools of the guest OS to allocate the added space.
- You cannot reduce a container's virtual disk by more than 16TB in total if the container is running. To do this, stop the container first and run `vzctl set <CTID> --save --offline --diskspace <new_disk_size>`.
- You cannot reduce XFS file systems (the default choice for CentOS 7 and Red Hat Enterprise Linux 7).

3.3.1.1 Resizing Virtual Disks Offline

You can resize virtual hard disks of stopped virtual machines or containers with the `prl_disk_tool resize --size` command. For example:

```
# prl_disk_tool resize --hdd /vz/vmprivate/d35d28e5-11f7-4b3f-9065-8fef6178bc5b/ \
harddisk.hdd --size 100G
```

3.3.1.2 Resizing Virtual Disks Online

Warning: Reducing virtual disk capacity of running virtual machines is prohibited.

You can resize virtual hard disks of running containers with the `prl_disk_tool resize --size` command. For example:

```
# prl_disk_tool resize --hdd /vz/private/85094686-4829-4ffa-a230-961c32c82bbe/root.hdd --size 100G
```

To resize virtual hard disks of running virtual machines, use the `prlctl set --device-set --size --no-fs-resize` command. For example, to resize the `hdd0` hard disk of the `MyVM` virtual machine to 100G, execute:

```
# prlctl set MyVM --device-set hdd0 --size 100G --no-fs-resize
```

With the `--no-fs-resize` option specified, the last partition on the hard disk is not resized.

3.3.1.3 Checking the Minimum Disk Capacity

If, before reducing disk capacity, you want to know the minimum size to which it can be reduced, use the `prl_disk_tool resize --info` command. For example, if the disk `hdd0` of the virtual machine `MyVM` is emulated by the image `/vz/vmprivate/d35d28e5-11f7-4b3f-9065-8fef6178bc5b/harddisk.hdd`, run the following command:

```
# prl_disk_tool resize --info --hdd /vz/vmprivate/d35d28e5-11f7-4b3f-9065- \
8fef6178bc5b/harddisk.hdd
Disk information:
...
Minimum: 2338M
...
```

3.3.2 Compacting Disks

Warning: Only use `prl_disk_tool` on disks of stopped virtual machines.

In Virtuozzo Hybrid Server, you can reduce the space your virtual machines and containers occupy on the physical server's disk drive by compacting their virtual disks. Doing so frees up server disk space for hosting more virtual machines and containers.

To compact a virtual disk, you can use the `prl_disk_tool compact` command. For example, to compact the disk `/vz/vmprivate/d35d28e5-11f7-4b3f-9065-8fef6178bc5b/harddisk.hdd`, run this command:

```
# prl_disk_tool compact --hdd /vz/vmprivate/d35d28e5-11f7-4b3f-9065-8fef6178bc5b/ \
harddisk.hdd/
```

To check the space that was freed by compacting the virtual disk, you can use standard Linux utilities (for example, `df`).

3.4 Managing Network Accounting and Bandwidth

This section explains how to perform the following tasks in Virtuozzo Hybrid Server:

- Configure network classes
- View network traffic statistics
- Toggling network bandwidth management
- Configure bandwidth limits

3.4.1 Network Traffic Parameters

The table below summarizes the network traffic parameters that you can control in Virtuozzo Hybrid Server.

Parameter	Description
traffic_shaping	If set to yes, traffic limitations for outgoing traffic are set for virtual machines and containers. The default is no.
bandwidth	This parameter lists all network adapters installed on the hardware node and their bandwidth.
totalrate	This parameter defines the bandwidth to allocate for each network class. It is active if traffic shaping is turned on.
rate	If traffic shaping is turned on, this parameter specifies the bandwidth guarantee for virtual machines and containers.
ratebound	If this parameter is set to yes, the bandwidth guarantee (the global rate parameter) is also the limit for the virtual machine or container, and the virtual machine or container cannot borrow the bandwidth from the totalrate bandwidth pool.

3.4.2 Configuring Network Classes

Virtuozzo Hybrid Server allows you to track the inbound and outbound network traffic and shape the outgoing traffic for virtual machines and containers. To provide the ability to distinguish between types of traffic, e.g., domestic and international, a concept of network classes is introduced. IP address ranges fall into several network classes. Each network class represents traffic to specific resources. Access to such resources is charged at a different rate than the rest of the traffic.

Classes are specified in the `/etc/vz/conf/networks_classes` file. The file is in the ASCII format, and all empty lines and lines starting with the `#` sign are ignored. Other lines have the following format:

```
<class_id> <IP_address>/<prefix_length>
```

where `<class_id>` defines the network class ID, and the `<IP_address>/<prefix_length>` pair defines the range of IP addresses for this class. There may be several lines for each class.

Classes 0 and 1 have special meanings:

- Class 0 defines the IP address range for which no accounting is performed. Usually, it corresponds to the hardware node subnet (the node itself and its virtual machines and containers). Setting up class 0 is not required; however, its correct setup improves performance.
- Class 1 is defined by Virtuozzo Hybrid Server to match any IP address. It must be always present in the network classes definition file. Therefore, it is suggested not to change the default line in the

networks_classes file.

```
1 0.0.0.0/0
```

If your virtual machines and containers are using IPv6 addresses, you can also add the following line to this file:

```
1 ::/0
```

Other classes should be defined after class 1. They represent exceptions from the “matching-everything” rule of class 1. The example below illustrates a possible configuration of the network classes definition file containing rules for both IPv4 and IPv6 addresses:

```
# Hardware node networks
0 192.168.0.0/16
0 fe80::/64
# any IP address (all traffic)
1 0.0.0.0/0
1 ::/0
# class 2 - addresses for the "foreign" traffic
2 10.0.0.0/8
2 2001:db8::/64
# inside "foreign" network there
# is a hole belonging to "local" traffic
1 10.10.16.0/24
1 2001:db8:3333::/64
```

In this example, IPv4 addresses in the range of 192.168.0.0 to 192.168.255.255 and IPv6 addresses in the range of fe80:: to fe80::ffff:ffff:ffff:ffff are treated as class 0 addresses and no accounting is done for the traffic from virtual machines and containers destined to these addresses.

Class 2 matches the following IP addresses:

- IPv4 addresses from 10.0.0.0 to 10.255.255.255 with the exception of addresses in the sub-range of 10.10.16.0 to 10.10.16.255, which are treated as class 1.
- IPv6 addresses from 2001:db8:: to 2001:db8::ffff:ffff:ffff:ffff with the exception of addresses in the sub-range of 2001:db8:3333:: to 2001:db8:3333::ffff:ffff:ffff:ffff, which are also treated as class 1.

All other IP addresses (both IPv4 and IPv6) belong to class 1.

To apply changes after editing the /etc/vz/conf/networks_classes file, restart either the virtual machine(s) or/and container(s) for which changes have been made or the hardware node itself if the changes are global.

3.4.3 Viewing Network Traffic Statistics

In Virtuozzo Hybrid Server, you can view the current network traffic statistics for virtual machines and containers using the `vznetstat` utility. For example:

```
# vznetstat
UUID          Net.Class  Input(bytes)  Input(pkts)  Output(bytes)  Output(pkts)
0             0          566093064    2800575     3120481       41736
47406484...  0          67489        155         8033          110
fbb30afa-...  0          9369         78          12692         71
```

By default, `vznetstat` shows network statistics for both virtual machines and containers. Keep in mind that the `vznetstat` utility displays statistics only about virtual machines and containers that were started at least once.

The `vznetstat` utility displays the following information:

Column	Description
UUID	UUID assigned to virtual machine or container.
Net.Class	ID of the network class for which network statistics is calculated.
Input(bytes)	Amount of incoming traffic, in bytes.
Input(pkts)	Amount of incoming traffic, in packets.
Output(bytes)	Amount of outgoing traffic, in bytes.
Output(pkts)	Amount of outgoing traffic, in packets.

For example, from the command output above, you can see that around 9 MB of data were uploaded to the container `MyCT`, (2) about 12 MB were downloaded from it, and all the traffic was exchanged with servers from class 0 networks.

If necessary, you can view network traffic statistics separately for virtual machine or container by passing the `-t` option to `vznetstat`:

- For containers only:

```
# vznetstat -t ct
CTID          Net.Class  Input(bytes)  Input(pkts)  Output(bytes)  Output(pkts)
0             0          566093064    2800575     3120481       41736
fbb30afa-...  0          9369         78          12692         71
```

- For virtual machines only:

```
# vznetstat -t vm
UUID          Net.Class  Input(bytes)  Input(pkts)  Output(bytes)  Output(pkts)
```

```

0          0          566093064    2800575    3120481    41736
47406484... 0          67489       155        8033       110

```

You can also view network statistics for a particular virtual machine or container by specifying its ID after the `-v` option, for example:

```

# vznetstat -v fbb30afa-e770-4081-9d9e-6b9c262eb091
UUID          Net.Class  Input(bytes)  Input(pkts)  Output(bytes)  Output(pkts)
fbb30afa-...  0          9369          78           12692          71

```

This command displays statistics only for the container `MyCT`.

3.4.4 Configuring Traffic Shaping

Traffic shaping (also known as network bandwidth management) allows you to control what network bandwidth a virtual machine or container may use for outgoing traffic. This feature is disabled by default.

Note the following:

- Traffic within a host cannot be shaped in the current version of Virtuozzo Hybrid Server. This includes traffic between virtual machines and containers on the same host and between those and the host itself.
- Incoming traffic cannot be shaped for virtual machines and containers in the current version of Virtuozzo Hybrid Server.
- VM and container traffic is shaped by marking packets' `fwmark` field. If the hardware node uses custom packet filtering rules set with the `MARK` and `CONNMARK` iptables modules, the traffic shaping feature of Virtuozzo Hybrid Server will not work. Traffic statistics reported by `vznetstat`, however, will be correct even if the node uses custom traffic marks.

The following parameters control traffic shaping in Virtuozzo Hybrid Server:

- `TRAFFIC_SHAPING`, enables and disables traffic shaping.
- `BANDWIDTH`, sets bandwidth for specific network adapters.
- `TOTALRATE`, sets the size of a bandwidth pool divided between virtual machines and containers on the host.
- `RATEMPU`, limits packet rate in addition to byte rate.
- `RATE`, sets a bandwidth guarantee for virtual machines and containers.

- RATEBOUND, forces RATE as a limit.

Traffic shaping in Virtuozzo Hybrid Server works as follows. The bandwidth pool for a given network class (set by TOTALRATE) is divided among the virtual machines and containers transmitting data proportionally to their RATE settings. If the sum of RATE values of all virtual machines and containers transmitting data does not exceed TOTALRATE, each virtual machine or container gets the bandwidth equal to or greater than its RATE value (unless RATEBOUND is enabled for the VM or container). If the sum of RATE values of all virtual machines and containers transmitting data exceeds the TOTALRATE value, each virtual machine or container may get less than its RATE value.

To enable and configure traffic shaping, do the following:

1. Set the value of TRAFFIC_SHAPING to yes in the global configuration file `/etc/vz/vz.conf`.
2. Set the parameters BANDWIDTH, TOTALRATE in `/etc/vz/vz.conf`.
3. If required, set the optional parameters RATEMPU, RATE, RATEBOUND in `/etc/vz/vz.conf`.
4. If required, set RATE and RATEBOUND for specific virtual machines and containers with `prlctl set --rate` and `prlctl set --ratebound` commands.
5. To apply changes, restart either the virtual machines and containers for which changes have been made or the hardware node itself if the changes are global.

The following sections provide more details on and explain how to set traffic shaping parameters listed above.

3.4.4.1 Setting the BANDWIDTH Parameter

The BANDWIDTH parameter is used for shaping traffic of specific network adapters. For example, for two Fast Ethernet cards, a typical setting may look like `enp0s5 enp0s6:100000` where `enp0s5` and `enp0s6` are network adapter names. By default, the parameter is set to `100000` which corresponds to a 100 Mbps Fast Ethernet card.

3.4.4.2 Setting the TOTALRATE Parameter

The TOTALRATE parameter specifies the size of a bandwidth pool for specific network classes on the host. Virtual machines and containers can borrow bandwidth from the pool for communicating with hosts from the corresponding network class. The parameter thus limits the total available outgoing traffic for a network class that virtual machines and containers can consume.

The parameter is set as `<NIC>:<network_class>:<bandwidth_in_Kbps>`. For example, to set the pool size to 4 Mbps for network class 1 on the Ethernet adapter `enp0s5`, set `TOTALRATE` to `enp0s5:1:4000`. Multiple entries can be separated by spaces, e.g., `enp0s5:1:4000 enp0s6:2:8000`.

3.4.4.3 Setting the RATEMPU Parameter

The optional `RATEMPU` parameter (where “MPU” stands for “minimum packet unit”) limits the packet rate by making packets smaller than MPU in size consume HTB tokens. With it, small packets can be accounted as larger ones and limited by `TOTALRATE` and `RATE` parameters. Approximately, the maximum packets per second rate can be calculated as `TOTALRATE / RATEMPU`.

This parameter has the following syntax: `<NIC>:<network_class>[:<MPU_in_bytes_per_packet>]`. If the part `<MPU_in_bytes_per_packet>` is omitted, the default value of 1000 bytes is used. Multiple entries can be separated by spaces, e.g., `enp0s5:1:2000 enp0s6:2:4000`. To set the `RATEMPU` parameter for all known Ethernet devices set `<NIC>` to an asterisk (*). For example, to set the minimal packet size to 2 Kb for network class 1 on all the Ethernet adapters on the node, change the value to `*:1:2000`.

3.4.4.4 Setting the RATE and RATEBOUND Parameters

The optional `RATE` parameter allows you to guarantee virtual machines and containers outgoing bandwidth to destinations in a specific network class on a specific Ethernet device. The guaranteed bandwidth is not a limit (unless the `RATEBOUND` parameter is also set to on, see below). A virtual machine or container can additionally obtain unused bandwidth from the bandwidth pool defined by `TOTALRATE`.

You can set the guaranteed bandwidth in two ways:

1. For all virtual machines and containers on the host by setting `RATE` in the global configuration file `/etc/vz/vz.conf`.

The parameter is set as `<NIC>:<network_class>:<bandwidth_in_Kbps>`. For example, to guarantee all virtual machines and containers on the host the bandwidth of at least 8 Kbps for outgoing traffic in network class 1 on the Ethernet device `enp0s5`, set the `RATE` parameter to `enp0s5:1:8`.

2. For specific virtual machines or containers by means of the `prlctl set --rate` command.

For example, to guarantee the container `MyCT` the bandwidth of at least 16 Kbps for outgoing traffic in network class 1, run

```
# prlctl set MyCT --rate 1:16
```

This command sets the bandwidth for the default network adapter only. If you need to set bandwidth for other network adapters, set RATE in `/etc/vz/vz.conf`.

Note: It is recommended to increase RATE value in 8 Kbps increments and set it to at least 8 Kbps.

The optional RATEBOUND parameter specifies whether the network bandwidth guaranteed by RATE is also a limit. By default, this feature is disabled for all newly created virtual machines and containers so they may additionally obtain unused bandwidth from the pool set by TOTALRATE.

You can limit bandwidth of virtual machines and containers to the guaranteed value as follows:

1. For all virtual machines and containers on the host by setting RATEBOUND in the global configuration file `/etc/vz/vz.conf` (omitted by default).
2. For specific virtual machines or containers by means of the `prlctl set --ratebound` command. For example:

```
# prlctl set MyCT --ratebound yes
```

If set, values of RATE and RATEBOUND provided for specific virtual machines and containers are chosen over global values in `/etc/vz/vz.conf`.

3.4.4.5 Traffic Shaping Example

The example below illustrates a scenario when the containers MyCT1 and MyCT2 have RATEBOUND set to no, and the virtual machine MyVM has RATEBOUND set to yes. With the default TOTALRATE of 4096 Kbps and RATE of 8 Kbps, the bandwidth pool will be distributed as follows:

MyCT1	MyCT2	MyVM	Consumed Bandwidth
transmits	idle	idle	MyCT1: 4096 Kbps
idle	idle	transmits	MyVM: 8 Kbps
transmits	transmits	idle	MyCT1: 2048 Kbps MyCT2: 2048 Kbps
transmits	idle	transmits	MyCT1: 4032 Kbps MyVM: 8 Kbps
transmits	transmits	transmits	MyCT1: 2016 Kbps MyCT2: 2016 Kbps MyVM: 8 Kbps

3.5 Managing Disk I/O Parameters

This section explains how to manage disk input and output (I/O) parameters in Virtuozzo Hybrid Server systems.

Note that I/O and IOPS limiting only works for supported I/O schedulers. The scheduler type is stored in `/sys/block/<device>/queue/scheduler`. For example:

```
# cat /sys/block/sda/queue/scheduler
noop deadline [cfq]
```

The I/O scheduler used is marked by square brackets. In this example, it is `cfq`.

The following I/O schedulers are supported:

- CFQ. A separate block device line with counters is added to the `iostat` file. For example:

```
# cat /proc/bc/100/iostat
flush 100 . 0 0 0 0 0 7389 1893968 0 0
fuse 100 . 0 0 0 0 0 0 0 0
sda 100 . 0 0 0 9000 1843380 245216 55845488 245028 188
```

- Deadline. The counters are added to the values in the `flush` line.

I/O and IOPS limiting does not work for devices with the `noop` I/O scheduler or without one (e.g., logical devices, CEPH RBD devices, and such).

3.5.1 Configuring Priority Levels for Virtual Machines and Containers

In Virtuozzo Hybrid Server, you can configure the disk I/O (input/output) priority level of virtual machines and containers. The higher the I/O priority level, the more time the virtual machine or container will get for its disk I/O activities as compared to the other virtual machines and containers on the hardware node. By default, any virtual machine or container on the hardware node has the I/O priority level set to 4. However, you can change the current I/O priority level in the range from 0 to 7 using the `--ioprio` option of the `prlctl set` command. For example, you can issue the following command to set the I/O priority of the container `MyCT` and the virtual machine `MyVM` to 6:

```
# prlctl set MyCT --ioprio 6
# prlctl set MyVM --ioprio 6
```

To check the I/O priority level currently applied to the container `MyCT` and the virtual machine `MyVM`, you can execute the following commands:

- For container MyCT:

```
# grep IOPRIO /etc/vz/conf/fbb30afa-e770-4081-9d9e-6b9c262eb091.conf
IOPRIO="6"
```

- For the virtual machine MyVM:

```
# prctl list MyVM --info | grep ioprio
cpu cpus=2 VT-x accl=high mode=32 ioprio=6 iolimit='0'
```

3.5.2 Limiting Disk I/O Bandwidth

In Virtuozzo Hybrid Server, you can configure the bandwidth virtual machines and containers are allowed to use for their disk input and output (I/O) operations. Limiting the disk I/O bandwidth can help you prevent the situations when high disk activities in one virtual machine or container (generated, for example, by transferring huge amounts of data to/from the virtual machine or container) can slow down the performance of other virtual machines and containers on the hardware node.

By default, the I/O bandwidth limit for all newly created virtual machines and containers is set to 0, which means that no limits are applied. To limit the disk I/O bandwidth for a virtual machine or container, use the `--iolimit` option of the `prctl set` command. For example, to set the I/O bandwidth limit for the container MyCT to 10 MB/s:

```
# prctl set MyCT --iolimit 10
```

By default, the limit is set in megabytes per second. In addition, you can use the following suffixes to indicate other measurement units:

- G sets the limit in gigabytes per second (1G).
- K sets the limit in kilobytes per second (10K).
- B sets the limit in bytes per second (10B).

Note: In the current version of Virtuozzo Hybrid Server, the maximum I/O bandwidth limit you can set for a virtual machine or container is 2 GB per second.

To check that the I/O speed limit has been successfully applied to the container MyCT, use the `prctl list` command:

```
# prlctl list MyCT -o iolimit
IOLIMIT
10485760
```

At any time, you can remove the I/O bandwidth limit set for container MyCT by running this command:

```
# prlctl set MyCT --iolimit 0
```

3.5.3 Limiting the Number of I/O Operations per Second

In Virtuozzo Hybrid Server, you can limit the maximum number of disk input and output operations per second virtual machines and containers are allowed to perform (known as the IOPS limit). You may consider setting the IOPS limit for virtual machines and containers with high disk activities to ensure that they do not affect the performance of other virtual machines and containers on the Node.

Note: By default all I/O inside containers is cached and the direct access flag (O_DIRECT) is ignored when opening files. This significantly reduces the number of IOPS required for container workload and helps avoid I/O bottlenecks on the Node. For instructions on how to configure honoring of the O_DIRECT flag inside containers, see *Setting the Direct Access Flag Inside Containers* on page 123 further.

By default, IOPS is not limited for newly created virtual machines and containers. To set the IOPS limit, you can use the `--iopslimit` option of the `prlctl set` command. For example, to allow the container MyCT and the virtual machine MyVM to perform no more than 100 disk I/O operations per second, you can run the following commands:

```
# prlctl set MyCT --iopslimit 100
# prlctl set MyVM --iopslimit 100
```

To ensure that the IOPS limit has been successfully applied, use the `prlctl list -i` command. For example:

```
# prlctl list MyCT -i | grep iopslimit
<...> iopslimit=100
```

At any time, you can remove the set IOPS limits by running this command:

```
# prlctl set MyCT --iopslimit 0
# prlctl set MyVM --iopslimit 0
```


3.5.3.1 Setting the Direct Access Flag Inside Containers

You can configure honoring of the `O_DIRECT` flag inside containers with the `sysctl` parameter `fs.odirect_enable`:

- To ignore the `O_DIRECT` flag inside a container, set `fs.odirect_enable` to `0` in that container.
- To honor the `O_DIRECT` flag inside the container, set `fs.odirect_enable` to `1` in that container.
- To have a container inherit the setting from the hardware node, set `fs.odirect_enable` to `2` in that container (default value). On the hardware node, `fs.odirect_enable` is `0` by default.

Note: The `fs.odirect_enable` parameter on the Node only affects honoring of the `O_DIRECT` flag in containers and not on the Node itself where the `O_DIRECT` flag is always honored.

3.5.4 Viewing Disk I/O Statistics

In Virtuozzo Hybrid Server, you can view disk input and output (I/O) statistics for all processes on the host. To do this:

1. Run the `vmstat` utility.
2. Press **F2** or **S** to switch to the **Setup** menu.
3. In the **Setup** column, choose **Columns**.
4. In **Available Columns**, choose from the following parameters to add to the output (**Active Columns**):

Parameter	Description	Column
RBYTES	Number of bytes read for the process.	IO_RBYTES
WBYTES	Number of bytes written for the process.	IO_WBYTES
IO_READ_RATE	Process read rate, in bytes per second.	DISK READ
IO_WRITE_RATE	Process write rate, in bytes per second.	DISK WRITE
IO_RATE	Process total I/O rate, in bytes per second.	DISK R/W
IO_PRIORITY	Process I/O priority.	IO

To add a parameter, select it and press **F5** or **Enter**. To remove a parameter from **Active Columns**, select it and press **F9**.

5. When you finish managing columns, press **F10** to save the changes and view the output.

3.5.5 Setting I/O Limits for Backup and Migration Operations

Backup and migration operations with containers and virtual machines can generate a high I/O load on the server, thus reducing the performance of other virtual environments or the server itself. You can avoid such situations by setting I/O limits for these operations.

To set an I/O limit, do the following:

1. In the `/etc/vz/vz.conf` global configuration file, locate the following section:

```
# VZ Tools limits
<...>
# Uncomment next line to specify required disk IO bandwidth in Bps (10485760 - 10MBps)
# VZ_TOOLS_IOLIMIT=10485760
```

2. Uncomment the `VZ_TOOLS_IOLIMIT` parameter, and set the I/O limit for backup and migration operations in bytes per second.
3. Save the file.

When setting I/O limits, pay attention to the following:

- `VZ_TOOLS_IOLIMIT` is a global parameter that has effect on all virtual environments on the server.
- The `VZ_TOOLS_IOLIMIT` parameter controls the I/O load only for backup and migration operations. Restore operations are not limited.
- Simultaneous operations do not share the limit and are limited separately.
- For migration, only the limit set on the source server applies, while the limit set on the destination server is ignored.

3.5.6 Improving Disk I/O Performance for Virtual Machines

In case of VM virtual disks mapped to QCOW2 disk images, the disk space is only allocated when it is actually needed by the VM and the cache size depends on how much space is allocated. In Virtuozzo Hybrid Server, you can significantly improve disk I/O performance by means of cache policies applied to VM disks based on their allocated space size.

The default cache policy is used for typical VM workloads and disk sizes up to 1TB. If a guest needs

continuous access to an entire disk that is larger than 1TB, the default cache size will not be sufficient and will reduce disk I/O performance. In this case, you can increase cache size to speed up disk access as follows:

1. In the `/etc/libvirt/qemu.conf` configuration file, uncomment and set the `qcow2_cache_policy` parameter to `disk`.
2. Restart the `libvirtd` daemon to apply the changes.

```
# systemctl restart libvirtd
```

The disk policy will provide a cache large enough to handle the entire disk workload.

3.6 Managing Container Memory Parameters

This section describes the VSwap memory management system. You will learn to do the following:

- Configure the main VSwap parameters for containers.
- Set the memory allocation limit in containers.
- Configure OOM killer behavior.
- Enhance the VSwap functionality.

3.6.1 Configuring Main VSwap Parameters

Virtuozzo Hybrid Server utilizes the VSwap scheme for managing memory-related parameters in containers. Like many other memory management schemes used on standalone Linux computers, this scheme is based on two main parameters:

- `RAM` determines the total size of RAM that can be used by the processes of a container.
- `swap` determines the total size of swap that can be used by a container for swapping out memory once the RAM is exceeded.

The memory management scheme works as follows:

1. You set for a container a certain amount of RAM and swap space that can be used by the processes running in the container.
2. When the container exceeds the RAM limit set for it, swapping starts. The process is similar to that on a standalone computer. Container's swap space resides in a swap file on the physical node. When the

swap-out for a container starts, the container starts to consume physical swap. If a swap file is not configured on the node, the `SWAPPAGES` parameter is ignored.

3. Once the container exceeds its swap limit, the system invokes the OOM Killer for this container.
4. The OOM Killer chooses one or more processes running in the affected container and forcibly kills them.

By default, any newly created container starts using the new memory management scheme. To find out the amount of RAM and swap space set for a container, you can check the values of the `PHYS_PAGES` and `SWAPPAGES` parameters in the container configuration file, for example:

```
# grep PHYS_PAGES /etc/vz/conf/26bc47f6-353f-444b-bc35-b634a88dbbcc.conf
PHYS_PAGES="65536:65536"
# grep SWAPPAGES /etc/vz/conf/26bc47f6-353f-444b-bc35-b634a88dbbcc.conf
SWAPPAGES="65536"
```

In this example, the value of the `PHYS_PAGES` parameter for the container `MyCT` is set to 65536. The `PHYS_PAGES` parameter displays the amount of RAM in 4-KB pages, so the total amount of RAM set for the container `MyCT` equals to 256 MB. The value of the `SWAPPAGES` parameter is also set to 256 MB.

To configure the amounts of RAM and swap space for the container `MyCT`, use the `--memsize` and `--swappages` options of the `prlctl set` command. For example, you can execute the following command to set the amount of RAM and SWAP in the container `MyCT` to 1 GB and 512 MB, respectively:

```
# prlctl set MyCT --memsize 1G --swappages 512M
```

3.6.2 Configuring Container Memory Guarantees

A memory guarantee is a percentage of container's RAM that it is guaranteed to have.

Important: The total memory guaranteed to all running virtual environments on the host must not exceed host's physical RAM size. If starting a virtual environment with a memory guarantee would increase the total memory guarantee on the host beyond host's physical RAM size, the virtual environment will not start. If setting a memory guarantee for a running virtual environment would increase the total memory guarantee on the host beyond host's physical RAM size, the memory guarantee will not be set.

For containers, the memory guarantee value is set to 0% by default. To change the default value, use the `prlctl set --memguarantee` command. For example:

```
# prlctl set MyCT --memguarantee 80
```

To revert to the default setting, run

```
# prlctl set MyCT --memguarantee auto
```

3.6.3 Configuring Container Memory Allocation Limit

When an application starts in a container, it allocates a certain amount of memory for its needs. Usually, the allocated memory is much more than the application actually requires for its execution. This may lead to a situation when you cannot run an application in the container even if it has enough free memory. To deal with such situations, the VSwap memory management scheme introduces the new `vm_overcommit` option. Using it, you can configure the amount of memory applications in a container may allocate, irrespective of the amount of RAM and swap space assigned to the container.

The amount of memory that can be allocated by applications of a container is the sum of RAM and swap space set for this container multiplied by a memory overcommit factor. In the default (basic) container configuration file, this factor is set to 1.5. For example, if a container is based on the default configuration file and assigned 1 GB of RAM and 512 MB of swap, the memory allocation limit for the container will be 2304 MB. You can configure this limit and set it, for example, to 3 GB by running this command:

```
# vzctl set MyCT --vm_overcommit 2 --save
```

This command uses the factor of 2 to increase the memory allocation limit to 3 GB:

$(1 \text{ GB of RAM} + 512 \text{ MB of swap}) * 2 = 3 \text{ GB}$

Now applications in the container `MyCT` can allocate up to 3 GB of memory, if necessary.

3.6.4 Configuring Container OOM Killer Behavior

The OOM killer selects one or more container processes to end based on the badness score reflected in `/proc/<PID>/oom_score`. The badness score is calculated from process memory, total memory, and badness score adjustment, and then clipped to the range from 0 to 1000. Each point of the badness score stands for one thousandth of container memory. The process to be killed is the one with the highest resulting badness score.

The OOM killer can be configured by means of the standard `systemd` parameter `OOMScoreAdjust` that adjusts badness score for a process, modifying its `/proc/<PID>/oom_score_adj` value. The parameter accepts integers

between -1000 (processes of this unit will not be killed) and 1000 (processes of this unit will very likely be killed under memory pressure). As with the badness score itself, each adjustment point stands for 1/1000 of total container memory. In an out-of-memory situation, the adjustment will guarantee that the process will be allowed to occupy at least `<oom_score_adj>` thousandths of container memory while there are other processes with higher badness running in the container.

To adjust the badness score of service's processes, add the `OOMScoreAdjust` parameter to its configuration file. For example:

```
[Service]
...
OOMScoreAdjust=500
...
```

To apply the changes if the service is running, first reload systemd configuration with `systemctl daemon-reload`, then restart the service itself with `systemctl restart <service>`.

3.6.5 Tuning VSwap

The VSwap management scheme can be extended by using UBC parameters. For example, you can set the `numproc` parameter to configure the maximal number of threads a container may create or the `numfile` parameter to specify the number of files that may be opened by all processes in the container.

3.7 Managing Virtual Machine Memory Parameters

This section describes how to configure memory parameters available for virtual machines:

- Memory size
- Video memory size
- Memory hotplugging
- Memory guarantees

3.7.1 Configuring Virtual Machine Memory Size

To increase or reduce the amount of memory that will be available to the virtual machine, use the `--memsize` option of the `prlctl set` command. The following example shows how to increase the RAM of the virtual machine `MyVM` from 1GB to 2GB and check that the new value has been successfully set:

```
# prlctl list -i MyVM | grep memory
memory 1024Mb
# prlctl set MyVM --memsize 2048
Set the memsize parameter to 2048Mb
The VM has been successfully configured.
# prlctl list -i MyVM | grep memory
memory 2048Mb
```

The changes are saved in the VM configuration file and applied to the VM on start. If the VM is running, it will need to be rebooted. To be able to increase or reduce virtual machine RAM size without reboot, enable memory hotplugging as described in [Enabling Virtual Machine Memory Hotplugging](#) on page 130.

Note: The value set with `prlctl --memsize` is not reported inside the VM as physical or other RAM size. A user logged in to the guest OS will see as much physical RAM as can be obtained by fully deflating the balloon (see `MaxNumaSize` in [Enabling Virtual Machine Memory Hotplugging](#) on page 130). The balloon size is not reported inside the VM as well. However, if the balloon is not fully deflated, a part of the reported physical RAM will appear to be occupied at all times (by what is in fact the balloon).

3.7.2 Configuring Virtual Machine Video Memory Size

To set the amount of video memory to be available to the virtual machine's video card, use the `--videosize` option of the `prlctl set` command. Assuming that the current video memory size of the virtual machine `MyVM` is set to 32 MB, you can increase it to 64 MB by running the following command:

```
# prlctl set MyVM --videosize 64
```

To check that the new value has been successfully set, use this command:

```
# prlctl list -i MyVM | grep video
video 64Mb
```

3.7.3 Enabling Virtual Machine Memory Hotplugging

Memory hotplugging allows increasing or reducing virtual machine RAM size on the fly, without the need to reboot the VM. Memory hotplugging is implemented as a combination of ballooning and addition/removal of virtual DIMM slots.

The algorithm is as follows. When a command to increase VM memory size to `RAM_size` is run (as described in *Configuring Virtual Machine Memory Size* on page 129), the memory is first expanded by deflating the VM's balloon. The balloon deflation limit, `MaxNumaSize`, is calculated automatically according to the formula

```
MaxNumaSize = (RAM_size + 4GB) rounded up to a multiple of 4GB
```

If fully deflating the balloon is not enough to obtain `RAM_size` (that is, `RAM_size` exceeds `MaxNumaSize`), then memory is further expanded by adding virtual DIMM slots (up to twice the `MaxNumaSize`) and `MaxNumaSize` is set equal to `RAM_size` (that is, the maximum balloon size grows as well). When a command to decrease VM memory size is run, the memory is shrunk by inflating the VM's balloon. The added virtual DIMM slots remain until VM restart. After restart, the VM has memory equal to `RAM_size`.

This feature is only supported for virtual machines with at least 1GB of RAM and is disabled by default. To enable it for a virtual machine (e.g., `MyVM`):

1. Make sure the VM is stopped.
2. Enable memory hotplugging for the VM:

```
# prlctl set MyVM --mem-hotplug on
```

3. Start the VM.

Now virtual machine RAM size can be increased and decreased with the `prlctl set --memsize` command without rebooting the VM.

3.7.4 Configuring Virtual Machine Memory Guarantees

A memory guarantee is a percentage of virtual machine's RAM that the VM is guaranteed to have.

Important: The total memory guaranteed to all running virtual environments on the host must not exceed host's physical RAM size. If starting a virtual environment with a memory guarantee would increase the total memory guarantee on the host beyond host's physical RAM size, the virtual environment will not start. If setting a memory guarantee for a running virtual environment would increase the total memory guarantee

on the host beyond host's physical RAM size, the memory guarantee will not be set.

For virtual machines, the memory guarantee value is set to 80% by default. To change the default value, use the `prlctl set --memguarantee` command. For example:

```
# prlctl set MyVM --memguarantee 60
```

To revert to the default setting, run

```
# prlctl set MyVM --memguarantee auto
```

Note: Virtual machines with memory guarantees can only be started with `prlctl start`. Starting such VMs differently (e.g., using `virsh`) will result in memory guarantees not being applied.

3.8 Managing Container Resource Configuration

Any container is configured by means of its own configuration file. You can manage container configurations in a number of ways:

1. Using configuration sample files shipped with Virtuozzo Hybrid Server. These files are used when a new container is being created (for details, see [Virtuozzo Containers](#) on page 3). Currently, the following configuration sample files are provided:
 - `basic` for creating standard containers.
 - `confixx` for creating containers that are to run the Confixx control panel.
 - `vswap.plesk` for creating containers with the Plesk control panel.
 - `vswap.256MB` for creating containers with 256 MB of main memory.
 - `vswap.512Mb` for creating containers with 512 MB of main memory.
 - `vswap.1024Mb` for creating containers with 1024 MB of main memory.
 - `vswap.2048Mb` for creating containers with 2048 MB of main memory.

Note: Configuration sample files cannot contain spaces in their names.

Any sample configuration file can also be applied to an existing container. You would do this if, for example, you want to upgrade or downgrade the overall resources configuration of a particular container:

```
# prlctl set MyCT --applyconfig basic
```

This command applies all the parameters from the `ve-basic.conf-sample` file to the container `MyCT`. When you install Virtuozzo Hybrid Server on your hardware node, the default container samples are put to the `/etc/vz/conf` directory. They have the following format: `ve-<name>.conf-sample` (for example, `ve-basic.conf-sample`).

- Using specific utilities for preparing configuration files in their entirety. The tasks these utilities perform are described in the following subsections of this section.
- The direct creating and editing of the corresponding container configuration file (`/etc/vz/conf/<UUID>.conf`). This can be performed with the help of any text editor. The instructions on how to edit container configuration files directly are provided in the four preceding sections. In this case you have to edit all the configuration parameters separately, one by one.

3.8.1 Splitting Server into Equal Pieces

Using the `vzsplit` command, you can create configurations for containers that would take a specific fraction of the hardware node resources. For example, to create a configuration `myconf` for up to 20 containers:

```
# vzsplit -n 20 -f myconf
Config /etc/vz/conf/ve-myconf.conf-sample was created
```

The configuration is calculated based on the hardware node resources. You can now use the `--config myconf` option of the `prlctl create` command to create containers based on this configuration.

3.8.2 Applying New Configuration Samples to Containers

Virtuozzo Hybrid Server allows you to change the configuration sample file a container is based on and, thus, to modify all the resources the container may consume and/or allocate at once. For example, if a container `MyCT` is based on the `basic` sample and you plan to run Plesk inside it, you may apply the `vswap.plesk` sample

to it instead. This will automatically adjust the container resource parameters for running Plesk. To do this, you can execute the following command on the hardware node:

```
# prlctl set MyCT --applyconfig vswap.plesk
```

This command reads the resource parameters from the `ve-vswap.plesk.conf-sample` file located in the `/etc/vz/conf` directory and applies them one by one to the container `MyCT`.

When applying new configuration samples to containers, keep in mind the following:

- All container sample files are located in the `/etc/vz/conf` directory on the hardware node and are named according to the following pattern: `ve-<name>.conf-sample`. You should specify only the `<name>` part of the corresponding sample name after the `--applyconfig` option (`vswap.plesk` in the example above).
- The `--applyconfig` option applies all the parameters from the specified sample file to the given container, except for the `OSTEMPLATE`, `TEMPLATES`, `VE_ROOT`, `VE_PRIVATE`, `HOSTNAME`, `IP_ADDRESS`, `TEMPLATE`, `NETIF` parameters (if they exist in the sample file).

You may need to restart your container if the changes cannot be set on the fly. In this case, you will see a corresponding alert.

3.9 Managing Virtual Machine Configuration Samples

The configuration of a virtual machine is defined by its `config.pvs` configuration file. This file in XML format is automatically created when you make a new virtual machine and contains all parameters of the virtual machine: memory, CPU, disk space, and so on.

Once a virtual machine is created, you can manually configure its parameters using the `prlctl` utility. However, if you need to configure multiple parameters for several virtual machines, this may become a tedious task. To facilitate your work, you can create virtual machine samples and use them to quickly and easily change the configuration of virtual machines. You can even further simplify the configuration process by creating a virtual machine template and several sample files. In this case, you can quickly make a new virtual machine on the basis of your template and apply the desired configuration file to it.

3.9.1 Creating a Configuration Sample

Before you can start using virtual machine configuration samples, you need to create at least one configuration sample. The easiest way of doing this is to follow the steps below:

1. Create a virtual machine configuration, for example:

```
# prlctl create VmConfiguration
```

2. Set the parameters for the virtual machine configuration as you want them to be. For example, you can use the `prlctl set` command to set the required amount of memory and disk space. All your parameters are saved to the `config.pvs` file of the `VmConfiguration` virtual machine. For the list of parameters that can be applied from a configuration sample, see [Parameters Applied from Configuration Samples](#) on page 134 below.
3. Copy the `config.pvs` file to the `/etc/vz/samples` directory. If this directory does not exist, create it:

```
# mkdir /etc/vz/samples
# cp /vz/vmprivate/<VmConfiguration_UUID>/config.pvs /etc/vz/samples/configMySQL.pvs
```

The latter command copies the `config.pvs` file to the `configMySQL.pvs` file.

3.9.2 Applying Configuration Samples to Virtual Machines

Now that you have created the configuration sample, you can apply it to any of your virtual machines. You can do this using the `--applyconfig` option with the `prlctl set` command and specifying the sample name without the `.pvs` extension. For example, to apply the `configMySQL` sample to the `VM1` virtual machine, you can run this command:

```
# prlctl set VM1 --applyconfig configMySQL
```

You can apply configuration samples to stopped virtual machines only.

3.9.3 Parameters Applied from Configuration Samples

The following parameters are applied to a virtual machine from a new configuration sample:

- All memory-related parameters (both RAM and video). To view these parameters in a sample file, locate the `<Memory>` and `<Video>` elements.
- All CPU-related parameters. To view these parameters in a sample file, locate the `<Cpu>` element.

- IO and IOPS parameters. To view these parameters in a sample file, locate the <IoLimit> and <IopsLimit> elements, respectively.
- Disk space parameter. To view this parameter in a sample file, locate the <Size> element enclosed in the <Hdd> element:

```
<Hdd id=0" dyn_lists="Partition 0">
<Index>0</Index>
<Size>65536</Size>
</Hdd>
```

The virtual disk to which the value of the <Size> element is applied is defined by the index number in the <Index> element. For example, in the example above, the disk space parameter (65536 MB) is applied to the virtual disk with index number 0. If the virtual machine does not have a virtual disk with the specified index, the parameter is ignored.

3.10 Monitoring Resources

In Virtuozzo Hybrid Server, you can use the `vztop` utility to monitor system resources in real time. When executed, the utility displays information about processor, swap and memory usage, number of tasks, load average, and uptime at the top of the screen. You can change the default meters by pressing F2 or S. For example, you can run the following command on the server to view your current system resources:

```
# vztop
1 [          0.0%] Tasks: 77, 65 thr; 1 running
2 [||||      2.6%] Load average: 0.02 0.03 0.05
3 [|||||     4.6%] Uptime: 06:46:48
4 [|         0.7%]
Mem[||||||||||||||||| 344M/3.68G]
Swp[          0K/3.87G]
```

The numbers on the left represent the number of CPUs/cores in the system. The progress bar shows their load and can be comprised of different colors. By default, the CPU progress bar is displayed in four colors:

- Blue: low priority processes
- Green: normal priority (user) processes
- Red: kernel processes
- Cyan: virtualization time

The memory progress bar is comprised of three colors:

- green: used memory pages
- blue: buffer pages
- yellow/orange: cache pages

The swap progress bar include only one color, red, which denotes used swap space.

The command output is updated in intervals set with the `-d` option in tenths of a second. If the `-d` option is omitted, the default interval is 1 second (i.e. `-d 10`).

CHAPTER 4

Managing Services and Processes

This chapter provides information on what services and processes are, how they influence the operation and performance of your system, and what tasks they perform in the system.

You will learn how to use the command line utilities in order to manage services and processes in Virtuozzo Hybrid Server. In particular, you will learn how to monitor active processes in your system, change the mode of the `xinetd`-dependent services, identify the container UUID where a process is running by the process ID, start, stop, or restart services and processes, and edit the service run levels.

4.1 What Are Services and Processes

Instances of any programs currently running in the system are referred to as processes. A process can be regarded as the virtual address space and the control information necessary for the execution of a program. A typical example of a process is the `vi` application running on your server or inside your Linux-based containers. Along with common processes, there are a great number of processes that provide an interface for other processes to call. They are called services. In many cases, services act as the brains behind many crucial system processes. They typically spend most of their time waiting for an event to occur or for a period when they are scheduled to perform some task. Many services provide the possibility for other servers on the network to connect to the given one via various network protocols. For example, the `nfs` service provides the NFS server functionality allowing file sharing in TCP/IP networks.

You may also come across the term “daemon” that is widely used in connection with processes and services. This term refers to a software program used for performing a specific function on the server system and is usually used as a synonym for “service”. It can be easily identified by `d` at the end of its name. For example,

`httpd` (HTTP daemon) represents a program that runs in the background of your system and waits for incoming requests to a web server. The daemon answers the requests automatically and serves the hypertext and multimedia documents over the Internet using HTTP.

When working with services, you should keep in mind the following. During the lifetime of a service, it uses many system resources. It uses the CPUs in the system to run its instructions and the system's physical memory to hold itself and its data. It opens and uses files within the file systems and may directly or indirectly use certain physical devices in the system. Therefore, in order not to decrease your system performance, you should run only those services on the hardware node that are really needed at the moment.

Besides, you should always remember that running services in the Host OS is much more dangerous than running them in virtual machines and containers. In case violators get access to one of the virtual machines and containers through any running service, they will be able to damage only the virtual machine or container where this service is running, but not the other virtual machines and containers on your server. The hardware node itself will also remain unhurt. And if the service were running on the hardware node, it would damage both the server and all virtual machines and containers residing on it. Thus, you should make sure that you run only those services on the server that are really necessary for its proper functioning. Launch all additional services you need at the moment inside separate virtual machines and containers. It can significantly improve your system safety.

4.2 Main Operations on Services and Processes

The ability to monitor and control processes and services in your system is essential because of the profound influence they have on the operation and performance of your whole system. The more you know about what each process or service is up to, the easier it will be to pinpoint and solve problems when they creep in.

The most common tasks associated with managing services running on the hardware node or inside a virtual machine or container are starting, stopping, enabling, and disabling a service. For example, you might need to start a service in order to use certain server-based applications, or you might need to stop or pause a service in order to perform testing or to troubleshoot a problem.

For `xinetd`-dependent services, you do not start and stop but enable and disable services. The services enabled in this way are started and stopped on the basis of the corresponding state of the `xinetd` daemon. Disabled services are not started whatever the `xinetd` state.

In Virtuozzo Hybrid Server, you can manage services on the hardware node and inside containers by means

of special Linux command-line utilities. You can do it either locally or from any server connected on the network.

As for processes, such Virtuozzo Hybrid Server utilities as `vzps`, `vztop`, `vzpid` enable you to see what a process is doing and to control it. Sometimes, your system may experience problems such as slowness or instability, and using these utilities can help you improve your ability to track down the causes. It goes without saying that in Virtuozzo Hybrid Server you can perform all those operations on processes you can do in a normal system, for example, kill a process by sending a terminate signal to it.

4.3 Managing Processes and Services

In Virtuozzo Hybrid Server, services and processes can be managed using the following command-line utilities:

- `vzps`
- `vzpid`
- `vztop`

With their help, you can perform the following tasks:

- Print the information about active processes on your hardware node.
- View the processes activity in real time.
- Change the mode of the services that can be either `xinetd`-dependent or standalone.
- Identify the container UUID where a process is running by the process ID.

Note: The maximum number of processes per container is limited to 131,072.

4.3.1 Viewing Active Processes and Services

The `vzps` utility provides certain additional functionality related to monitoring separate containers running on the hardware node. For example, you can use the `-E` switch with the `vzps` utility to:

- Display the container UUIDs where the processes are running.

- View the processes running inside a particular container.

`vzps` prints the information about active processes on your hardware node. When run without any options, `vzps` lists only those processes that are running on the current terminal. Below is an example output of `vzps`:

```
# vzps
PID TTY      TIME CMD
4684 pts/1    00:00:00 bash
27107 pts/1    00:00:00 vzps
```

Currently, the only processes assigned to the user/terminal are the bash shell and the `vzps` command itself. In the output, the PID (Process ID), TTY, TIME, and CMD fields are contained. TTY denotes which terminal the process is running on, TIME shows how much CPU time the process has used, and CMD is the name of the command that started the process.

Note: The IDs of the processes running inside containers and displayed by running the `vzps` command on the hardware node does not coincide with the IDs of the same processes shown by running the `ps` command inside these containers.

As you can see, the standard `vzps` command just lists the basics. To get more details about the processes running on your server, you will need to pass some command line arguments to `vzps`. For example, using the `aux` arguments with this command displays processes started by other users (a), processes with no terminal or one different from yours (x), the user who started the process and when it began (u).

```
# vzps aux
USER  PID %CPU %MEM  VSZ  RSS TTY   STAT  START  TIME  COMMAND
root   1  0.0  0.0  1516  128 ?    S     Jul14  0:37  init
root   5  0.0  0.0    0    0 ?    S     Jul14  0:03  [ubstatd]
root   6  0.0  0.0    0    0 ?    S     Jul14  3:20  [kswapd]
#27   7  0.0  0.0    0    0 ?    S     Jul14  0:00  [bdflush]
root   9  0.0  0.0    0    0 ?    S     Jul14  0:00  [kinoded]
root  1574 0.0  0.1   218  140 pts/4 S     09:30  0:00  -bash
```

There is a lot more information now. The fields USER, %CPU, %MEM, VSZ, RSS, STAT, and START have been added. Let us take a quick look at what they tell us.

The USER field shows you which user initiated the command. Many processes begin at system start time and often list root or some system account as the user. Other processes are, of course, run by actual users.

The %CPU, %MEM, VSZ, and RSS fields all deal with system resources. First, you can see what percentage of the CPU the process is currently utilizing. Along with CPU utilization, you can see the current memory utilization and its VSZ (virtual memory size) and RSS (resident set size). VSZ is the amount of memory the

program would take up if it were all in memory. RSS is the actual amount currently in memory. Knowing how much a process is currently eating will help determine if it is acting normally or has spun out of control.

You will notice a question mark in most of the TTY fields in the `vmstat` output. This is because most of these programs were started at boot time and/or by initialization scripts. The controlling terminal does not exist for these processes; thus, the question mark. On the other hand, the `bash` command has a TTY value of `pts/4`. This is a command being run from a remote connection and has a terminal associated with it. This information is helpful for you when you have more than one connection open to the machine and want to determine which window a command is running in.

STAT shows the current status of a process. In our example, many are sleeping, indicated by an `S` in the STAT field. This simply means that they are waiting for something. It could be user input or the availability of system resources. The other most common status is `R`, meaning that it is currently running.

You can also use the `vmstat` command to view the processes inside any running container. The example below shows you how to display all active processes inside the container `MyCT` with UUID

26bc47f6-353f-444b-bc35-b634a88dbbcc:

```
# vmstat -E 26bc47f6-353f-444b-bc35-b634a88dbbcc
          CTID      PID TTY          TIME CMD
26bc47f6-353f-444b-bc35-b634a88dbbcc 14663 ?          00:00:00 init
26bc47f6-353f-444b-bc35-b634a88dbbcc 14675 ?          00:00:00 kthreadd/26bc47
26bc47f6-353f-444b-bc35-b634a88dbbcc 14676 ?          00:00:00 khelper
26bc47f6-353f-444b-bc35-b634a88dbbcc 14797 ?          00:00:00 udevd
26bc47f6-353f-444b-bc35-b634a88dbbcc 15048 ?          00:00:00 rsyslogd
26bc47f6-353f-444b-bc35-b634a88dbbcc 15080 ?          00:00:00 sshd
26bc47f6-353f-444b-bc35-b634a88dbbcc 15088 ?          00:00:00 xinetd
26bc47f6-353f-444b-bc35-b634a88dbbcc 15097 ?          00:00:00 saslauthd
26bc47f6-353f-444b-bc35-b634a88dbbcc 15098 ?          00:00:00 saslauthd
26bc47f6-353f-444b-bc35-b634a88dbbcc 15116 ?          00:00:00 sendmail
26bc47f6-353f-444b-bc35-b634a88dbbcc 15125 ?          00:00:00 sendmail
26bc47f6-353f-444b-bc35-b634a88dbbcc 15134 ?          00:00:00 httpd
26bc47f6-353f-444b-bc35-b634a88dbbcc 15139 ?          00:00:00 httpd
26bc47f6-353f-444b-bc35-b634a88dbbcc 15144 ?          00:00:00 crond
26bc47f6-353f-444b-bc35-b634a88dbbcc 15151 ?          00:00:00 mingetty
26bc47f6-353f-444b-bc35-b634a88dbbcc 15152 ?          00:00:00 mingetty
```

4.3.2 Monitoring Processes in Real Time

The `vmstat` utility is rather similar to `vmstat` but is usually started full-screen and updates continuously with process information. This can help with programs that may infrequently cause problems and can be hard to see with `vmstat`. Overall system information is also presented, which makes a nice place to start looking for problems.

The `vtop` utility can be used just as the standard Linux `htop` utility. It shows a dynamic list of all processes running on the system with their full command lines.

By default, it shows information about processor, swap and memory usage, number of tasks, load average, and uptime at the top of the screen. You can change the default meters, along with display options, color schemes, and columns at the setup screen (S or F2).

`vtop` can be used interactively for sending signals to processes. For example, you can kill processes—without knowing their PIDs—by selecting them and pressing F9. You can also change process priority by pressing F7 (increase; can only be done by the root user) and F8 (decrease).

The `vtop` utility usually has an output like the following:

```
# vtop
1 [          0.0%] Tasks: 77, 65 thr; 1 running
2 [|||      2.6%] Load average: 0.02 0.03 0.05
3 [||||     4.6%] Uptime: 06:46:48
4 [|        0.7%]
Mem[|||||||||||||||||] 344M/3.68G
Swp[         0K/3.87G]

PID CTID USER  PRI NI VIRT  RES  SHR S CPU% MEM%  TIME+  Command
1     0  root   20  0 41620 4132 2368 S 0.0 0.1 0:05.91 /usr/lib/systemd/systemd
3164 0  root   20  0 19980 1380 1160 S 0.0 0.0 0:00.32 /usr/lib/systemd/systemd-
3163 0  root   21  1 1402M 56992 10204 S 0.0 1.5 4:12.41 /usr/libexec/qemu-kvm -na
3186 0  root   20  0 1402M 56992 10204 S 0.0 1.5 0:00.09 /usr/libexec/qemu-kvm -na
3185 0  root   20  0 1402M 56992 10204 S 0.7 1.5 2:16.83 /usr/libexec/qemu-kvm -na
3180 0  root   20  0 1402M 56992 10204 S 0.0 1.5 0:00.00 /usr/libexec/qemu-kvm -na
3084 0  smmsp 20  0 85712 2036 516 S 0.0 0.1 0:00.19 sendmail: Queue runner@01
3064 0  root   20  0 98M 2380 572 S 0.0 0.1 0:01.43 sendmail: accepting conne
3036 0  root   20  0 291M 4788 3580 S 0.0 0.1 0:00.00 /usr/sbin/virtlogd
3037 0  root   20  0 291M 4788 3580 S 0.0 0.1 0:00.00 /usr/sbin/virtlogd
2787 0  nobody20 0 15548 896 704 S 0.0 0.0 0:00.14 /sbin/dnsmasq --conf-file
2788 0  root   20  0 15520 184 0 S 0.0 0.0 0:00.00 /sbin/dnsmasq --conf-file
2479 0  root   20  0 1962M 33344 24160 S 0.7 0.9 3:13.12 /usr/sbin/pr1_disp_servic
9022 0  root   20  0 1962M 33344 24160 S 0.0 0.9 0:10.74 /usr/sbin/pr1_disp_servic
```

The column `CTID` shows the container UUID inside which the process is running (0 means that it is running on the server). `PRI` (PRIORITY) displays the kernel's internal priority for the process. `NI` (NICE) shows niceness (the nicer the process, the more it lets other processes take priority).

To organize processes by parenthood, you can switch to the tree view by pressing F5.

4.3.3 Determining Container UUIDs by Process IDs

Each process is identified by a unique PID (process identifier), which is the entry of that process in the kernel's process table. For example, when you start Apache, it is assigned a process ID. This PID is then used to monitor and control this program. The PID is always a positive integer. In Virtuozzo Hybrid Server, you can use the `vzpid` (retrieve process ID) utility to print the container UUID the process with the given id belongs to. Multiple process IDs can be specified as arguments. In this case the utility will print the container number for each of the processes.

The typical output of the `vzpid` utility is shown below:

```
# vzpid 14663
Pid          VEID      Name
14663       26bc47f6-...  init
```

Note: You can also display the container UUID where the corresponding process is running by using the `vzps` utility.

CHAPTER 5

Managing Network

This chapter familiarizes you with the Virtuozzo Hybrid Server network structure, lists networking components, and explains how to manage these components in your working environments. In particular, it provides the following information:

- How you can manage network adapters on the hardware node.
- What virtual networks are and how you can manage them on the hardware node.
- How to create virtual network adapters inside your virtual machines and containers and configure their parameters.
- How to connect virtual machines and containers to different networks.

5.1 Managing Network Adapters on the Hardware Node

Network adapters installed on the hardware node are used to provide virtual machines and containers with access to each other and to external networks. During the installation, Virtuozzo Hybrid Server registers all physical network adapters available on the server. Once Virtuozzo Hybrid Server has been successfully installed, you can manage network adapters on the hardware node using native RHEL7 utilities. You can also create bond and VLAN interfaces (for example, with the `nmtui` tool) and use them instead of physical ones.

Important:

1. Each network adapter must have only one configuration file in the `/etc/sysconfig/network-scripts/` directory.

2. Do not change host network settings while virtual environments are running. Otherwise you may need to restart the VEs to propagate the new network settings to them.
 3. NetworkManager does not manage virtual network adapters of containers and virtual machines, i.e. veth and vme devices.
-

5.2 Networking Modes in Virtuozzo Hybrid Server

This section describes networking modes available in Virtuozzo Hybrid Server.

In Virtuozzo Hybrid Server, any virtual machine or container can operate in one of the two networking modes: host-routed or bridged.

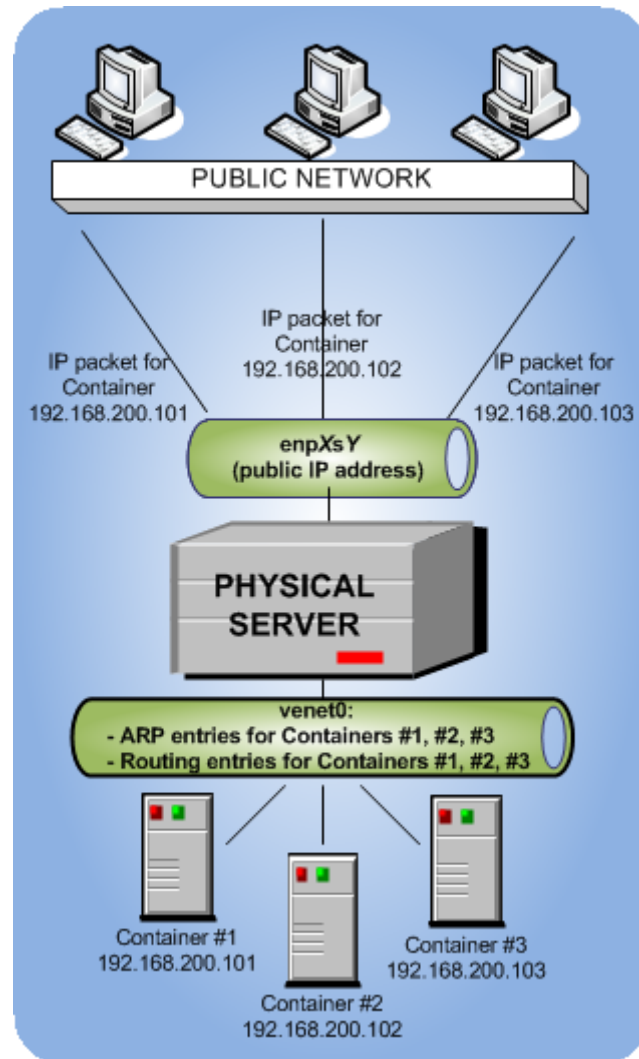
5.2.1 Container Network Modes

This section describes bridged and host-routed network modes for containers.

Note: IPSec connections inside containers are supported.

5.2.1.1 Host-Routed Mode for Containers

By default, a new container starts operating in the host-routed mode. In this mode, the container uses a special network adapter, `venet0`, to communicate with the server where it resides, with the other containers on the server, and with computers on external networks. The figure below demonstrates an example network configuration where all containers are set to work in the host-routed mode.



In this configuration:

- Containers #1, #2, and #3 use the `venet0` adapter as the default gateway to send and receive data to/from other networks. They also use this adapter to exchange the traffic between themselves.
- When containers #1, #2, and #3 start, the server creates ARP and routing entries for them in its ARP and routing tables. You can view the current ARP and routing entries on a server using the `arp -n` and `route -n` commands. For example:

```
# arp -n
Address          HWtype  HWaddress      Flags Mask    Iface
10.30.0.4        ether   00:1a:e2:c7:17:c1  C           enp0s5
10.30.23.162     ether   70:71:bc:42:f6:a0  C           enp0s5
192.168.200.101 *       *           MP            enp0s5
192.168.200.102 *       *           MP            enp0s5
192.168.200.103 *       *           MP            enp0s5
```



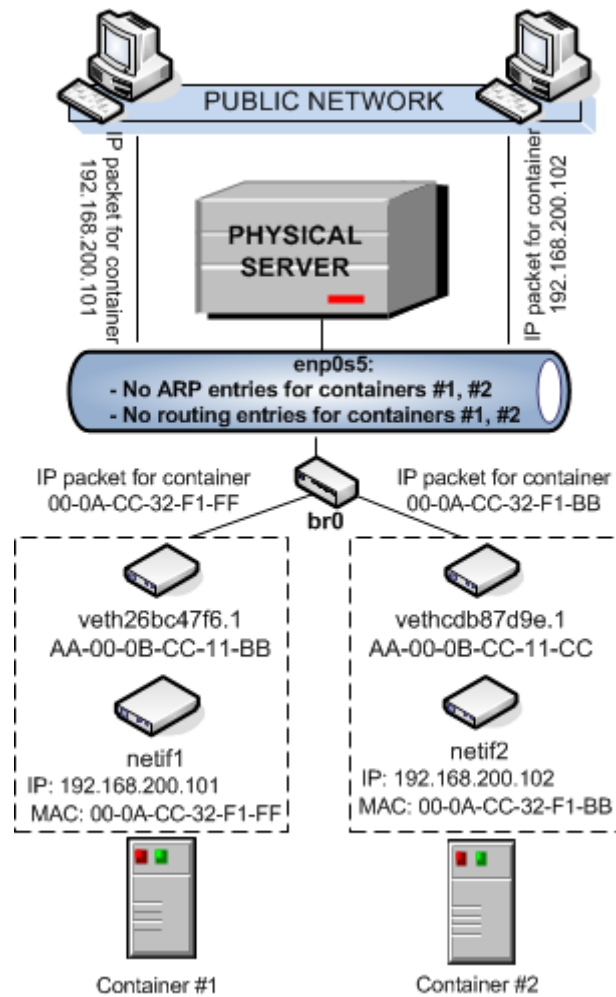
```
# route -n
Kernel IP routing table
Destination      Gateway          Genmask         Flags Metric Ref    Use Iface
192.168.200.101  *               255.255.255.255 UH    1000  0      0 venet0
192.168.200.102  *               255.255.255.255 UH    1000  0      0 venet0
192.168.200.103  *               255.255.255.255 UH    1000  0      0 venet0
10.30.0.0         *               255.255.0.0    U      0      0      0 enp0s5
default          virtuozzo.com   0.0.0.0         UG     0      0      0 enp0s5
```

As you can see, the ARP and routing tables contain entries about IP addresses 192.168.200.101, 192.168.200.102, and 192.168.200.103 that belong to containers #1, #2 and #3.

- All container outgoing network traffic goes to the `venet0` adapter and is forwarded via the `enp0s5` physical adapter to the destination, according to the routing table of the server.
- All container incoming network traffic is also processed by the `venet0` adapter. Consider the following situation:
 1. Computer X on the local network wants to send a data packet to container #1 with IP address 192.168.200.101, so it issues an ARP request which computer has this IP address.
 2. The server hosting container #1 replies with its MAC address.
 3. Computer X sends the data packet to the indicated MAC address.
 4. The server receives the packet and transmits it to `venet0` that forwards the packet to container #1.

5.2.1.2 Bridged Mode for Containers

The default network adapter of a container can operate in the host-routed mode only. You can, however, create additional virtual adapters in containers and make them operate in the bridged network mode. The following figure shows an example network configuration where containers #1 and #2 are set to work in the bridged mode.



In this configuration:

- Container #1 and container #2 have separate virtual adapters consisting of two network interfaces:
 - A `netif<X>` interface in the container (`netif1` and `netif2` in the figure). This interface represents a counterpart of a physical network adapter installed on a standalone server. Like any other physical adapter, it has a MAC address, can be assigned one or more IP addresses, included in different networks, and so on.
 - A veth interface on the hardware node (`veth26bc47f6.1` and `vethcdb87d9e.1` in the figure). This interface is mostly used to maintain the communication between the hardware node and Ethernet interfaces in containers.

Note: To simplify things, virtual adapters operating in the bridged mode are called veth adapters,

though it is not quite correct from the technical point of view.

Both interfaces are closely linked to each other, so a data packet entering one interface always comes out from the other one.

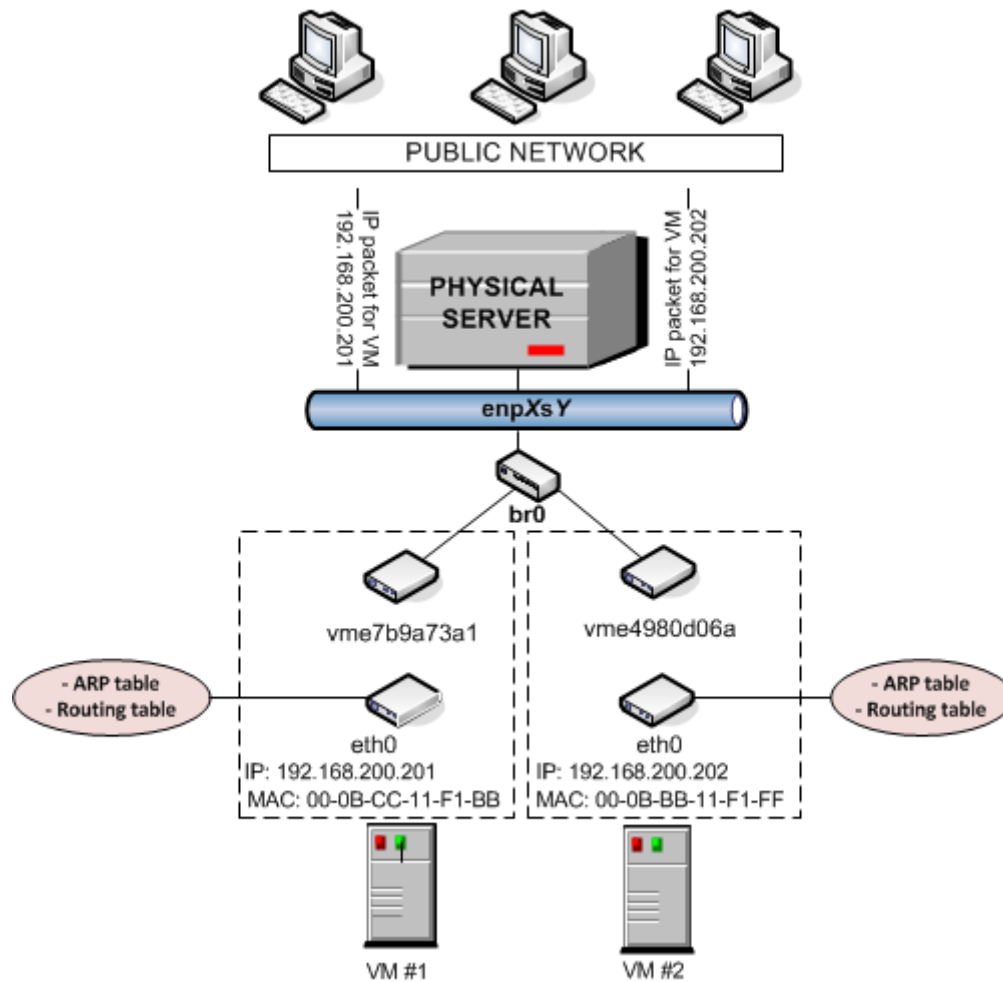
- Containers #1 and #2 keep their own ARP and routing tables that they consult when sending or receiving data.
- The veth adapters of both containers are bridged through the bridge `br0` to the physical network adapter `enp0s5`.
- All container outgoing traffic comes via the veth adapters to the bridge and are then transmitted through the `enp0s5` physical adapter to the destination, according to the routing tables stored in the containers.
- All incoming data packets for container #1 and #2 reach the `enp0s5` physical adapter first and are then sent through the bridge to the veth adapter of the destination container.

5.2.2 Virtual Machine Network Modes

This section describes bridged and host-routed network modes for virtual machines.

5.2.2.1 Bridged Mode for Virtual Machines

By default, a new virtual machine is created with a network adapter that operates in the bridged mode. The figure below demonstrates an example network configuration where two virtual machines, VM #1 and VM #2, are configured to work in the bridged mode.



In this configuration:

- Each virtual machine has a separate virtual adapter that exposes two interfaces: (1) an ethX interface in the virtual machine (eth0 in the figure) and a vme interface on the server (vme7b9a73a1 and vme4980d06a in the figure). Both interfaces are closely linked to each other, which means that an IP packet entering one interface always comes out of the other one. An eth adapter has a MAC address, can be assigned one or more IP addresses, belong to different network environments, and so on.

Note: To simplify things, virtual adapters operating in the bridged mode are called vme adapters, though it is not quite correct from the technical point of view.

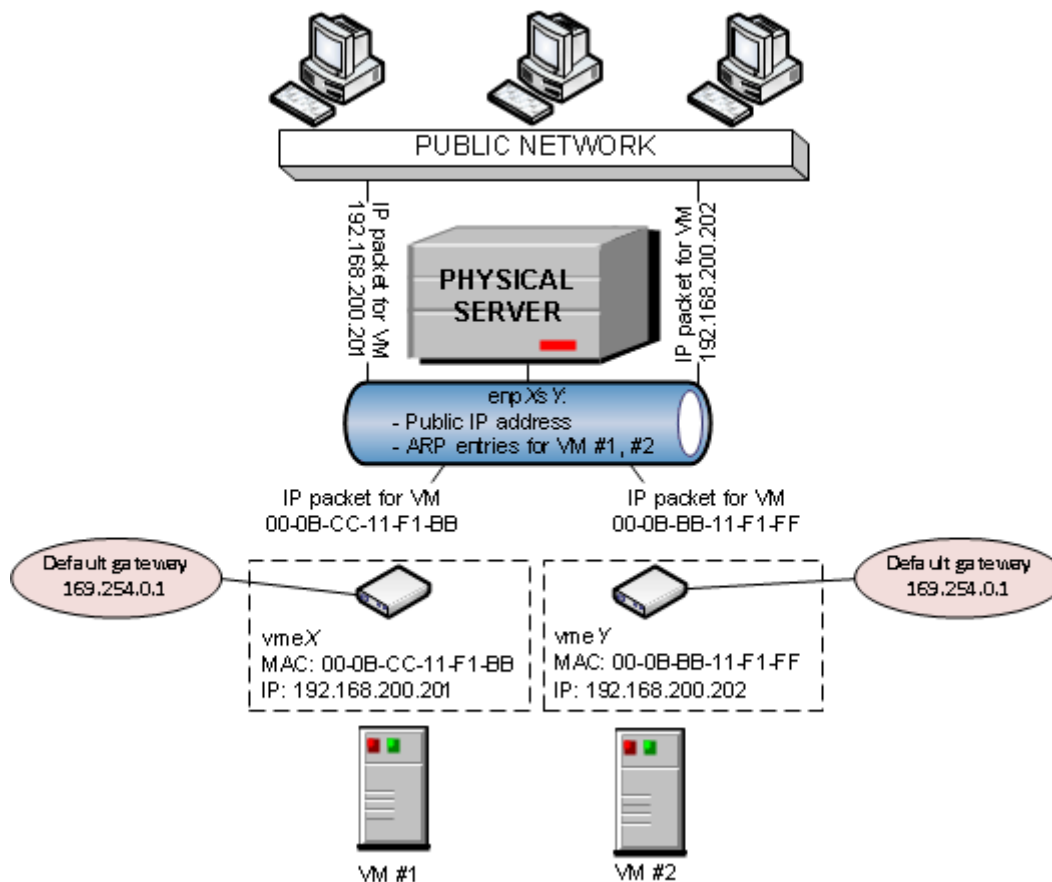
- VM #1 and VM #2 keep their own ARP and routing tables that they consult when sending or receiving data.
- The virtual adapters of both virtual machines are bridged through the bridge br0 to the physical

network adapter `enp0s5`.

- All outgoing data packets are sent from the virtual machines through the bridge and `enp0s5` physical adapter to the destination, according to their routing tables.
- All incoming data packets for VM #1 and VM #2 reach the `enp0s5` physical adapter first and are then transmitted through the bridge to the `vme` interface of the destination virtual machine.

5.2.2.2 Host-Routed Mode for Virtual Machines

The other network mode a virtual machine can work in is the host-routed mode. The figure below demonstrates an example network configuration where two virtual machines, VM #1 and VM #2, are set to operate in the host-routed mode.



In this configuration:

- Each virtual machine also has a virtual adapter exposing two interfaces: an `eth` interface in the virtual machine and a `vme` interface on the server.

- Unlike the bridged mode, the ARP entries for VM #1 and VM #2 are stored on the server rather than in the virtual machines themselves. The server creates these ARP entries and saves them to its ARP table when VM #1 and VM #2 start. You can use the `arp -n` command to view the current ARP entries on a server, for example:

```
# arp -n
Address          HWtype HWaddress      Flags Mask    Iface
10.30.0.4        ether  00:1a:e2:c7:17:c1  C           eth0
10.30.23.162     ether  70:71:bc:42:f6:a0  C           eth0
192.168.200.201 *        *           MP             eth0
192.168.200.202 *        *           MP             eth0
```

- Along with ARP entries, the server also creates routing entries for both virtual machines. So when the server receives a data packet destined for IP address 192.168.200.201, it knows that the packet must be forwarded to the `vme7b9a73a1` interface of VM #1.
- The server handles all incoming traffic for both virtual machines. Consider the following situation:
 1. Computer X on the network wants to send a data packet to VM #1 with IP address 192.168.200.201, so it issues an ARP request which computer has this IP address.
 2. The server replies with its own MAC address.
 3. Computer X sends the data packet to the indicated MAC address.
 4. The `enp0s5` physical adapter receives the packet and routes it to the `vme7b9a73a1` interface of VM #1.
- All outgoing network traffic sent from VM #1 and VM #2 are routed through the default gateway to the `enp0s5` adapter on the server. The default gateway for host-routed virtual machines is automatically assigned the IP address of 169.254.0.1. This special IP address is taken from the Automatic Private IP Addressing (APIPA) range and used exclusively to deliver data packets from virtual machines to the server.

5.2.3 Differences Between Host-Routed and Bridged Network Modes

The bridged network mode demonstrates a number of differences as compared to the host-routed one:

- Each `veth` or `vme` virtual adapter has a MAC address assigned to it while a host-routed adapter does not have any. Thanks to this fact:
 - Any virtual machine or container can see all broadcast and multicast packets received from or sent to the selected network adapter on the hardware node.

- Using bridged virtual adapters, you can host DHCP or Samba servers in virtual machines and containers.
- There is no more need to assign all network settings (IP addresses, subnet mask, gateway, and so on) to virtual machines and containers from the server. All network parameters can be set from inside virtual machines and containers.
- veth and vme adapters can be bridged among themselves and with other devices. If several veth and vme adapters are united into a bridge, this bridge can be used to handle network traffic for the virtual machines and containers whose veth and vme adapters are included in the bridge.
- Due to the fact that veth and vme adapters act as full members on the network (rather than “hidden” beyond virtual networks adapters on the server), they are more prone to security vulnerabilities: traffic sniffing, IP address collisions, and so on. Therefore, veth and vme adapters are recommended for use in trusted network environments only.

5.3 Configuring Virtual Machines and Containers in Host-Routed Mode

You can configure the following parameters of network adapters that operate in the host-routed mode:

- IP addresses and network masks
- DNS servers
- DNS search domains

5.3.1 Setting IP Addresses

The session below shows how to set IP addresses for the virtual machine MyVM and the container MyCT

```
# prlctl set MyVM --device-set net0 --ipadd 10.0.186.100/24
# prlctl set MyVM --device-set net0 --ipadd 1fe80::20c:29ff:fe01:fb07
# prlctl set MyCT --ipadd 10.0.186.101/24
# prlctl set MyCT --ipadd fe80::20c:29ff:fe01:fb08
```

net0 in the commands above denotes the network card in the virtual machine MyVM to assign the IP address to. You can view all network cards of a virtual machine using the `prlctl list VM_name -i` command. For the container MyCT, you do not need to specify the network card name; `prlctl set` automatically performs the

operation on the default adapter that always operates in the host-routed mode.

5.3.2 Setting DNS Server Addresses

To set a DNS server for the virtual machine MyVM and the container MyCT, you can use the following commands:

```
# prlctl set MyVM --device-set net0 --nameserver 192.168.1.165
# prlctl set MyCT --nameserver 192.168.1.165
```

5.3.3 Setting DNS Search Domains

To set a DNS search domain for the virtual machine MyVM and the container MyCT, run these commands:

```
# prlctl set MyVM --device-set net0 --searchdomain 192.168.10.10
# prlctl set MyCT --searchdomain 192.168.10.10
```

Note the following:

- You can only configure network settings of virtual machines that have Virtuozzo guest tools installed.
- Network adapters operating in the routed mode must have at least one static IP address assigned.
- To be able to assign network masks to containers operating in the `venet0` networking mode, set the `USE_VENET_MASK` parameter in the `/etc/vz/vz.conf` configuration file to `yes`.
- Containers can only have one network adapter operating in the host-routed mode. This adapter is automatically created when you create a container.
- You can set name servers and search domain in `/etc/vz/vz.conf` with the `NAMESERVER` and `SEARCHDOMAIN` parameters. If set to `inherit`, the values will be copied from `/etc/resolv.conf` on the host.

5.3.3.1 Switching Virtual Machine Adapters to Host-Routed Mode

By default, a virtual adapter in any newly created virtual machine starts operating in the bridged mode (see [Connecting Virtual Environments to Virtual Networks](#) on page 165 for details). To change the current network mode to host-routed, you can run the following command:

```
# prlctl set <VM_name> --device-set Net_ID --type routed
```

For example, to set the `net0` adapter in the virtual machine MyVM to operate in the host-routed mode, use this command:


```
# prlctl set MyVM --device-set net0 --type routed
```

5.4 Configuring Virtual Machines and Containers in Bridged Mode

Important: To create and manage virtual networks and virtual network bridges, you must use either the command-line interface or Virtuozzo Automator, but not both.

This section describes all operations related to configuring virtual machines and containers that operate in bridged mode.

5.4.1 Managing Virtual Networks

A virtual network acts as a binding interface between a virtual network adapter in a virtual machine or container and the corresponding network adapter on the hardware node. Using virtual networks, you can include virtual machines and containers in different networks. Virtuozzo Hybrid Server enables you to manage virtual networks as follows:

- Create virtual networks.
- Configure virtual network parameters.
- List existing virtual networks.
- Delete virtual networks.

These operations are described in the following subsections in detail.

5.4.1.1 Creating Virtual Networks

By default, Virtuozzo Hybrid Server creates the following virtual networks on the server:

- Bridged virtual network that is connected to one of the physical adapters on the hardware node (e.g., `enp0s5`) and provides virtual machines and containers included in this virtual network with access to the network behind this physical adapter.

- Host-only virtual network that is connected to a special virtual adapter on the server and allows the virtual machines and containers joined to this virtual network to access only the server and the other virtual machines and containers on this network.

You can create your own virtual networks using the `prlsrvctl` command. For example, to create a new virtual network `network1`, you can run:

```
# prlsrvctl net add network1
```

By default, the command creates a host-only virtual network, but you can change its type if needed (see *Configuring Virtual Network Parameters* on page 160).

5.4.1.2 Creating Network Bridges for Network Adapters

To connect a network adapter to a bridged virtual network, you need to create a network bridge first. A network adapter can be physical (`enp<X>s<Y>`) or logical: a VLAN (`enp<X>s<Y>.<N>`) or a bonding interface (`bond<N>`).

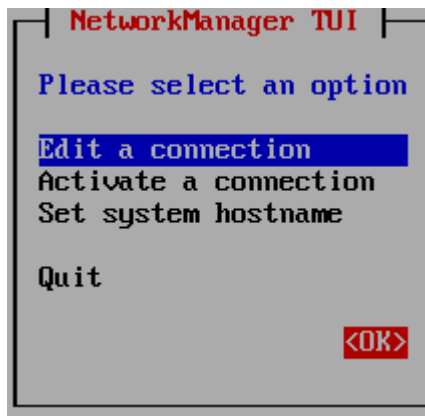
For example, to create a network bridge for the VLAN interface, you can use the NetworkManager text-based user interface tool, `nmtui`, as follows:

1. On the node, start `nmtui`:

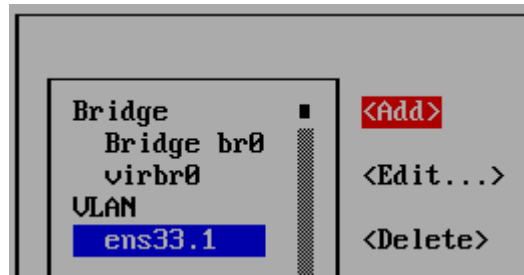
```
# nmtui
```

In the tool TUI, use the arrow keys and **Tab** to navigate through the options, **Enter** to select an option, and **Space** to set and clear check boxes.

2. On the **NetworkManager TUI** screen, select **Edit a connection** from the menu.



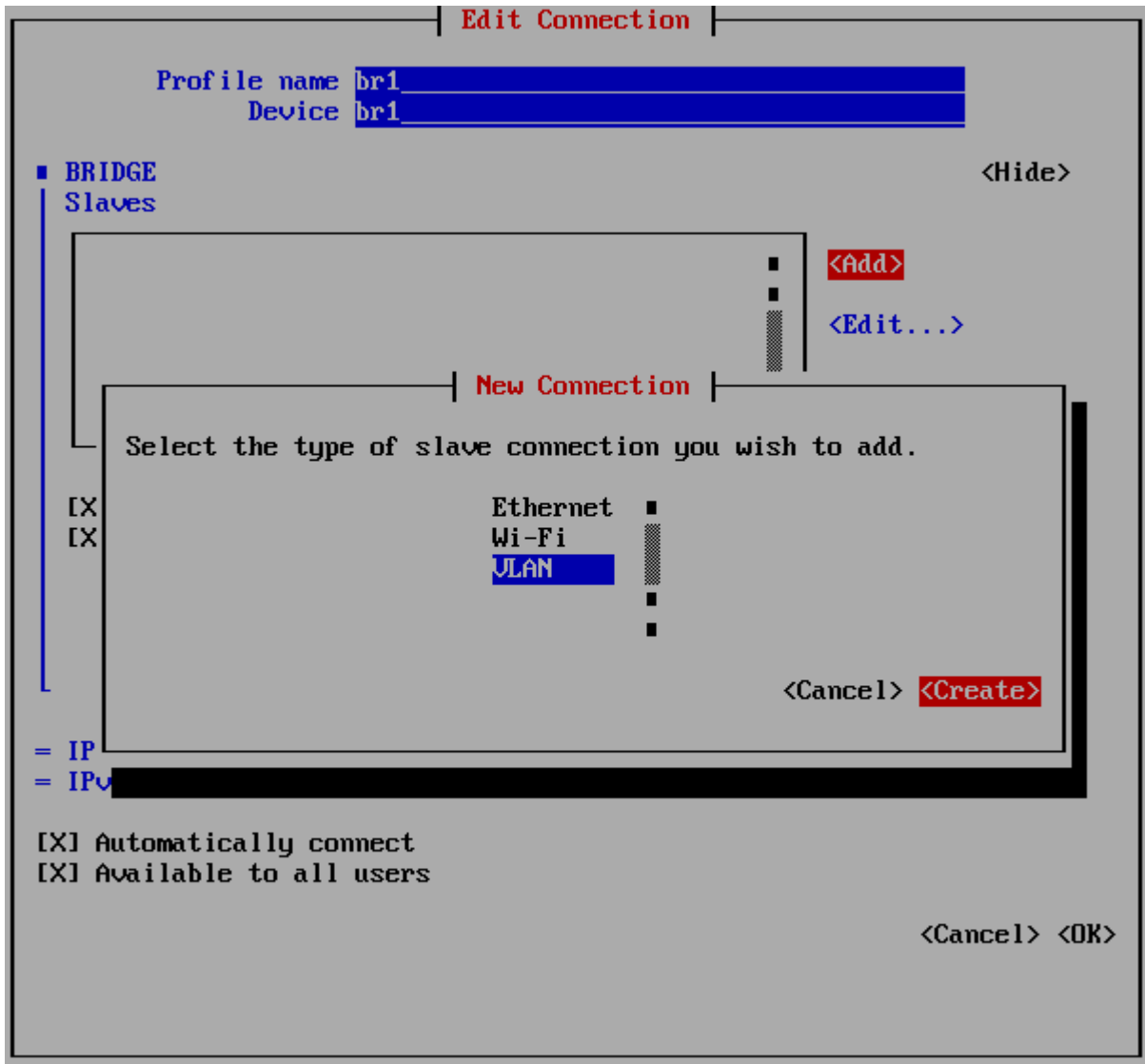
3. On the next screen, select **Add** to add a new connection.



4. To add a new network bridge, choose **Bridge** from the drop-down list in the **New connection** window and press **Create**.

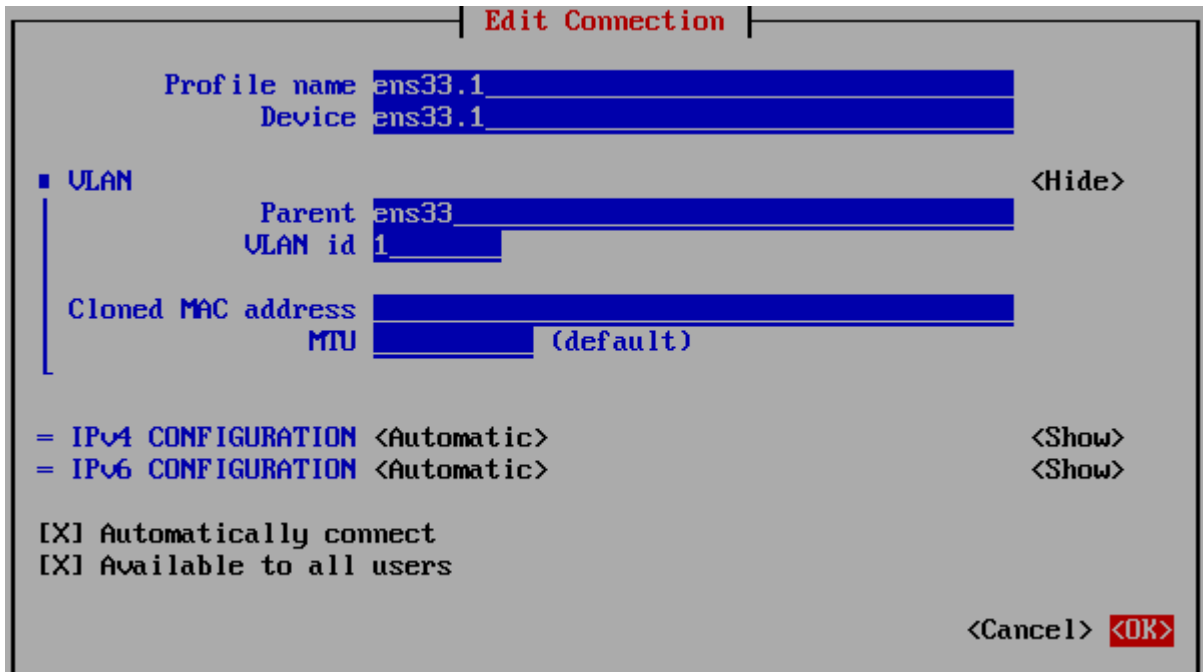


5. On the **Edit connection** screen:
- 5.1. In the **Profile name** field, enter the connection profile name. This name with the `ifcfg-` prefix will be used for creating the interface configuration file in the `/etc/sysconfig/network-scripts/` directory.
 - 5.2. In the **Device** field, specify the device name of the new network bridge.
 - 5.3. Press **Add** to specify a slave network interface.
 - 5.4. In the **New connection** window, choose **VLAN** from the drop-down list and press **Create**.



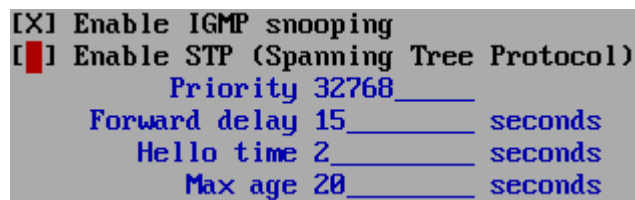
- 5.5. In the **Edit connection** window, specify the profile name and device name of the VLAN interface in the **Profile name** and **Device** fields, respectively, and press **OK**.

The **Parent** and **VLAN ID** fields are filled in automatically.



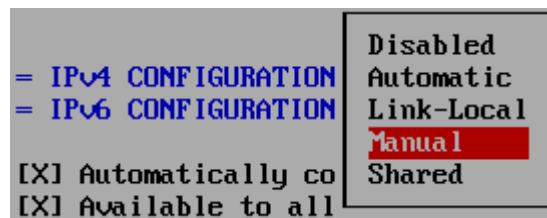
The chosen VLAN interface will appear in the **Slaves** section.

5.6. Clear the **Enable STP (Spanning Tree Protocol)** check box.



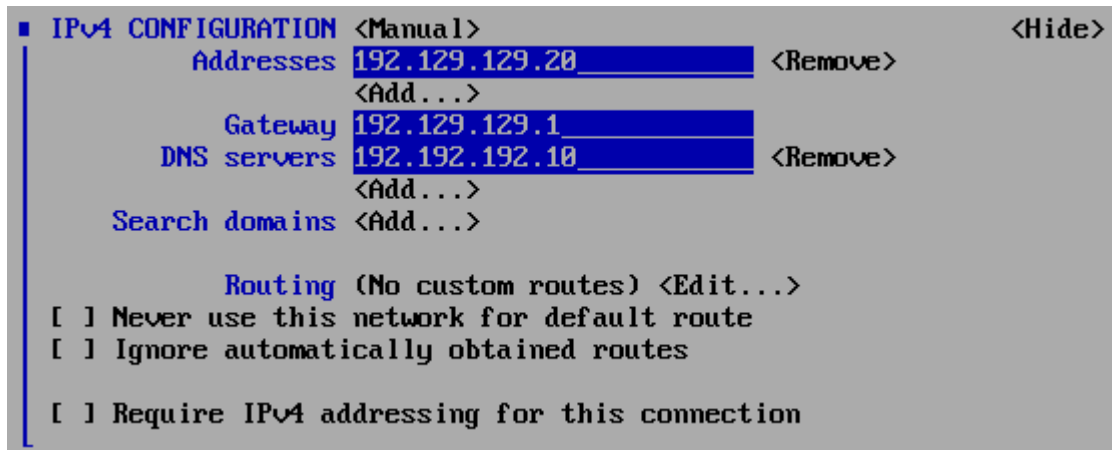
5.7. Configure static IP parameters of the network bridge:

5.7.1. In the **IPv4 CONFIGURATION** section, press **Automatic** and choose **Manual** from the drop-down list.



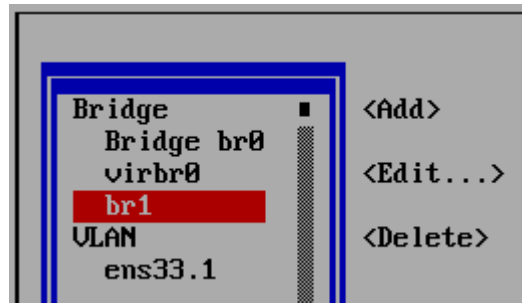
5.7.2. Press **Show** to expand the section.

5.7.3. Assign a static IP address, set the default gateway and the DNS server for the network bridge in the corresponding fields.



5.8. Configure other network parameters if required and press **OK**.

The network bridge for the VLAN adapter will appear in the list of existing connections.



6. To exit nmtui, press **Back** and then **Quit**.

After creating the network bridge, you can check its configuration file stored in the `/etc/sysconfig/network-scripts/` directory. For example:

```
# cat /etc/sysconfig/network-scripts/ifcfg-br1
```

5.4.1.3 Configuring Virtual Network Parameters

Virtuozzo Hybrid Server allows you to configure the following parameters for a virtual network:

- The networking mode in which the virtual network is operating.

Note: Before changing the virtual network type to bridged, a network bridge must be created for the virtual network. See *Creating Network Bridges for Network Adapters* on page 156.

- The description of the virtual network.

All these operations can be performed using the `prlsrvctl` utility. For example, if you need to configure the `network1` virtual network. This virtual network is currently configured as a host-only network and has the following description: This is a host-only virtual network. To change these parameters, you can execute the following command:

```
# prlsrvctl net set network1 -t bridged --ifname enp0s6 -d "This is now a bridged \
virtual network"
```

This command configured the `network1` virtual network as follows:

1. Changes the virtual network type to bridged.
2. Changes the virtual network description to the following: "This is now a bridged virtual network".

5.4.1.4 Listing Virtual Networks

To list the virtual networks existing on the hardware node, you can use the `prlsrvctl` utility as shown below.

```
# prlsrvctl net list
Network ID      Type      Bound To      Bridge
Host-Only      host-only
Bridged        bridged   enp0s5        br0
```

This utility displays the following information on virtual networks:

Column	Description
Network ID	The name assigned to the virtual network.
Type	The networking mode set for the virtual network.
Bound To	The adapter on the hardware node connected to the virtual networks, if any.

5.4.1.5 Connecting Virtual Networks to Adapters

By connecting an adapter on the physical server to a virtual network, you can join all virtual machines and containers included in the virtual network to the network to which the corresponding adapter is connected.

Consider the following example:

- The `enp0s6` physical adapter and the `network1` virtual network exist on the hardware node. For information on creating virtual networks, see *Creating Virtual Networks* on page 155.

- The `enp0s6` physical adapter is connected to the local network.
- The `br1` network bridge for the `enp0s6` physical adapter is created. For information on creating network bridges, see *Creating Network Bridges for Network Adapters* on page 156.
- The container `MyCT` is connected to the `network1` virtual network. Detailed information on joining virtual machines and containers to virtual networks is given in *Connecting Virtual Environments to Virtual Networks* on page 165.

To connect the `enp0s6` adapter to the `network1` virtual network and thus to join the container `MyCT` to the network behind `enp0s6`, run this command on the server:

```
# prlsrvctl net set network1 -i enp0s6
```

To check that the `enp0s6` physical adapter has been successfully added to the `network1` virtual network, you can execute the following command:

```
# prlsrvctl net list
Network ID      Type      Bound To      Bridge
Host-Only      host-only
Bridged        bridged   enp0s5        br0
network1       bridged   enp0s6        br1
```

As you can see, the `enp0s6` adapter is now joined to the `network1` virtual network. That means that the container `MyCT` whose virtual network adapter is connected to `network1` can access the local network behind `enp0s6`.

5.4.1.6 Deleting Virtual Networks

At any time, you can remove a virtual network that you do not need any more from the physical server. To do this, you can use the `prlsrvctl` utility. For example, you can delete the `network1` virtual network by running the following command:

```
# prlsrvctl net del network1
```

To check that `network1` has been successfully removed, execute this command:

```
# prlsrvctl net list
Network ID      Type      Bound To
Host-Only      host-only
Bridged        bridged   enp0s5
```


5.4.2 Managing Virtual Network Adapters in Virtual Environments

Virtuozzo Hybrid Server provides you with ample opportunities of configuring virtual network adapters in virtual environments and including them in different network environments. This section shows you the way to perform the following operations:

- Create new virtual network adapters and delete existing ones.
- Configure the parameters of an existing virtual network adapter.
- Join virtual network adapters to virtual networks.

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

5.4.2.1 Creating and Deleting Virtual Adapters

A virtual environment can have up to 15 virtual network adapters. Each adapter can be connected to a different network. For example, if you need to create a new virtual adapter for the virtual machine MyVM. To do this, you can execute the following command:

```
# prctl set MyVM --device-add net
```

To check that the network adapter (net1) has been successfully added to the virtual machine, run this command:

```
# prctl list --info MyVM
ID: {f3b3d134-f512-324b-b0b1-dbd642f5220b}
Name: MyVM
<...>
net1 (+) dev='vme4208fa77' network='Bridged' mac=001C4208FA77 card=virtio
```

At any time, you can remove the newly created network adapter (net1) by executing the following command:

```
# prctl set MyVM --device-del net1
```

5.4.2.2 Configuring Virtual Adapter Parameters

Virtuozzo Hybrid Server allows you to configure the following parameters of virtual adapters:

Configuring MAC Addresses

If you need for some reason to regenerate the current MAC address of a network adapter, you can use the following command:

```
# prlctl set MyVM --device-set net1 --mac 00:1C:42:2D:74:00
```

This command sets the MAC address of `00:1C:42:2D:74:00` for the `net1` adapter in the virtual machine `MyVM`. If you do not know what MAC address to assign to your virtual adapter, you can generate a new one with `prlctl set`. For example:

```
# prlctl set MyVM --device-set net1 --mac auto
```

Configuring IP Parameters

As any other standalone server, each virtual environment must have a number of TCP/IP settings configured in the proper way to successfully operate on the network. These settings include:

- IP address
- Default gateway
- DNS server

Note: You can configure the network parameters only of virtual machines that have Virtuozzo guest tools installed.

Usually, you define all these settings when you create a virtual environment. However, if you have not yet set any of the settings or want to modify any of them, you can use the `prlctl set` command. For example, you can execute the following command to assign the IP address of `192.129.129.20` to the `net1` adapter in the virtual machine `MyVM`, set the default gateway to `192.129.129.1` and the DNS server to `192.192.192.10`:

```
# prlctl set MyVM --device-set net1 --ipadd 192.129.129.20 --gw 192.129.129.1 \  
--nameserver 192.192.192.10
```

Along with a static assignment of network parameters to a virtual adapter, you can make the adapter receive its TCP/IP settings automatically using the Dynamic Host Configuration Protocol (DHCP). For example, you can run this command to make the `net1` adapter in the virtual machine `MyVM` get its IP settings through DHCP:

```
# prlctl set MyVM --device-set net1 --dhcp yes
```

5.4.2.3 Connecting Virtual Environments to Virtual Networks

In Virtuozzo Hybrid Server, you can connect virtual environments to virtual networks of the following types:

- Bridged virtual network allows a virtual environment to use one of the physical server network adapters, which makes it appear as a separate computer on the network the corresponding adapter belongs to.
- Host-only virtual network allows a virtual environment to access only the physical server and the virtual environments joined to this network.

By default, any newly created adapter is connected to the Bridged network. To join a virtual machine to another network, use the `prlctl set` command. For example, the following session demonstrates how you can connect the `net0` adapter of the virtual machine `MyVM` to the `network1` virtual network.

Before connecting the virtual machine `MyVM` to the `network1` virtual network, you may wish to check the network adapter associated with this virtual network. You can do it, for example, using the following command:

```
# prlsrvctl net list
Network ID      Type      Bound To
Host-Only      host-only
Bridged        bridged   enp0s5
network1       bridged   enp0s6
```

From the command output, you can see that the `network1` virtual network is attached to the `enp0s6` physical adapter on the physical server. That means that, after connecting the virtual machine `MyVM` to the `network1` virtual network, the virtual machine will be able to access all computers on the network where the `enp0s6` adapter is connected.

Now you can run the following command to join the `net1` adapter of the virtual machine `MyVM` to the `network1` virtual network:

```
# prlctl set MyVM --device-set net1 --network network1
```

To check that the network adapter (`net1`) has been successfully joined to the `network1` virtual network, execute

```
# prlctl list --info MyVM
ID: {f3b3d134-f512-324b-b0b1-dbd642f5220b}
Name: MyVM
<...>
net1 (+) dev='vme4208fa77' network='network1' mac=001C4208FA77 card=virtio
```

CHAPTER 6

Managing Licenses

This chapter describes how to install, update, view, and transfer licenses on your servers.

Note: If you want to use the Virtuozzo Storage functionality, you need to install a separate license in addition to a Virtuozzo Hybrid Server license. For more details, see [Managing Storage Licenses](#).

6.1 Installing the License

Virtuozzo Hybrid Server requires that a different license be installed on each server. You can activate a license during or after the installation. In the latter case, you can install a license from a product key, an activation code, or a license file.

If your Virtuozzo Hybrid Server node cannot access the Internet directly, you can activate a license via a proxy server.

6.1.1 Using a Proxy Server to Activate Licenses

In situations when a direct Internet connection is not possible, you can specify an HTTP proxy to access the Key Administration (KA) server through. To do that, add the HTTP proxy information to the `/etc/vz/vz.conf` file as follows:

```
HTTP_PROXY="http://<host>:<port>/"
HTTP_PROXY_USER="<username>"
HTTP_PROXY_PASSWORD="<password>"
```

Note the following:

- You may need to create `/etc/vz/vz.conf` first.
- The proxy server must have the port 5224 approved for SSL traffic.
- The `vzlicutils` package is required for license auto-updating to work. If the package is not installed on your system for some reason, you can install it manually with `yum`. To enable license auto-updates after installing the package, launch the `vzlicmon` service with `systemctl start vzlicmon`. The service will show as `/usr/sbin/vzlicmonitor` in the running processes list.

Once the access to the proxy server is configured, proceed to install the license.

Virtuozzo Hybrid Server licenses are updated automatically by default. A few days before the current license expires, the `vzlicmon` service (a part of the `vzlicutils` RPM package) attempts to contact the Virtuozzo KA server over the Internet and obtain a new license.

6.1.2 Installing the License from Product Keys, Activation Codes, or License Files

To install a license from a product key or an activation code, run the `vzlicload -p` command or, to install from a license file, the `vzlicload -f` command. For example:

```
# vzlicload -p <key_or_code>
# vzlicload -f <license_file>
```

When you install a product key or activate a code, a license file is generated and installed in the `etc/vz/licenses/` directory on the server. The difference between the product key and activation code is that the code needs to be activated online, so the server must be connected to the Internet. To activate the code, the installation tool accesses the Key Authentication (KA) licensing server and transmits the specified activation code to it. The KA server generates a license file, sends it back, and the license file is installed on the server automatically.

6.2 Updating the License

You can use the `vzlicupdate` utility to update the license currently installed on the server. When executed, the utility tries to connect to the Key Authentication (KA) server, retrieve a new license, and install it on the server.

To update your license, do the following:

1. Make sure that the host where you wish to update the license is connected to the Internet.
2. Run `vzlicupdate` on the server (your server must have at least one public IPv4 address).

By default, `vzlicupdate` tries to access the KA server at `ka.virtuozzo.com` . However, you can explicitly specify what KA server to use using the `--server` option, e.g., `vzlicupdate --server ka.server.com` .

6.2.1 Switching License to a New HWID

If your Virtuozzo Hybrid Server license has become invalid due to changed HWID (e.g., after adding or removing a network card), you can have the license switch to the new HWID as follows:

```
# vzlicupdate -t -a <activation_code>
```

6.3 Transferring the License to Another Server

In case you need to transfer a license installed on one server to another server, you can do so as described below.

If you have a Virtuozzo Hybrid Server product key:

1. Remove the installed license from the source server with the `vzlicload -r <serial>` command (the serial is shown in `vzlicview` output).
2. Install the product key on the destination server:

```
# vzlicload -p <product_key>
```

A new license file will be generated and installed in the `/etc/vz/license/` directory on the server.

If you have a Virtuozzo Hybrid Server activation code:

1. Shut down the source server or remove the installed license from it with the `vzlicload -r <serial>` command (the serial is shown in `vzlicview` output).
2. Make sure that the destination server is up, connected to the Internet, and has at least one public IPv4 address.
3. On the destination server, run

```
# vzlicupdate -t -a <activation_code>
```

When executed, `vzlicupdate` sends the activation code to the KA server. The KA server verifies the code, generates a new license file, sends it back, and the license file is installed on the server automatically.

4. To check that the license transfer has been successful, run `vzlicview`. The information about the newly installed license should be displayed.

6.4 Viewing the License

You can use the `vzlicview` tool to view the information on the installed license and find out its current status. For example:

```
# vzlicview
Searching for installed licenses...
VZSRV
  owner_name="Autogenerated Trial Licenses"
  status="ACTIVE"
  version=7.0
  owner_id=70295522.10134133
  hwid="3D6F.FFB2.1CD5.1E47.1325.BBE0.6C66.ACF5"
  serial="BF3D.E943.18C5.5555.5D10.99D9.0310.DFA0"
  expiration="07/08/2016 03:00:00"
  start_date="06/07/2016 03:00:00"
  issue_date="06/08/2016 16:27:11"
  graceperiod=259200 (259200)
  key_number="PSBM.10134133.0001"
  cpu_total=8 (1)
  ct_total="unlimited" (0)
  architecture="x86"
  architecture="x86_64"
  platform="Linux"
  product="PSBM"
  nr_vms="unlimited" (2)
  subscription="70295522:dd482d1a-5268-4980-ae06-2a288a4fb7ab"
```

You can also view contents of license files with the `vzlicview -f` command. The output does not differ from that in the example above.

The license parameters are described in the following table.

Column	Description
<code>owner_name</code>	The name of the license owner.
<code>status</code>	License status. For description of license statuses, see License Statuses .
<code>version</code>	The version of Virtuozzo Hybrid Server for which the license was issued.

Continued on next page

Table 6.4.1 -- continued from previous page

Column	Description
owner_id	The ID of the license owner.
hwid	Server hardware identifier.
serial	License serial number. In particular, used to identify license files in <code>/etc/vz/licenses/</code> .
expiration	License expiration date, if the license is time-limited.
start_date	The date on which the license becomes active.
issue_date	The date on which the license was issued.
graceperiod	Period, in seconds, during which Virtuozzo Hybrid Server continues to function after the license has expired or if the number of running virtual machines and containers exceeds the limit defined by the license.
license_update_date	The date on which the license was last updated.
key_number	Number under which the license is registered on the Key Authentication server.
cpu_total	The total number of physical CPUs that the server is allowed to have.
ct_total	The total number of containers that are allowed to be simultaneously running on the server.
architecture	System architecture with which the license is compatible.
platform	Operating system with which the license is compatible.
product	Name of the product for which the license has been issued.
keyserver_host	The hostname and port of the Key Authentication server.
nr_vms	The number of virtual machines that are allowed to be simultaneously running on the server.
subscription	Virtuozzo Hybrid Server feature subscription key.

6.4.1 License Statuses

When viewing information on your licenses, pay special attention to the license status that can be one of the following:

Status	Description
ACTIVE	The installed license is valid and active.
VALID	The license is valid and can be installed.

Continued on next page

Table 6.4.1.1 -- continued from previous page

Status	Description
EXPIRED	The license has expired. If an installed license expires, running virtual environments continue to run. Newly started VEs, however, are suspended after 10 minutes. If they cannot be suspended, they are stopped gracefully. If this cannot be done, they are stopped forcefully. If the node is rebooted or if the license daemon is restarted, all VEs are treated as new and suspended (stopped) after 10 minutes.
GRACED	The license is installed on the server but is currently on the grace period because it has expired or the number of running virtual machines and containers exceeds the limit defined by the license.
INVALID	The license is invalid (e.g., because of server architecture mismatch) or corrupt.

CHAPTER 7

Keeping Your System Up To Date

This chapter explains how to keep the software on Virtuozzo Hybrid Server nodes up to date.

Important: It is strongly recommended to have all nodes run the same version of Virtuozzo Hybrid Server. At the very least make sure that product versions are only one major update apart. For example, before you update a node to Virtuozzo Hybrid Server 7.0 Update 7, make sure that all other machines in the cluster already run Update 6. The reason is that VMs created on newer nodes may fail to start on older nodes, thus causing issues with migration, restore from backup, high availability, and such.

The components that may need updating are:

- Virtuozzo Hybrid Server software:
 - Tools and libraries
 - ReadyKernel patch
 - Kernel (typically included in major updates)
- Virtual machines and containers hosted on the node:
 - Container EZ templates
 - KVM/QEMU hypervisor and guest tools in virtual machines

You can update Virtuozzo Hybrid Server nodes in several ways:

- Automatically (default)

- Automatically using smart updates
- Manually using `yum`, the standard tool for RPM-based Linux operating systems

All of these methods as well as ways to update the kernel, virtual machines, and containers are described further in this chapter.

7.1 Updating Overview

The typical way to switch to a new Virtuozzo Hybrid Server release is as follows:

1. The nodes are updated automatically by default. Or you can update them via `yum` at any time.

Note: Automatic and smart updating is disabled for Virtuozzo Storage nodes by default. Update them manually according to [Updating Nodes in Virtuozzo Storage Clusters](#) on page 187.

2. If a new kernel is installed, the system will attempt to notify you via `sendmail` (if it is running). In this case, you need to manually reboot the nodes, one by one, and switch to the new kernel. On each updated node:
 - 2.1. Stop all running virtual environments or migrate them to other Virtuozzo Hybrid Server 7 nodes to avoid downtime.
 - 2.2. Restart the node and switch to the new kernel.
 - 2.3. Start the virtual environments or migrate them back to the updated node, depending on what you did in step 1.

Note: You can also perform these steps in Virtuozzo Automator.

3. If you use Virtuozzo Automator, update the container `va-mn` either using it (see [Updating System Software](#)) or by running the following commands on the node:

```
# prlctl exec va-mn yum update
# prlctl restart va-mn
```

4. If you use Virtuozzo Storage management panel, update the container `vstorage-ui` by running the following commands on the node:

```
# prlctl exec vstorage-ui yum update
# prlctl restart vstorage-ui
```

7.2 Using A Proxy Server to Install Updates

If your server cannot access the Internet directly, you can update Virtuozzo Hybrid Server via a proxy server. To do that, specify the proxy server information in the `/etc/yum.conf` file as follows:

```
proxy=http://<hostname_or_IP_address>:<port>
proxy_username=<username>
proxy_password=<password>
```

After configuring `yum` to use the proxy server, proceed to installing updates as usual.

7.3 Updating Automatically

Starting from version 7.5, newly deployed Virtuozzo Hybrid Server nodes are configured to get updates automatically by default. Automatic updating requires a valid license to be installed. Manual updating remains possible and does not depend on license status.

Note the following, however:

- Automatic updating is not enabled for nodes upgraded from older versions of Virtuozzo Hybrid Server. You can enable automatic updates for them as explained in [Enabling Automatic Updates for Upgraded Nodes](#) on page 176.
- Automatic updating is disabled for Virtuozzo Storage clusters. For more information, see [Updating Nodes in Virtuozzo Storage Clusters](#) on page 187.

The following update policies are available:

- **fast**, get updates immediately after release.
- **slow**, get updates two weeks after release.
- **stable**, get updates six weeks after release.
- **auto**, a special policy that lets the Virtuozzo update server assign one of the other policies to a node.

Each node that requests updates automatically through the **auto** policy is registered by the Virtuozzo update server and assigned a regular update policy: **fast**, **slow**, or **stable**. To make automatic updating predictable,

a policy is assigned to a node only once. You can change it manually, however, as explained in *Changing Node Update Policy* on page 176.

Each node is configured as follows during installation of Virtuozzo Hybrid Server:

- The repository file `/etc/yum.repos.d/vz-auto-update.repo` is created. It lists node's unique repositories for automatic updating. Such URLs contain a `$hwid` variable that is replaced by node's hardware ID during updates. The file also lists the static repositories for each update policy.

The repositories in `/etc/yum.repos.d/vz-auto-update.repo` are disabled by default, while those in `/etc/yum.repos.d/virtuozzo.repo` are enabled. This ensures that manual updating via `yum update` works as usual.

Note: Repositories in `/etc/yum.repos.d/virtuozzo.repo` are the same as those for the **fast** update policy. So running `yum update` delivers the latest updates as soon as they are released.

- The `vzautoupdate` tool for working with update policies is installed.
- The timer service `vzautoupdate.timer` is enabled and started. It will run `vzautoupdate.service` daily to get updates. A schedule, i.e. the time of day to trigger an update at, is chosen randomly and written to `/etc/systemd/system/vzautoupdate.timer.d/override.conf`.
- Automatic updating is enabled. An update policy is not assigned yet.

When the timer triggers an update, the following happens, given that node's license is valid:

1. The repositories in `/etc/yum.repos.d/vz-auto-update.repo` are enabled, while those in `/etc/yum.repos.d/virtuozzo.repo` are disabled.
2. The `vzautoupdate` tool contacts the Virtuozzo update server at node's HWID-based repository URLs.
3. The update server checks node's license. If an update policy has been assigned to the node, the HWID-based URL redirects to a static repository for that update policy. Otherwise, the update server registers the node and assigns a policy to it. Then the HWID-based URL redirects to a static repository for that update policy.

Note: If a license is not installed or has expired, the HWID-based repositories are skipped as unavailable and packages are not updated.

4. Packages are downloaded and installed as usual by `yum update`.
5. The repositories in `/etc/yum.repos.d/vz-auto-update.repo` are disabled, while those in `/etc/yum.repos.d/virtuozzo.repo` are enabled again.
6. If an update contains a new kernel and a reboot is recommended, the tool attempts to notify the local administrator via `sendmail` (if it is running).

Note: The operation log is saved to `/var/log/vz_auto_update.log`.

7.3.1 Enabling Automatic Updates for Upgraded Nodes

To enable automatic updates for a node that has been upgraded to Virtuozzo Hybrid Server 7.5, do the following:

1. Install the required package:

```
# yum install vzautoupdate
```

Doing so delivers the repository file `/etc/yum.repos.d/vz-auto-update.repo` and enables automatic updating.

2. Enable and start the timer service:

```
# systemctl enable vzautoupdate.timer; systemctl start vzautoupdate.timer
```

The node will automatically request updates according to schedule. To see the schedule, run

```
# vzautoupdate get-schedule
```

You can also check `/etc/systemd/system/vzautoupdate.timer.d/override.conf`.

7.3.2 Changing Node Update Policy

You can manually switch between automatic updating and specific update policies with `vzautoupdate set-policy`.

Note: If you disable automatic updating for a node, the Virtuozzo update server will unregister that node

and clear its update policy after some time.

Switching to a policy means that when you run the update tool, it will get packages from that policy's repositories. For example, to change node update policy to **stable**, run

```
# vzautoupdate set-policy stable
```

If you need to check packages available for a policy, use `vzautoupdate available-updates`. For example:

```
# vzautoupdate available-updates stable
```

To update packages from the selected repository, run

```
# vzautoupdate update
```

Note: Running `yum update` will deliver packages from the repositories in `/etc/yum.repos.d/virtuozzo.repo`. They are the same as those for the **fast** update policy.

To switch back to automatic updating, run

```
# vzautoupdate set-policy auto
```

If the node is still registered at the Virtuozzo update server, its update policy will be as previously assigned. Otherwise the node will get a policy according to the rules explained earlier.

7.4 Updating Automatically with Smart Updates

Smart updates build on automatic updates, improving their safety and providing the user with more control. Enabling smart updates means joining Virtuozzo Hybrid Server nodes to a Kubernetes cluster and creating an update manager entity to update the nodes according to set rules. Enabling smart updates disables automatic updates.

As it is recommended to set up a highly available Kubernetes cluster in production, three or more control plane nodes in the same subnet are required to set up smart updates. In addition, you will need a free IP address at which the control plane will be available.

Note the following:

- Smart updates can currently update ReadyKernel patches as well as Virtuozzo Hybrid Server and VzLinux RPM packages on both standalone nodes and those in Virtuozzo Storage clusters.
- Node update policies are initially obtained from the Virtuozzo update server as is with automatic updates. Then, however, a local update schedule, maintenance window, and policies are used for each node. They can be configured by changing the local update manager specification. Nodes with the **auto** policy are assigned one of the following policies:
 - **fast**, updates installed immediately after publication, assigned to 10% of nodes
 - **slow**, updates installed two weeks after publication, assigned to 40% of nodes
 - **stable**, updates installed six weeks after publication, assigned to 50% of nodes
- Updates are first checked for during the setup and then each 24 hours. Also, a check is performed each time the RPM database on the node is modified. This includes modifying the lists of banned RPM packages and ReadyKernel patches, as well as manual installation, updating, or removal of RPM packages. Available updates are installed as soon as the update policy permits. A custom schedule and a maintenance window can be set (see *Setting the Maintenance Window and Schedule* on page 182). If an update fails, the next attempt is made after at least 4 hours.
- If a critical service on the node (e.g., VM dispatcher or Virtuozzo Storage mount point) stops working during an update or within 10 minutes after it, the ReadyKernel patch or all packages from the RPM transaction that caused it are added to a ban list. Banned patches and packages will not be installed in the future unless manually removed from the ban list (as explained in *Excluding RPM Packages and ReadyKernel Patches from Updates* on page 184).
- The following repositories are checked for updates by default:
 - `virtuozzolinix-base`
 - `virtuozzolinix-updates`
 - `virtuozzo-os`
 - `virtuozzo-updates`

You can change this list manually (see *Setting the Repositories* on page 183).

- Nodes are updated one by one. The node that has not been updated the longest is the first one in the queue.
- Updating of Virtuozzo Storage nodes starts only if the storage cluster has the **HEALTHY** status.

- Updated nodes need to be rebooted manually. For such nodes, a dedicated Prometheus metric, `node_vz_needs_reboot`, is set to 1.

7.4.1 Enabling Smart Updates

To set up smart updates with a highly available lightweight Kubernetes cluster, do the following:

1. Install the required RPM package:

```
# yum install smart-updates
```

This will install lightweight Kubernetes (k3s) among other dependencies.

2. Set up a highly available Kubernetes cluster.

Fault tolerance is ensured by `keepalived`. It provides a virtual IP address that is assigned to a control plane node. If it fails, the IP address migrates to a healthy control plane node.

- 2.1. On each control plane node, prepare a service configuration file. Use a sample supplied by Kubernetes:

```
# cp /usr/share/k3s/k3s_keepalived.conf.sample /etc/keepalived/keepalived.conf
```

Replace the placeholders in `keepalived.conf` as follows:

- `${INTERFACE}` with the name of the node's network interface that will negotiate for the virtual IP address. E.g., `eth0`.
- `${AUTH_PASS}` with a password, e.g., `mypasswd`. It must be the same on all control plane nodes.
- `${APISERVER_VIP}` with the virtual IP address at which the control plane will be available. E.g., `192.168.1.2`.

- 2.2. On each control plane node, prepare a health check script. It will be called periodically to check that the node holding the virtual IP address is operational. Use a sample supplied by Kubernetes as well:

```
# cp /usr/share/k3s/check_apiserver.sh.sample /etc/keepalived/check_apiserver.sh
```

Replace the placeholder in `check_apiserver.sh` as follows:

- `${APISERVER_VIP}` with the virtual IP address at which the control plane will be available. E.g., `192.168.1.2`.

- 2.3. On each control plane node, configure the firewall:

```
# firewall-cmd --permanent --add-rich-rule='rule protocol value="vrrp" accept'
# firewall-cmd --permanent \
--add-port=8472/udp \
--add-port=9443/tcp \
--add-port=2379-2380/tcp
# firewall-cmd --reload
```

- 2.4. On the first control plane node, create the Kubernetes server configuration file and enable the `keepalived`, `docker`, and `k3s-server` services:

```
# echo "K3S_TOKEN=<my_secret_token>" >> /etc/k3s/k3s-server.env
# echo -e "K3S_EXTRA_FLAGS=\"--cluster-init\"" >> /etc/k3s/k3s-server.env
# systemctl enable keepalived docker k3s-server --now
```

Where `<my_secret_token>` is an arbitrary secret string.

- 2.5. On the second and third control plane nodes, create the Kubernetes server configuration files and enable the `keepalived`, `docker`, and `k3s-server` services:

```
# echo "K3S_TOKEN=<my_secret_token>" >> /etc/k3s/k3s-server.env
# echo "K3S_URL=https://<virt_IP>:9443" >> /etc/k3s/k3s-server.env
# systemctl enable keepalived docker k3s-server --now
```

Where:

- `<my_secret_token>` is the secret string from the previous step.
- `<virt_IP>` is the virtual IP address from `/etc/keepalived/keepalived.conf`.

- 2.6. On each agent node, open the required port, create the Kubernetes agent configuration file, and enable the `docker` and `k3s-agent` services:

```
# firewall-cmd --permanent --add-port=8472/udp && firewall-cmd --reload
# echo "K3S_TOKEN=<my_secret_token>" >> /etc/k3s/k3s-agent.env
# echo "K3S_URL=https://<virt_IP>:9443" >> /etc/k3s/k3s-agent.env
# systemctl enable docker k3s-agent --now
```

Where:

- `<my_secret_token>` is the secret string from the previous step.
- `<virt_IP>` is the virtual IP address from `/etc/keepalived/keepalived.conf`.

Note: You can repeat this step later to add new nodes to the Kubernetes cluster.

- 2.7. Check that the cluster is healthy. For example:

```
# kubectl get nodes
NAME                STATUS    ROLES    AGE     VERSION
node1.example.local Ready    etcd,master 3h36m   v1.19.7-k3s1
node2.example.local Ready    etcd,master 3h33m   v1.19.7-k3s1
node3.example.local Ready    etcd,master 3h33m   v1.19.7-k3s1
```

3. Deploy the maintenance operator in the Kubernetes cluster:

```
# kubectl apply -f /usr/share/smart-updates/maintenance-operator.yaml
```

4. Wait until the maintenance operator is ready:

```
# kubectl wait --namespace=vz-maintenance-operator \
--for=condition=Complete job vz-maintenance-deployment-job-v<version> --timeout=1h

job.batch/vz-maintenance-deployment-job-v<version> condition met
```

Where <version> is the version of the smart-updates package, e.g., 1.0.0. You can also obtain the full job name from the maintenance operator configuration file. For example:

```
# grep "vz-maintenance-deployment-job-v" /usr/share/smart-updates/maintenance-operator.yaml
name: vz-maintenance-deployment-job-v1.0.0
```

5. Create an UpdateManager instance to initiate synchronization of updates:

```
# kubectl apply -f /usr/share/smart-updates/update-manager.yaml
```

7.4.2 Checking Node Status

You can now check the state of updates on nodes in the Kubernetes cluster with

```
# kubectl -n vz-maintenance-operator get updatemanager -o <yaml|json>
```

A typical response (abridged) may look like this:

```
<...>
"status": {
  "nodes": [
    {
      "id": "node1.example.local",
      "kernel": "3.10.0-1160.21.1.vz7.174.10",
      "readyKernel": {
        "status": "UpToDate"
      },
      "rpmPackages": {
        "status": "UpToDate"
      },
    },
  ],
}
```

```

    "stageName": "stable"
  },
  {
    "id": "node2.example.local",
    "kernel": "3.10.0-1160.21.1.vz7.174.10",
    "readyKernel": {
      "status": "UpToDate"
    },
    "rpmPackages": {
      "numberOfPackagesToUpdate": 1,
      "status": "UpToDate"
    },
    "stageName": "fast"
  },
  {
    "id": "node3.example.local",
    "kernel": "3.10.0-1160.21.1.vz7.174.10",
    "readyKernel": {
      "status": "UpToDate"
    },
    "rpmPackages": {
      "status": "UpToDate"
    },
    "stageName": "slow"
  }
]
}
<...>

```

The sample above shows that all nodes are up-to-date, as per their update policies. It also reports that one RPM package has recently been installed on the second node, `node2.example.local`.

7.4.3 Setting the Maintenance Window and Schedule

You can set one or more maintenance windows during which updating is allowed to start. The format is `0h00m`, e.g., `1h30m` for a window that is one hour and 30 minutes long.

You can also set an update schedule similar to that used by automatic updates. The format is standard Unix cron format, e.g., `0 1 * * 1-5` means “each day at 01:00 from Monday to Friday”.

Do the following:

1. Open the update manager specification for editing:

```
# kubectl -n vz-maintenance-operator edit updatemanagers default
```

2. Add the required parameters to the spec part. For example:

```
spec:
  <...>
  maintenanceWindows:
    - duration: 1h30m
      schedule: 0 1 * * * 1-5
  <...>
```

3. Save and close the specification.

7.4.4 Setting the Repositories

RPM package updates are obtained from default repositories listed early in this section. You can, however, change them separately for each policy. If you do so, only the repositories that you provide will be used by smart updates. You will still be able to manually install RPM packages from unlisted repositories.

Do the following:

1. Open the update manager specification for editing:

```
# kubectl -n vz-maintenance-operator edit updatemanagers default
```

2. Add the needed repository IDs under the `repositories` key for the desired policy. Add the key if needed.

Note: You can get repository IDs from `yum repolist`.

For example, to use only VzLinux repositories for the **fast** policy:

```
spec:
  stages:
    - name: fast
      nodeSelector:
        matchLabels:
          maintenance.virtuozzo.io/auto-updates-policy: fast
      repositories:
        - virtuoazzolinux-base
        - virtuoazzolinux-updates
```

3. Save and close the specification.

7.4.5 Excluding RPM Packages and ReadyKernel Patches from Updates

You can prevent specific RPM packages and ReadyKernel patches from being installed during updating.

The format for RPM packages is <name>-<version>-<release>.<arch>, e.g., bash-4.2.46-34.v17.x86_64. The format for ReadyKernel patches is X-Y-rZ, e.g., 123-4-r5.

Do the following:

1. Open the update manager specification for editing:

```
# kubectl -n vz-maintenance-operator edit updatemanagers default
```

2. List the RPM packages and ReadyKernel patches in the spec section, under bannedPackages and bannedPatches keys, respectively. For example:

```
spec:
  <...>
  bannedPackages:
  - bash-4.2.46-34.v17.x86_64
  bannedPatches:
  - 123-4-r5
  <...>
```

3. Save and close the specification.

To allow installation of banned patches and packages again, remove them from the list.

7.4.6 Changing Node Update Policies

To change a node's update policy, re-label the node in the update manager specification as follows:

```
# kubectl label node <node> maintenance.virtuozzo.io/auto-updates-policy=<policy> --overwrite
```

For example:

```
# kubectl label node node1.example.local maintenance.virtuozzo.io/auto-updates-policy=fast --overwr
```

Such re-labelling does not overwrite node update policies on the Virtuozzo update server. If smart updates are disabled, automatic updates will continue working as before.

7.4.7 Suspending Updates for All or Specific Nodes

You can prevent updates from being installed either across the entire Kubernetes cluster or on particular nodes. It may be useful to prepare the nodes for maintenance, for example.

To suspend updating cluster-wide, set `suspend: true` in the update manager specification:

1. Open the update manager specification for editing:

```
# kubectl -n vz-maintenance-operator edit updatemanagers default
```

2. Add the required parameters to the spec part. For example:

```
spec:
  <...>
  suspend: true
  <...>
```

3. Save and close the specification.

To suspend updates on a specific node, assign the `suspend` label to it:

```
# kubectl label nodes node1.example.local maintenance.virtuozzo.io/suspend=""
```

The change will be reflected in the node status. For example:

```
# kubectl -n vz-maintenance-operator get updatemanager -o json
"status": {
  "nodes": [
    {
      "id": "node1.example.local",
      "kernel": "3.10.0-1160.21.1.vz7.174.10",
      "readyKernel": {
        "reason": "ManuallySuspended",
        "status": "Suspended"
      },
      "rpmPackages": {
        "reason": "ManuallySuspended",
        "status": "Suspended"
      },
      "stageName": "stable"
    },
    <...>
```

To resume updates for a suspended node, remove the `suspend` label. For example:

```
# kubectl label nodes node.example.local maintenance.virtuozzo.io/suspend-
```

Again, the change will be reflected in the node status. For example:

```
# kubectl -n vz-maintenance-operator get updatemanager -o json
"status": {
  "nodes": [
    {
      "id": "node1.example.local",
      "kernel": "3.10.0-1160.21.1.vz7.174.10",
      "readyKernel": {
        "status": "UpToDate"
      },
      "rpmPackages": {
        "status": "UpToDate"
      },
      "stageName": "stable"
    },
    <...>
  ]
}
```

7.4.8 Disabling Smart Updates

To disable smart updates and revert to automatic updates, do the following:

1. Delete the Kubernetes entities related to the maintenance operator:

```
# kubectl delete -f /usr/share/smart-updates/maintenance-operator.yaml
```

2. If you deployed Kubernetes solely for smart updates and no longer need it, disable its server and agent services:

```
# systemctl disable --now k3s-*
```

Run this command on each node.

7.5 Updating Manually

You can update Virtuozzo Hybrid Server on a node which not in a Virtuozzo Storage cluster by running `yum update`. It will do the following:

1. Access Virtuozzo Hybrid Server repositories.
2. Check for available updates for tools, libraries, EZ templates, Virtuozzo Hybrid Server kernel, and the latest ReadyKernel patch for the current kernel. If a new Virtuozzo Hybrid Server kernel is available, it will check for the corresponding ReadyKernel patch as well.
3. Install the updates on your system.

Repeat these steps on each Virtuozzo Hybrid Server node which is not in a Virtuozzo Storage cluster.

7.6 Updating Nodes in Virtuozzo Storage Clusters

Important: If nodes in the cluster run different versions (updates) of Virtuozzo Hybrid Server, make sure to create new cluster services on the nodes running the newest version of Virtuozzo Hybrid Server.

When updating Virtuozzo Storage cluster services running on Virtuozzo Hybrid Server nodes, follow these steps:

1. Ensure the cluster is fully healthy using `vstorage -c <cluster_name> top`.
2. Disable `shaman.service` on just one cluster node you want to update, as described in [Disabling High Availability on Nodes](#) on page 202.
3. Install the updates on the cluster node with `yum update`.
4. Reboot the updated node.
5. Reenable `shaman.service` once the node is back online:

```
# systemctl enable shaman.service
# systemctl start shaman.service
# systemctl enable pdrs.service
# systemctl start pdrs.service
```

6. Ensure the cluster is fully healthy using `vstorage -c <cluster_name> top`.

Repeat these steps for each cluster node by updating them individually. The cluster must be healthy before and after each node is updated.

Note: After upgrading a node in a mixed cluster, you cannot migrate VEs (virtual machines and containers) created in datastores with encoding EC 3+2, 5+2, 7+2, or 17+3 from VHS 7.5 Update 4 to VHS 7.5 Update 3. However, the migration of VEs created in local datastores and datastores with a 3-replica and 2-replica data redundancy mode is available. A mixed cluster is not supported and exists during the upgrade only.

7.7 Updating the Kernel

To get a new kernel for Virtuozzo Hybrid Server, update the `vzkernel` and `vzkernel-devel` packages:

```
# yum update vzkernel vzkernel-devel
```

Reboot the node to switch to the new kernel. You may want to temporarily live-migrate running virtual environments to other nodes to avoid downtime.

7.8 Updating the Kernel with ReadyKernel

Virtuozzo ReadyKernel is a kpatch-based service shipped with Virtuozzo Hybrid Server and available out-of-the-box on hardware nodes with active licenses. ReadyKernel offers a more convenient, rebootless alternative to updating the kernel the usual way and allows you not to wait for scheduled server downtime to apply critical security updates. ReadyKernel enables you to receive cumulative kernel patches that fix critical security issues and apply these patches without having to reboot the server. ReadyKernel updates are released for Virtuozzo Hybrid Server kernels younger than 18 months. When a kernel becomes older, it should be updated, e.g., to the latest one, so you can keep receiving ReadyKernel updates.

Upon installation, the patches are loaded into server RAM and immediately applied to the kernel. If the server reboots, these patches are reapplied to the kernel on boot.

If later you install a new kernel or major kernel update that requires a reboot, the downloaded patches will remain on the server but will not be applied.

Note: At any time, you can check the details of the applied ReadyKernel patch with `readykernel info`.

ReadyKernel patches can be received and installed automatically or manually as described in the following sections.

7.8.1 Installing ReadyKernel Patches Automatically

If automatic updating was not disabled during the installation, ReadyKernel will check for new patches daily at 12:00 server time. If a patch is available, ReadyKernel will download, install, and load it for the current kernel.

If automatic updating is disabled, you can re-enable it with the following command:

```
# readykernel autoupdate enable <hour>
```

The service will check for patches daily at the specified <hour> (set in 24-hour format, server time) by means of the `cron.d` script.

To disable automatic updating, run

```
# readykernel autoupdate disable
```

7.8.2 Managing ReadyKernel Patches Manually

7.8.2.1 Downloading, Installing, and Loading ReadyKernel Patches

To download, install, and instantly load the latest ReadyKernel patch for the current kernel, do the following:

1. Check for new ReadyKernel patches:

```
# readykernel check-update
```

2. If a new patch is available, download, install, and instantly load it for the current kernel by running:

```
# readykernel update
```

Note: You can also do this with `yum update`.

ReadyKernel patches are cumulative, i.e. the latest patch includes all the previous ones. To keep the kernel secure, you only need to install and load the latest patch.

7.8.2.2 Loading and Unloading ReadyKernel Patches

To manually load the latest installed ReadyKernel patch to the kernel, do one of the following:

- If an older patch is already loaded, unload it first, then load the latest patch by running:

```
# readykernel load-replace
```

- If no older patches are loaded, load the latest patch by running:

```
# readykernel load
```

To unload the patch from the current kernel, run

```
# readykernel unload
```

7.8.2.3 Installing and Removing ReadyKernel Patches for Specific Kernels

If multiple kernels are installed on the server, you can install a ReadyKernel patch for a specific kernel:

```
# yum install readykernel-patch-<kernel_version>
```

To remove a specific ReadyKernel patch from the server, run

```
# yum remove readykernel-patch-<kernel_version>
```

7.8.2.4 Downgrading ReadyKernel Patches

If you experience problems with the latest ReadyKernel patch, you can downgrade it to an older version if one is available.

To downgrade a patch for the current kernel to the previous version, run

```
# yum downgrade readykernel-patch-$(uname -r)
```

To downgrade a patch for a specific kernel to the previous version, run

```
# yum downgrade readykernel-patch-<kernel_version>
```

You can run these commands multiple times to downgrade to the patch version you need. Alternatively, you can downgrade a patch to a specific version by specifying the desired patch version. For example:

```
# yum downgrade readykernel-patch-12.7-0.4-17.v17
```

7.8.3 Disabling Loading of ReadyKernel Patches on Boot

If for some reason you do not want ReadyKernel patches to be applied at boot time, run the following command:

```
# readykernel autoload disable
```

To re-enable automatic loading of ReadyKernel patches on boot, run

```
# readykernel autoload enable
```

7.8.4 Managing ReadyKernel Logs

ReadyKernel logs event information in `/var/log/messages` and `/var/log/kpatch.log`. You can specify logging parameters for the latter in the configuration file `/etc/logrotate.d/kpatch`. For more information on parameters you can use, see the `logrotate` man page.

7.9 Updating Software in Virtual Machines

To keep software in your virtual machines up to date, you can use the same means you would use on standalone computers running the corresponding operating systems:

- In Linux-based virtual machines, you can use the native Linux updaters (`up2date`, `yum`, or `yast`).
- In Windows-based virtual machines, you can use the native Windows updaters (e.g., the Windows Update tool).

7.9.1 Updating the KVM/QEMU Hypervisor in Virtual Machines

Virtuozzo Hybrid Server can update KVM/QEMU hypervisor live in running virtual machines that have KVM/QEMU version 2.6.0 or newer.

To do this, install the `vz-qemu-engine-updater` package and update the `qemu-kvm-vz` package:

```
# yum install vz-qemu-engine-updater
# yum update qemu-kvm-vz
```

Updating `qemu-kvm-vz` starts a 10-minute timer to give `yum` time to complete the operation. After that the `vz-qemu-engine-updater` tool begins updating KVM/QEMU in each running virtual machine, one at a time.

Immediate updating may not be possible in several cases:

- `yum` is currently locked on the node (in this case, no VMs can be updated automatically until the lock is released).
- A VM is changing states (e.g., from running to stopped).
- Configuration changes are being applied to a VM.
- A backup of a VM is being created.
- Any other `prlctl` operation is executed on a VM.

The KVM/QEMU updater will skip such VMs and queue them for a later update. The updater will perform a set number of retries to update VMs that have been skipped, each retry after a set delay. If retries are exhausted or the update fails for some reason, the virtual machine is left running with the outdated KVM/QEMU.

To manually disable automatic updates, mask the updater service:

```
# systemctl mask vz-qemu-engine-updater.service
```

To re-enable updates, unmask the updater service:

```
# systemctl unmask vz-qemu-engine-updater.service
```

To check if virtual machines have been successfully updated, view the log file:

```
# journalctl -u vz-qemu-engine-updater
<...>VM MyVM (was using qemu-kvm-vz-2.6.0-28.3.6.vz7.30) has been successfully updated.
<...>Finished updating VMs.
<...>Successfully updated QEMU engine for all running VMs.
```

To configure the number of retries, the delay between them, and other parameters, refer to the `vz-qemu-engine-updater.json` man page and edit the `/var/lib/vz-qemu-engine-updater.json` configuration file.

7.9.1.1 Updating the KVM/QEMU Hypervisor Manually

If, for some reason, the KVM/QEMU hypervisor was not automatically updated in a running VM, you can do it manually as follows:

1. Make sure that the VM uses an outdated version of the hypervisor.

Check the version installed on the node:

```
# rpm -qa qemu-kvm-vz
qemu-kvm-vz-<version>.x86_64
```

And compare it with the version the VM currently uses:

```
# virsh qemu-monitor-command <VM_name> '{"execute": "query-version"}'
<...>"package": "(qemu-kvm-vz-<version>)"<...>
```

2. Make sure no `prlctl` operations are being executed on the VM.
3. Update the hypervisor that the VM uses:

```
# prlctl update-qemu <VM_name>
```

7.9.2 Updating Virtuozzo Guest Tools in Virtual Machines

Starting from Virtuozzo Hybrid Server 7.0.4 (Update 4), Virtuozzo guest tools in virtual machines are updated automatically via a weekly cron job that starts the `vz-guest-tools-updater` tool.

The following requirements must be met:

- The `vz-guest-tools-updater` package must be installed on the node.
- The virtual machine must have the `--tools-autoupdate` parameter set to `on` (this is the default behavior).

Important: The first automatic installation of Virtuozzo guest tools to a running SUSE Linux Enterprise Server/Desktop 15 SP6 virtual machine with the `vz-guest-tools-updater` tool requires stopping the VM. If you prefer the first-time installation of the guest tools without downtime, install them manually from our system ISO after ensuring the previously mounted ISO is ejected. Use the following CLI command:

```
# prlctl installtools MyVM
```

Otherwise, via Virtuozzo Automator, go to the **Configure** tab and select **Install Guest Tools**.

With the Virtuozzo guest tools set up, the subsequent run of `vz-guest-tools-updater` for the running VM will update the guest tools online (without downtime) to the latest version.

Note: Starting from Virtuozzo Hybrid Server 7.0.7 (Update 7), guest tools are also automatically installed in virtual machines on their next start (see *Installing Virtuozzo Guest Tools* on page 19). To disable automatic installation of guest tools, set the `InstallTools` parameter to `false` in the `/etc/vz/tools-update.conf` configuration file.

The `vz-guest-tools-updater` tool builds a list of VMs with the enabled `--tools-autoupdate` parameter and outdated guest tools. After that, a 5-minute timer triggers simultaneous guest tools update in a configurable number of VMs. If an update attempt fails, the tool will queue that VM for another try. If the second attempt fails, the VM's guest tools will be left outdated.

Important: By default, the updater tool obtains the new guest tools from the official repository. If the repository is not accessible, the updater tool forcibly mounts the guest tools image to VM's optical disk drive

even if it is already in use.

Windows virtual machines need to be restarted to complete the update of guest tools. On every such update, administrators inside these VMs receive a reboot notification upon login or immediately if they are logged in.

You can configure the number of VMs whose guest tools are to be updated simultaneously by changing the value of the `MaxVMs` parameter in the `/etc/vz/tools-update.conf` configuration file.

To check the update status of guest tools in one or more VMs, use the `--get-state` option for the `vz-guest-tools-updater` tool and specify VM names in a sequence. For example:

```
# vz-guest-tools-updater --get-state <VM1_name> [<VM2_name> ...]
```

If the guest tools in the given virtual machine are up to date, the command output will be as follows:

```
{<VM_UUID>} (<VM_name>): Tools are up to date
```

To disable automatic updating of Virtuozzo guest tools for a VM, run the following command:

```
# prctl set <VM_name> --tools-autoupdate off
```

To manually update guest tools in one or more VMs, start the `vz-guest-tools-updater` script by specifying VM names in a sequence. For example:

```
# vz-guest-tools-updater <VM1_name> [<VM2_name> ...]
```

Note: If run without any parameters, `vz-guest-tools-updater` starts updating the guest tools in `MaxVMs` random VMs.

7.10 Updating Containers

Virtuozzo Hybrid Server provides three means of keeping your containers up to date:

- Updating EZ template RPM packages on the node so that new containers are up to date.
- Updating caches of the EZ templates installed on the node to create new containers quicker.
- Updating EZ templates inside existing containers to keep existing containers up to date.

7.10.1 Updating EZ Templates

You can update an EZ template like any other RPM package using the `yum update` command. For example:

```
# yum update centos-6-x86_64-ez
...
Updated:
  centos-6-x86_64-ez.noarch 0:4.7.0-1
Complete!
```

Note the following:

- Updating an EZ template requires that you append `ez` to template name.
- You can also use the `vzpkg update template` command to update EZ templates.

7.10.2 Updating EZ Template Caches

With the release of new updates for the corresponding Linux distribution, the created EZ template cache can become obsolete. Virtuozzo Hybrid Server allows you to quickly update your EZ template caches using the `vzpkg update cache` command.

Note: If you are going to update the cache of a commercial EZ template (e.g., Red Hat Enterprise Server 6 or SLES 11), you should first update software packages in the remote repository used to handle this EZ template and then proceed with updating the EZ template cache.

When executed, `vzpkg update cache` checks the cache directory in the template area (`/vz/template/cache` by default) on the hardware node and updates all existing tarballs in this directory. However, you can explicitly indicate the tarball for what EZ template should be updated by specifying the EZ template name. For example, to update the tarball for the `centos-6-x86_64` EZ template, run this command:

```
# vzpkg update cache centos-6-x86_64
Loading "rpm2vzrpm" plugin
Setting up Update Process
Setting up repositories
base0          100% |=====| 951 B    00:00
base1          100% |=====| 951 B    00:00
base2          100% |=====| 951 B    00:00
base3          100% |=====| 951 B    00:00
<...>
```

Upon the `vzpkg update cache` execution, the old tarball name gets the `-old` suffix (e.g.,

centos-x86.tar.gz-old).

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

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

7.10.3 Updating EZ Templates in Existing Containers

You can update packages of a container EZ template with the `vzpkg update` command. For example, to update all packages of the template `centos-6-x86_64` used in the container with the UUID

`26bc47f6-353f-444b-bc35-b634a88dbbcc`, run

```
# vzpkg update 26bc47f6-353f-444b-bc35-b634a88dbbcc centos-6-x86_64
<...>
  Updating: httpd                ### [1/4]
  Updating: vzdev                ### [2/4]
  Cleanup  : vzdev                ### [3/4]
  Cleanup  : httpd                ### [4/4]
Updated: httpd.i386 0:2.0.54-10.2 vzdev.noarch 0:1.0-4.swsoft
Complete!
Updated:
httpd      i386      0:2.0.54-10.2
vzdev     noarch     0:1.0-4.swsoft
```

Note the following:

- Updating EZ templates is supported for running containers only.
- If you are going to update the cache of a commercial EZ template, first update software packages in the remote repository used to handle this EZ template and then proceed with updating the EZ template cache.

As you can see from the example above, the `httpd` and `vzdev` applications have been updated for the `centos-6-x86_64` EZ template. If you wish to update all EZ templates (including the EZ template) inside the container at once, run:

```
# vzpkg update 26bc47f6-353f-444b-bc35-b634a88dbbcc
<...>
Running Transaction
  Updating  : hwdata                #### [1/2]
  Cleanup   : hwdata                #### [2/2]
Updated: hwdata.noarch 0:1.0-3.swsoft
```

```
Complete!  
Updated:  
  hwdata                noarch      0:0.158.1-1
```

In the example above, only the `hwdata` package inside the container needed to be updated.

7.11 Downgrading to Older Versions

Warning: Back up the system before downgrading.

You can roll back updates by undoing the respective `yum` transactions with the `yum history undo` command.

If `systemd` is to be downgraded as part of the rollback, perform the procedure as follows:

1. Back up the `systemd` configuration file:

```
# mv /etc/yum/protected.d/systemd.conf /etc/yum/protected.d/systemd.conf.backup
```

2. Undo the required `yum` transaction:

```
# yum history undo <transaction_ID>
```

3. Restore the `systemd` configuration file from backup:

```
# mv /etc/yum/protected.d/systemd.conf.backup /etc/yum/protected.d/systemd.conf
```

CHAPTER 8

Managing High Availability Clusters

This chapter explains managing high availability (HA) for servers that participate in Virtuozzo Storage clusters. High availability requires a Virtuozzo Hybrid Server license to be installed.

High availability keeps virtual machines, containers, and iSCSI targets operational even if the node they are located on fails. In such cases, the affected virtual environments continue working on other, healthy nodes in the cluster. High availability is ensured by:

- Metadata redundancy. For a Virtuozzo Storage cluster to function, not all but just the majority of MDS servers must be up. By setting up multiple MDS servers in the cluster you will make sure that if an MDS server fails, other MDS servers will continue controlling the cluster.
- Data redundancy. Copies of each piece of data are stored across different storage nodes to ensure that the data is available even if some of the storage nodes are inaccessible.

Note: The redundancy is achieved by one of two methods: replication or erasure coding (for details, see [Understanding Data Redundancy](#)).

- Monitoring of node health.

You may need to follow different instructions in this chapter based on your scenario which can be one of these:

- You use Virtuozzo Storage with CLI management, you have not enabled HA for virtual machines and containers by means of the client server role during installation, and you want to enable it. For details

on the client server role, see [|vstorage| Server Roles](#).

In this case, continue reading this chapter.

- You use Virtuozone Storage with CLI management, you have enabled HA for virtual machines and containers by means of the client server role during installation, and you want to change the default resource relocation mode. For details on the client server role, see [|vstorage| Server Roles](#).

In this case, HA in the DRS mode is automatically enabled for virtual machines and containers on the node. To change the resource relocation mode, e.g., to round-robin, follow the instructions in [Configuring Resource Relocation Modes](#) on page 202.

- You use Virtuozone Storage with GUI management, you have not assigned iSCSI network roles (see [Network Interface Roles](#)) or created S3 clusters yet in the Virtuozone Storage management panel (see [Creating the S3 Cluster](#)), and you want to configure HA manually, e.g., to manually choose a resource relocation mode.

In this case, continue reading this chapter.

- You use Virtuozone Storage with GUI management, you have assigned the iSCSI network role to a node's network interface (see [Network Interface Roles](#)) or joined a node to an S3 cluster in the Virtuozone Storage management panel (see [Creating the S3 Cluster](#)), and you want to change the default resource relocation mode.

In this case, HA in the round-robin mode is automatically enabled for virtual machines and containers on the node. To change the resource relocation mode, e.g., to DRS, follow the instructions in [Configuring Resource Relocation Modes for Nodes Participating in S3 or iSCSI Export](#) on page 204.

8.1 Prerequisites for High Availability

For the high availability feature to work, the following prerequisites must be met:

- A Virtuozone Storage cluster must be set up in your network. High availability is supported only for nodes joined to Virtuozone Storage clusters.
- The Virtuozone Storage cluster must have 5 or more nodes.
- Virtual machines and containers residing on a node must be stored in the cluster:
 - If you use Virtuozone Storage with GUI management, log in to the management panel, create datastores, and place your virtual machines and containers in them as described in [Managing](#)

Datstores.

- If you use Virtuozone Storage with CLI management, virtual machines and containers are automatically configured to be stored in the cluster on nodes for which the **Client Server Role** was chosen during installation. You can also configure them manually as explained in [Stage 3: Configuring Virtual Machines and Containers](#).
- The chosen redundancy mode must protect your data against a simultaneous failure of two or more nodes. To see the list of redundancy modes and their differences, refer to [Understanding Data Redundancy](#).
 - If you use Virtuozone Storage with GUI management, you select the redundancy mode while creating datstores.
 - If you use Virtuozone Storage with CLI management, you need to change the default redundancy parameters for the directories storing your virtual machines and containers.

For example, to set the 3:2 replicas mode for the `vmprivate` and `private` directories (with VMs and containers, respectively) in the cluster `stor1`, run the following commands:

```
# vstorage set-attr -R /vstorage/stor1/vmprivate replicas=3
# vstorage set-attr -R /vstorage/stor1/private replicas=3
```

- (Virtuozone Hybrid Server 7.0.6 and newer) Each node in the cluster must have the `SERVER_UUID` parameter set in the `/etc/vz/vz.conf` file. Virtuozone Storage requires it to provide access to container disks. For the parameter description, see [Global System Configuration File](#).

8.2 Enabling and Disabling High Availability for Nodes

To enable high availability on a node means to enable it for all virtual machines and containers on this node that are stored in the Virtuozone Storage cluster.

Note the following:

- When you assign the iSCSI network role to a node's network interface (see [Network Interface Roles](#)) or joined a node to an S3 cluster in the Virtuozone Storage management panel (see [Creating the S3 Cluster](#)), high availability is automatically enabled for virtual machines and containers on this node in the round-robin mode.

= When you enable HA for virtual machines and containers for the client server role during installation of Virtuozzo Storage with CLI management (see [|vstorage| Server Roles](#)), high availability is automatically enabled for virtual machines and containers on this node in the DRS mode.

To enable high availability on a node, do the following:

1. Update Virtuozzo Storage to the latest version with `yum update`.
2. Run the `hastart` script:

```
# hastart -c <cluster> -n <storage_network/network_mask>
```

Where `<cluster>` is the cluster name, e.g., `vstor1`, `<storage_network>` is the cluster internal network, and `<network_mask>` covers all the nodes in the cluster.

The script will automatically install, configure, and start the HA services on the node as well as add the node to the HA configuration.

After enabling HA for the node, you can check the result with the `shaman stat` command.

8.2.1 Disabling High Availability for Specific Virtual Machines and Containers

By default, if high availability is enabled on a node, it affects all virtual machines and containers on said node. If necessary, you can disable HA for specific virtual machines and containers using the `prlctl set` command. For example:

```
# prlctl set MyVM --ha-enable no
```

To re-enable HA support, run:

```
# prlctl set MyVM --ha-enable yes
```

8.2.2 Enabling High Availability for iSCSI Targets

Note: If you use Virtuozzo Storage with GUI management, enable HA for iSCSI targets by assigning the iSCSI network role to a node's network interface (see [Network Interface Roles](#)). Keep in mind that doing so also enables HA for virtual machines and containers on this node that are stored in the Virtuozzo Storage cluster.

If you use Virtuozzo Storage with CLI management, high availability for iSCSI targets is disabled by default. To enable it on a cluster node, do the following:

1. If not yet done, enable HA on the node using the `hastart` script as described in the previous section.
2. Add the iSCSI shaman role to the node. For example, on a node in the cluster `vstor1`, run:

```
# shaman -c vstor1 add-role ISCSI
```

If you want HA to work only for iSCSI targets, change the shaman roles on the node by running the following command:

```
# shaman -c vstor1 set-roles ISCSI
```

8.2.3 Disabling High Availability on Nodes

Disabling HA on a node disables it for all virtual environments and iSCSI targets on this node that are stored in the Virtuozzo Storage cluster.

To disable HA on a node, do the following:

1. Disable and stop HA services:

```
# systemctl disable shaman.service
# systemctl stop shaman.service
# systemctl disable pdrs.service
# systemctl stop pdrs.service
```

2. Remove the node from the HA configuration. For example, for a node in the cluster `vstor1`, run:

```
# shaman -c vstor1 leave
```

8.3 Configuring Resource Relocation Modes

You can configure how the cluster will deal with situations when a node fails. Three modes are available:

- DRS (default; only for virtual environments). In this mode, virtual machines and containers which were running on a failed node are relocated to healthy nodes based on available RAM and license capacity. This mode can be used for nodes on which the `pdrs` daemon is running. To learn how to configure this mode, refer to [pdrs Configuration File](#).

Note: If CPU pools are used, virtual machines and containers can only be relocated to other nodes in

the same CPU pool. For details, see *Managing CPU Pools* on page 206.

The DRS mode works as follows. The master DRS continuously collects the following data from each healthy node in the cluster via SNMP:

- Total node RAM
- Total RAM used by virtual machines
- Total RAM used by containers
- Maximum running virtual machines allowed
- Maximum running containers allowed
- Maximum running virtual machines and containers allowed

If a node fails, the shaman service sends a list of virtual machines and containers which were running on that node to the master DRS that sorts it by most required RAM. Using the collected data on node RAM and licenses, the master DRS then attempts to find a node with the most available RAM and a suitable license for the virtual environment on top of the list (requiring the most RAM). If such a node exists, the master DRS marks the virtual environment for relocation to that node. Otherwise, it marks the virtual environment as broken. Then the master DRS processes the next virtual environment down the list, adjusting the collected node data by the requirements of the previous virtual environment. Having processed all virtual environments on the list, the master DRS sends the list to the shaman service for actual relocation.

- Round-robin (default fallback). In this mode, virtual machines, containers, and other resources from a failed node are relocated to healthy nodes in the round-robin manner. This mode cannot be configured. To switch to this mode, run:

```
# shaman set-config RESOURCE_RELOCATION_MODE=round-robin
```

- Spare. In this mode, virtual machines, containers, and other resources from a failed node are relocated to a spare node—an empty node with enough resources and a license to host all virtual environments from any given node in the cluster. Such a node is required for high availability to work in this mode.

Before switching to this mode, make sure the spare node is added to the HA configuration and has no resources (virtual machines, containers, iSCSI targets, and S3 clusters) stored on it. To check this, run the `shaman stat` command on any node in the cluster and check that **RESOURCES** column shows zeroes for the node:

```
# shaman stat
Cluster 'stor1'
Nodes: 3
Resources: 12

  NODE_IP      STATUS    ROLES                RESOURCES
* 10.10.20.1   Active   VM:QEMU,CT:VZ7,ISCSI,S3 0 CT, 0 S3, 0 VM, 0 ISCSI
M 10.10.20.2   Active   VM:QEMU,CT:VZ7,ISCSI,S3 4 CT, 1 S3, 3 VM, 1 ISCSI
  10.10.20.3   Active   VM:QEMU,CT:VZ7,S3       1 CT, 1 S3, 1 VM, 0 ISCSI
```

In the example above, the current node (marked by the asterisk) is empty and can be used as spare.

If the node is not empty, you can free it:

- From VMs and containers by migrating them to the other cluster nodes
- From iSCSI targets by unregistering them on said node and registering them on the other cluster nodes
- From S3 resources by releasing said node from the S3 cluster

Once you have a spare node in your cluster, you can switch to the spare mode by running:

```
# shaman set-config RESOURCE_RELOCATION_MODE=spare
```

Additionally, you can set a fallback relocation mode in case the chosen relocation mode fails. For example:

```
# shaman set-config RESOURCE_RELOCATION_MODE=drs,spare
```

In case several types of resources are to be relocated from a failed node, the default sequence `drs, round-robin` works as follows:

- Virtual machines and containers are relocated using the DRS mode, that is, according to the available RAM and license capacity on cluster nodes. If no suitable nodes are found for some virtual environments, these resources will be marked as broken and relocated to the master node in the stopped state.
- All other types of resources (iSCSI, NFS, S3) are relocated using the round-robin mode.

8.4 Configuring Resource Relocation Modes for Nodes Participating in S3 or iSCSI Export

Important: Follow instructions in this section only if you use Virtuoizzo Storage with GUI management in the scenario described below. Otherwise skip this section.

- If you use Virtuoizzo Storage with GUI management and have assigned the iSCSI network role to a node's network interface in the Virtuoizzo Storage management panel (see [Network Interface Roles](#)), high availability is automatically enabled for virtual machines, containers, and iSCSI targets on this node in the round-robin resource relocation mode.
- If you use Virtuoizzo Storage with GUI management and have joined a node to an S3 cluster in the Virtuoizzo Storage management panel (see [Creating the S3 Cluster](#)), high availability is automatically enabled for virtual machines, containers, and S3 resources on this node in the round-robin resource relocation mode.

Note: All resource relocation modes are described in [Configuring Resource Relocation Modes](#) on page 202.

If you are fine with the round-robin mode, you do not need to configure anything else on the node. If, however, you want to change the resource relocation mode, e.g., to DRS, log in to the node via SSH, and run the `hastart` script as described in [Enabling and Disabling High Availability for Nodes](#) on page 200.

After executing the script, the HA mode will automatically change to DRS. If you need to change it again, follow the instructions in [Configuring Resource Relocation Modes](#) on page 202.

8.5 Configuring HA Priority for Virtual Machines and Containers

High availability priority defines which virtual machines and containers will be relocated first if the node they are located on fails. The higher is priority, the higher is the chance a virtual machine or container has to be relocated to a healthy node, if the Virtuoizzo Storage cluster does not have enough disk resources.

By default, all newly created virtual machines and containers have the priority set to 0. You can use the `prlctl set` command to configure the default priority of a virtual machine or container, for example:

```
# prlctl set MyVM1 --ha-prio 1
# prlctl set MyVM2 --ha-prio 2
```

These commands set the HA priority for the MyVM1 and MyVM2 virtual machines to 1 and 2, respectively. If the node where these VMs are located fails, MyVM2 will be the first to relocate to a healthy node, followed by MyVM1 and then by all other virtual machines that have the default priority of 0.

8.6 Managing CPU Pools

Warning: This feature is experimental. Libvirt may not be aware of new CPU features that may already be used in CPU pools. This may lead to issues with migration to destination nodes that do not have these unreported CPU features. In addition, CPU pools only work on Intel Ivy Bridge and newer CPUs. They do not work on AMD processors.

In Virtuozzo Hybrid Server, you can avoid stopping virtual environments on a node (e.g., for node maintenance) by temporarily migrating them live to another node. For live migration to be possible, the CPUs on the source and destination nodes must be manufactured by the same vendor, and the CPU features of the destination node must be the same or exceed those of the source node.

Such a requirement may lead to two issues:

1. If the target node has more CPU features than the source node, live migration back to the source server will not be possible.
2. If a node in a high availability cluster fails and its virtual environments are relocated to another node, that destination node may have a CPU from a different vendor or with a different set of features. This will prevent live migration back to the original node when it goes up again.

CPU pools solve these two issues by dividing your Virtuozzo Hybrid Server nodes into groups (pools) in which live migration between any two nodes is always guaranteed. This is achieved by determining CPU features common for all nodes in a pool and masking (disabling) the rest of the CPU features on nodes that have more of them. So a CPU pool is a group of nodes with equal CPU features.

Note the following:

- Adding nodes with same CPUs to different CPU pools does not prevent live migration between such nodes.
- If `xsave` instructions supported by CPUs on both source and destination nodes use different buffer sizes, migration will fail.

8.6.1 Adding Nodes to CPU Pools

Note: Nodes with CPUs from different vendors cannot be added to same CPU pools.

A node that is to be added to a CPU pool must not have running virtual machines and containers on it. To meet this requirement while avoiding virtual environment downtime, you can migrate all running virtual machines and containers live to a different node (and migrate them back live after the node has been added to a pool).

The easiest way to add a node to a CPU pool is to run the following command on it:

```
# cpupools join
```

The node will be added to a default CPU pool.

Default pools have the following features and limitations:

- The naming pattern is `default_{intel}N`, e.g., `default_intel0`, `default_intel1`, etc.
- A preset, unchangeable basic CPU mask provides maximum hardware compatibility at the expense of advanced CPU features. Different CPU feature masks are used for different CPU vendors.
- Nodes which do not support the basic CPU feature mask are placed in different default CPU pools, e.g., `default_intel1`, `default_intel2`, etc.
- Nodes cannot be added to specific default CPU pools on purpose.

To make sure that as many common CPU features as possible are enabled for nodes in a pool for best performance, you can move the required nodes to a custom CPU pool. To do this:

1. On the node to be added to a custom CPU pool, run the `cpupools move` command. For example:

```
# cpupools move mypool
```

The node will be moved to the CPU pool `mypool`. If the CPU pool does not exist, it will be created.

Note: Custom CPU pools are created with the same basic CPU feature mask as default pools.

2. On any node in the new pool, run the `cpupools recal` command to update the CPU feature mask and make sure that as many common CPU features as possible are enabled. For example:

```
# cpupools recalc mypool
```

Now that node is in the desired CPU pool, you can migrate the node's virtual machines and containers back live.

The general recommendation is to group nodes with CPUs of the similar microarchitecture, generation, or family as they have similar features. This way most of the CPU features will remain available for nodes after applying the CPU feature mask to the pool. This approach will help ensure the best possible performance for nodes and at the same time guarantee live migration compatibility.

8.6.2 Monitoring CPU Pools

To see which CPU pools exist in your cluster and which nodes are in them, run the `cpupools stat` command on any node in the cluster. For example:

```
# cpupools stat
default_intel0:
320117e17894401a
bec9df1651b041d8
eaea4fc0ddb24597
mypool:
ca35929579a448db
* f9f2832d4e5f4996
```

The identifiers listed are Virtuozzo Storage node IDs which you can obtain with the `shaman -v stat` command. For example:

```
# shaman -v stat
Cluster 'vstor1'
Nodes: 5
Resources: 1
  NODE_IP      STATUS   NODE_ID      RESOURCES
  10.29.26.130 Active   bec9df1651b041d8  0 CT
* 10.29.26.134 Active   f9f2832d4e5f4996  0 CT
  10.29.26.137 Active   ca35929579a448db  0 CT
  10.29.26.141 Active   320117e17894401a  0 CT
M 10.29.26.68  Active   eaea4fc0ddb24597  1 CT
...
```

Note: The asterisk marks the current node (on which the command has been run).

8.6.3 Removing Nodes from CPU Pools

To remove the current node from a CPU pool, run the `cpupools leave` command on it:

```
# cpupools leave
```

8.7 Monitoring Cluster Status

You can use the `shaman top` and `shaman stat` commands to monitor the overall status of a cluster and cluster resources. The `shaman stat` command shows a snapshot of the current cluster status and clustering resources while `shaman top` displays a dynamic real-time view of the cluster. The following shows an example output of the `shaman stat` command:

```
# shaman stat
Cluster 'vstor1'
Nodes: 3
Resources: 8

      NODE_IP      STATUS      RESOURCES
M 10.30.24.176    Active      0 CT, 0 VM, 1 ISCSI
* 10.30.25.33     Active      1 CT, 3 VM, 1 ISCSI
  10.30.26.26     Active      0 CT, 0 VM, 2 ISCSI

      CT ID      STATE      STATUS      OWNER_IP      PRIORITY
      101      stopped    Active      10.30.25.33    0

      VM NAME     STATE      STATUS      OWNER_IP      PRIORITY
      vm1        stopped    Active      10.30.25.33    0
      vm2        running    Active      10.30.25.33    0
      vm3        stopped    Active      10.30.25.33    0

      ISCSI ID      STATE      STATUS      OWNER_IP      PRIORITY
      iqn.2014-04.com.pstorage:ps1  running    Active      10.30.24.176    0
      iqn.2014-04.com.pstorage:ps2  running    Active      10.30.25.33     0
      iqn.2014-04.com.pstorage:ps3  running    Active      10.30.26.26     0
```

The table below explains all output fields:

Field	Description
Cluster	Name of the cluster. The “M” next to the server denotes that the server is the main server in the cluster (called the master server), and the asterisk (*) indicates the server where the <code>shaman stat</code> command was executed.
Nodes	Number of servers with enabled HA support in the cluster.
Resources	Number of resources shaman keeps control of.
NODE_IP	IP address assigned to the server.
STATUS	Server status. It can be one of the following: <ul style="list-style-type: none"> • Active. The server is up and running and controlled by shaman. • Inactive. The server is up and running, but shaman does not control the server (e.g., the <code>shamand</code> service is stopped).
RESOURCES	Resources hosted on the server.
CT ID	Container ID.
VM NAME	Virtual machine name.
ISCSI ID	iSCSI target name.
STATE	Denotes whether the virtual machine/container is running or stopped.
STATUS	HA status of the virtual machine, container, or iSCSI target (as reported by shaman). It can be one of the following: <ul style="list-style-type: none"> • Active. Healthy virtual machines, containers, or iSCSI targets hosted in the cluster. • Broken. Virtual machines, containers, or iSCSI targets that could not be relocated from a failed server to a healthy one. • Pool. Virtual machines, containers, or iSCSI targets waiting for relocation from a failed server to a healthy one.
OWNER_IP	IP address of the server where the virtual machine, container, or iSCSI target is hosted.
PRIORITY	Current priority of the virtual machine/container. For details, see Configuring HA Priority for Virtual Machines and Containers on page 205.

The output of the `shaman top` command is similar to that of `shaman stat`. Additionally, you can use keyboard keys to change the command output on the fly. The following keys are supported:

- g: Group or ungroup resources by their status.
- v: Show or hide additional information.
- ENTER or SPACE: Update the screen.
- q or Esc or CTRL-C: Quit.
- h: Show the help screen.

8.8 Managing Cluster Resources with Scripts

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

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

The following example describes the process of using the `relocate` script:

1. The `shaman-monitor` daemon checks at regular intervals whether some virtual machines and containers require relocation (usually, when a server fails and the virtual machines and containers hosted on it should be relocated to a healthy server).
2. If some virtual machines or containers are scheduled for relocation, `shaman-monitor` calls the common script `/usr/share/shaman/relocate`.
3. The common script `relocate` calls the scripts `/usr/share/shaman/vm-/relocate` and `/usr/share/shaman/ct-/relocate` to relocate the virtual machines and containers to a healthy server.

If necessary, you can customize any of the scripts to meet your demands.

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

8.9 Troubleshooting Shaman Resources

The high availability feature is managed by the `shaman-monitor` service running on each cluster node. Each node's service has its own resources repository in the `<cluster_mount>/.shaman/md.<host_ID>/resources` directory.

Once initiated, `shaman-monitor` enumerates and maintains a list of all resources, that is, virtual machines and containers, on the node in its repository directory. Each resource is presented by a file in node's shaman repository that contains a path to that resource.

One of the running `shaman-monitor` services in the cluster is elected the master shaman service (and the node it runs on becomes a master shaman node). If a node fails, the shaman master starts processing failed node's shaman repository. Resources that were running on the failed node are relocated to, registered, and started on healthy nodes by their respective shaman services. Resources that were not running or could not be relocated according to the policy (see [Configuring Resource Relocation Modes](#) on page 202) are registered on the master shaman node and remain stopped.

8.9.1 Possible Issues

In certain situations that involve moving and re-registering VEs (like failed migrations or backups), shaman's list of resources may become outdated and some of the resources may not actually be where shaman expects them to be. This may cause various issues in case of HA events:

- If during a previous shaman action a command for a resource returned an error, the resource will be marked as broken and moved to `<cluster_mount>/.shaman/broken`. Broken resources are ignored during events.
- A resource may be present in shaman's repository on a node but not actually registered in it. This VE may actually be running on a different node. If an HA failover occurs on the node that has unregistered VEs, the master will try to re-register the resources on other nodes while they are already running somewhere else. VEs duplicated in such a way will not start and the hypervisor will consider them invalid.
- A virtual environment may happen to be registered and running on one node but the corresponding resource may exist in a shaman's repository on a different node. If an HA failover occurs on the node, such VEs will not be relocated, because shaman is unaware of them. In addition, if you attempt to migrate such a VE, the source node will try to pass the corresponding shaman resource to the

destination node. This will result in error since the resource is not present on the source node.

- Two shaman resources in different shaman repositories may point to the same virtual environment. As a result, VE may become duplicated and hypervisor will consider one of the duplicates invalid.
- A virtual environment's shaman resource may be missing from repositories. This is the same as disabling HA for that VE, except for migration that will result in an error after the resource is not found.

8.9.2 Shaman Resource Consistency Best Practices

Resources may become broken if the underlying storage has had no write access during an HA event. To avoid this, starting from Virtuozzo Hybrid Server 7 Update 10, when a VE restarts, shaman automatically registers it on the node where it is actually running. This also fixes any issues with VE location being different from the location of its corresponding shaman resource.

In general, you can locate any broken resources in shaman repositories by taking note of VE records marked "B" in the output of `shaman stat`. If such VEs are shown in the output of `prlctl list` on nodes where they reside, you can restart them to fix their shaman resources. Otherwise you can try to re-register all broken resources on a node by running

```
# shaman [-c <cluster_name>] cleanup-broken
```

In particular, this command cleans the shaman repository of resources with incorrect paths.

Besides that, you can verify if local resources point to local VEs by running

```
# shaman [-c <cluster_name>] sync
```

Note: Since the `shaman sync` command only fixes shaman resources, you need to manually [clean up](#) any unneeded (stopped or invalid) VE duplicates before launching `shaman sync` in your cluster.

This command updates shaman resources on the local node according to the following rules:

- If a VE is present on a node but its shaman resource is missing or located on another node, the resource is re-registered on the local node.
- If a VE is not present on a node but its resource is, this shaman resource is deleted.
- If both the VE and its shaman resource are present on a node but their parameters (e.g., resource relocation priority) differ, VE parameters overwrite resource parameters so that both are in sync.

You may want to run `shaman sync` on all nodes in the cluster to fix all of the resources.

CHAPTER 9

Hardening Virtuozzo Hybrid Server

Similar to other hypervisor-based technologies, the health of a Virtuozzo Hybrid Server server is critical to ensure the uptime of all resident virtual environments.

This chapter provides information on best practices necessary to harden a Virtuozzo Hybrid Server server and ensure its continuous uptime.

9.1 Update Policy

Keeping Virtuozzo Hybrid Server up to date is critical for the security and reliability of the system. To mitigate any potential security risks, Virtuozzo Hybrid Server software must be updated in a timely manner.

Software packages that have not been updated for a significant period of time can result in security vulnerabilities and other serious functionality issues compromising the entire system.

For more details, see *Keeping Your System Up To Date* on page 172.

9.2 Audit Policy

The audit policy defines the significant events which need to be logged on server. Logs have two important roles: provide a means for near-real-time monitoring of the system and allow you to investigate past actions. When considering system security, audit events will often identify unauthorized attempts to access resources. The events originate from interactive user sessions or system processes and services.

As defined by the Filesystem Hierarchy Standard (FHS), events are logged to files which reside in the `/var/log` directory. Files that you need to pay attention to are listed in the table:

File	Description	How to examine
<code>/var/log/lastlog</code>	Records of each user's last login	<code>lastlog</code>
<code>/var/log/messages</code>	System messages from <code>syslogd</code>	<code>cat /var/log/messages</code>
<code>/var/log/wtmp</code>	Records of all logins and logouts	<code>who /var/log/wtmp</code>

9.2.1 Storing Logs Remotely

It is recommended to store logs remotely. This will let you detect intrusion even if an attacker gained root privileges and modified local logs to hide their presence. You can change log location by configuring the `rsyslogd` daemon.

For example, you can add the following lines to the end of the `/etc/rsyslog.conf` configuration file:

```
kern.warning;*.err;authpriv.none\t@<remote_host>
*.info;mail.none;authpriv.none;cron.none\t@<remote_host>
*.emerg\t@<remote_host>
local7.*\t@<remote_host>
```

where `<remote_host>` is the FQDN of the destination server where logs need to be stored.

9.2.2 Viewing Critical Audit Messages

The most important security messages are tracked by `syslog authpriv` and stored in the `/var/log/secure` log file by default. It tracks all attempts to access the computer from a local interactive logon, network logon, network service startups, change of privileges, etc. Failed logon attempts may show a trend for password attacks. Successful logon messages are important for identifying which user logged on at a given time.

9.3 Mount Policy

The mount policy can be defined by mount options that can help you prevent unexpected usage of files. These options are listed in the table:

Option	Description
noexec	Forbid direct execution of any binaries on the mounted file system.
nodev	Do not interpret character or block special devices on the file system.
nosuid	Forbid the set-user-identifier or set-group-identifier bits to have effect.
nouser	Forbid an ordinary (non-root) user to mount the file system.

You can add these mount options to corresponding partitions in `/etc/fstab`. For example, the `noexec` option can be applied to the `/tmp` partition, while all of the above options can be applied to removable media mounts (CDROMs, DVDROMs, floppy drives, USB memory cards, etc.).

9.4 Service Policy

To be able to log in to your Virtuozzo Hybrid Server server for administration purposes, make sure that services listed in the table are enabled on the server.

Service	Description
network	Provides network connectivity for the Virtuozzo Hybrid Server server itself and virtual environments residing on it.
sshd	Most of the Virtuozzo Hybrid Server servers reside in datacenters and are managed remotely.
cron	Virtuozzo Hybrid Server uses a number of cron-based tools for periodical checking and reporting of system health parameters.
rsyslogd	System events logging.
prl-disp	Virtuozzo Hybrid Server management service.
libvirtd	Performs management tasks on virtual environments.

The following best practices apply:

sshd:

- Configure your SSH daemon to use protocol version 2.
- Prohibit remote root login as most attacks are performed to this account. Login as a non-privileged user and switch to the root credentials using `sudo` package if required.
- Prohibit authentication based on `hosts` and `rhosts` as they are known to be vulnerable.

rsyslogd:

- Do not use remote logging over UDP protocol.
- Use TCP transport and SSH tunnel for remote logging, if packets pass through an untrusted network.

pr1-disp:

- Block the remote access to pr1-disp if you do not use virtual environment migration, remote backup/restoration, or remote access to Virtuozzo Hybrid Server servers via pr1ctl or Virtuozzo Hybrid Server SDK.
- Enable encryption of all the data transmitted between management services on different nodes by running `pr1srvctl set --min-security-level high` and restarting pr1-disp. Doing this will significantly slow down virtual environment migration.

Additionally, it is recommended to have only hardware-related services running on your Virtuozzo Hybrid Server server. For example, you can run `smartd` or `snmpd` on the server, but make sure to isolate services like web or mail servers inside virtual environments in case they are attacked.

9.5 Account Policy

It is recommended to minimize the number of accounts in the host OS to make it more secure.

The general recommendations for all Linux distributions are:

- Create a non-privileged account for performing non-privileged tasks in the system.
- Use `sudo` for performing privileged tasks.
- Disable remote root logon, use a non-privileged user for this.
- Disable system user logon.
- Force periodical password changes.
- Disable accounts after a number of login failures.

9.6 Networking Policy

A Virtuozzo Hybrid Server server should be isolated from the Internet. It should only have an internal IP address and firewall policies applied to it. To still be able to update the license and install updates from Virtuozzo Hybrid Server repositories, you can set up a proxy server and follow the instructions in *Using a Proxy Server to Activate Licenses* on page 166 and *Using A Proxy Server to Install Updates* on page 174.

Virtuozzo Hybrid Server enables Linux kernel firewall during installation. To see the list of ports opened by default, refer to *Configuring Server Ports*.

CHAPTER 10

Monitoring

This chapter is about monitoring. You can monitor nodes running Virtuozzo Hybrid Server 7.5 and newer, as well as VEs hosted on them, via Zabbix or Prometheus. Additionally, you will learn how to monitor Virtuozzo Hybrid Server objects via the Simple Network Management Protocol (SNMP).

10.1 Monitoring System Objects via SNMP

You can monitor Virtuozzo Hybrid Server objects via the Simple Network Management Protocol (SNMP). The implementation conforms to the same Structure of Management Information (SMI) rules as the data in the standard SNMP context: all objects are organized in a tree; each object identifier (OID) is a series of integers corresponding to tree nodes and separated by dots.

General information:

- The OID of the root subtree with all the objects you can monitor is 1.3.6.1.4.1.26171.1.1.
- The `VIRTUOZZO-RMOND-SMI.txt` information base file is required to monitor the objects. Its default location is `/usr/share/snmp/mibs`.

The following subsections describe ways to enable and use SNMP to monitor cluster objects.

10.1.1 Enabling SNMP Access

To enable SNMP access on a hardware node, do the following:

1. Install the `rmond` package.

```
# yum install rmond
```

- If you have a Virtuozzo Storage cluster, add the following lines to the `/etc/snmp/snmpd.local.conf` file:

```
rwcommunity pdrs 127.0.0.1 .1.3.6.1.4.1.26171
rwcommunity pdrs <IP_network/subnet_mask> .1.3.6.1.4.1.26171
```

Where `<IP_network>` is the Virtuozzo Storage cluster network and `<subnet_mask>` covers all the nodes participating in the cluster.

- Make sure the `snmpd` service is running.

10.1.2 Accessing System Objects via SNMP

You can access Virtuozzo Hybrid Server objects with SNMP tools of your choice, e.g., the free Net-SNMP suite for Linux. For example, to display information on the node, do as follows:

- Install the `net-snmp-utils` package:

```
# yum install net-snmp-utils
```

- Run the following `snmpwalk` command:

```
# snmpwalk -m /usr/share/snmp/mibs/VIRTUOZZO-RMOND-SMI.txt -c public -v2c 127.0.0.1 \
.1.3.6.1.4.1.26171.1.1
```

Typical output may be the following:

```
VIRTUOZZO-RMOND-SMI::rmondVeId."{b5f09b02-7358-4998-860c-c622653859f3}" = STRING: \
{b5f09b02-7358-4998-860c-c622653859f3}
VIRTUOZZO-RMOND-SMI::rmondVeName."{b5f09b02-7358-4998-860c-c622653859f3}" = STRING: centos
VIRTUOZZO-RMOND-SMI::rmondVeState."{b5f09b02-7358-4998-860c-c622653859f3}" = INTEGER: \
running(805306372)
VIRTUOZZO-RMOND-SMI::rmondVePerfectNode."{b5f09b02-7358-4998-860c-c622653859f3}" = STRING:
VIRTUOZZO-RMOND-SMI::rmondVeMemoryTotal."{b5f09b02-7358-4998-860c-c622653859f3}" = Counter64: \
2147483648
VIRTUOZZO-RMOND-SMI::rmondVeMemoryUsage."{b5f09b02-7358-4998-860c-c622653859f3}" = Counter64: \
1788
VIRTUOZZO-RMOND-SMI::rmondVeSwapTotal."{b5f09b02-7358-4998-860c-c622653859f3}" = Counter64: 0
VIRTUOZZO-RMOND-SMI::rmondVeSwapUsage."{b5f09b02-7358-4998-860c-c622653859f3}" = Counter64: 0
VIRTUOZZO-RMOND-SMI::rmondVeCpuNumber."{b5f09b02-7358-4998-860c-c622653859f3}" = INTEGER: 2
VIRTUOZZO-RMOND-SMI::rmondVeCpuLimit."{b5f09b02-7358-4998-860c-c622653859f3}" = INTEGER: 0
VIRTUOZZO-RMOND-SMI::rmondVeCpuUnits."{b5f09b02-7358-4998-860c-c622653859f3}" = INTEGER: 0
VIRTUOZZO-RMOND-SMI::rmondVeCpuSystem."{b5f09b02-7358-4998-860c-c622653859f3}" = INTEGER: 0
VIRTUOZZO-RMOND-SMI::rmondVeCpuUser."{b5f09b02-7358-4998-860c-c622653859f3}" = INTEGER: 0
VIRTUOZZO-RMOND-SMI::rmondVeType."{b5f09b02-7358-4998-860c-c622653859f3}" = INTEGER: vm(0)
VIRTUOZZO-RMOND-SMI::rmondVeUuid."{b5f09b02-7358-4998-860c-c622653859f3}" = STRING: \
{b5f09b02-7358-4998-860c-c622653859f3}
```

```

VIRTUOZZO-RMOND-SMI::rmondVeDiskName."{b5f09b02-7358-4998-860c-c622653859f3}".2881840264.\
645796544 = STRING: /vz/vmprivate/b5f09b02-7358-4998-860c-c622653859f3/harddisk.hdd
VIRTUOZZO-RMOND-SMI::rmondVeDiskTotal."{b5f09b02-7358-4998-860c-c622653859f3}".2881840264.\
645796544 = Counter64: 0
VIRTUOZZO-RMOND-SMI::rmondVeDiskUsage."{b5f09b02-7358-4998-860c-c622653859f3}".2881840264.\
645796544 = Counter64: 0
VIRTUOZZO-RMOND-SMI::rmondVeDiskReadRequests."{b5f09b02-7358-4998-860c-c622653859f3}."\
2881840264.645796544 = Counter64: 2
VIRTUOZZO-RMOND-SMI::rmondVeDiskWriteRequests."{b5f09b02-7358-4998-860c-c622653859f3}."\
2881840264.645796544 = Counter64: 16714
VIRTUOZZO-RMOND-SMI::rmondVeDiskReadBytes."{b5f09b02-7358-4998-860c-c622653859f3}".2881840264.\
645796544 = Counter64: 77824
VIRTUOZZO-RMOND-SMI::rmondVeDiskWriteBytes."{b5f09b02-7358-4998-860c-c622653859f3}".2881840264.\
645796544 = Counter64: 1073883648
VIRTUOZZO-RMOND-SMI::rmondVeDiskHash1."{b5f09b02-7358-4998-860c-c622653859f3}".2881840264.\
645796544 = Counter32: 2881840264
VIRTUOZZO-RMOND-SMI::rmondVeDiskHash2."{b5f09b02-7358-4998-860c-c622653859f3}".2881840264.\
645796544 = Counter32: 645796544
VIRTUOZZO-RMOND-SMI::rmondVeNetworkInterface."{b5f09b02-7358-4998-860c-c622653859f3}"." = STRING:
VIRTUOZZO-RMOND-SMI::rmondVeNetworkInBytes."{b5f09b02-7358-4998-860c-c622653859f3}"." = \
Counter64: 6743089
VIRTUOZZO-RMOND-SMI::rmondVeNetworkOutBytes."{b5f09b02-7358-4998-860c-c622653859f3}"." = \
Counter64: 12472
VIRTUOZZO-RMOND-SMI::rmondVeNetworkInPackets."{b5f09b02-7358-4998-860c-c622653859f3}"." = \
Counter64: 33658
VIRTUOZZO-RMOND-SMI::rmondVeNetworkOutPackets."{b5f09b02-7358-4998-860c-c622653859f3}"." = \
Counter64: 159
VIRTUOZZO-RMOND-SMI::rmondVeNetworkMacAddress."{b5f09b02-7358-4998-860c-c622653859f3}"." = \
STRING: 001C42C329B0
VIRTUOZZO-RMOND-SMI::rmondVeVCpuOrdinal."{b5f09b02-7358-4998-860c-c622653859f3}".0 = INTEGER: 0
VIRTUOZZO-RMOND-SMI::rmondVeVCpuOrdinal."{b5f09b02-7358-4998-860c-c622653859f3}".1 = INTEGER: 1
VIRTUOZZO-RMOND-SMI::rmondVeVCpuTime."{b5f09b02-7358-4998-860c-c622653859f3}".0 = Counter64: \
5972000000
VIRTUOZZO-RMOND-SMI::rmondVeVCpuTime."{b5f09b02-7358-4998-860c-c622653859f3}".1 = Counter64: \
2672000000
VIRTUOZZO-RMOND-SMI::rmondLocalVeNumber.0 = INTEGER: 1
VIRTUOZZO-RMOND-SMI::rmondVeLimit.0 = INTEGER: 65535
VIRTUOZZO-RMOND-SMI::rmondLicenseVeNumber.0 = INTEGER: 65535
VIRTUOZZO-RMOND-SMI::rmondLicenseCtNumber.0 = INTEGER: 65535
VIRTUOZZO-RMOND-SMI::rmondLicenseVmNumber.0 = INTEGER: 65535
VIRTUOZZO-RMOND-SMI::rmondLicenseCtUsage.0 = INTEGER: 1
VIRTUOZZO-RMOND-SMI::rmondLicenseVmUsage.0 = INTEGER: 0
VIRTUOZZO-RMOND-SMI::rmondLicenseVmUsage.0 = No more variables left in this MIB View (It is past \
the end of the MIB tree)

```

10.1.3 Description of System Objects

The tables below describe objects you can monitor:

Table 10.1.3.1: Objects related to virtual environments

Object	Description
VIRTUOZZO-RMOND-SMI::rmondVeId	Virtual environment ID.
VIRTUOZZO-RMOND-SMI::rmondVeName	Virtual environment name.
VIRTUOZZO-RMOND-SMI::rmondVeState	Virtual environment state.
rVIRTUOZZO-RMOND-SMI::mondVePerfectNode	The perfect node for the virtual environment.
VIRTUOZZO-RMOND-SMI::rmondVeMemoryTotal	Total memory set for the virtual environment.
VIRTUOZZO-RMOND-SMI::rmondVeMemoryUsage	Memory usage inside the virtual environment.
VIRTUOZZO-RMOND-SMI::rmondVeSwapTotal	Total swap space set for the virtual environment.
VIRTUOZZO-RMOND-SMI::rmondVeSwapUsage	Swap space usage inside the virtual environment.
VIRTUOZZO-RMOND-SMI::rmondVeCpuNumber	Number of logical CPU cores set for the virtual environment.
VIRTUOZZO-RMOND-SMI::rmondVeCpuLimit	CPU limit set for the virtual environment.
VIRTUOZZO-RMOND-SMI::rmondVeCpuUnits	CPU units allocated to the virtual environment.
VIRTUOZZO-RMOND-SMI::rmondVeCpuSystem	CPU usage of the system processes inside the virtual environment.
VIRTUOZZO-RMOND-SMI::rmondVeCpuUser	CPU usage of the user processes inside the virtual environment.
VIRTUOZZO-RMOND-SMI::rmondVeType	Virtual environment type.
VIRTUOZZO-RMOND-SMI::rmondVeUuid	Virtual environment UUID reported by the dispatcher.

Table 10.1.3.2: Objects related to VM disks

Object	Description
VIRTUOZZO-RMOND-SMI::rmondVeDiskName	Full path to the VM hard disk.
VIRTUOZZO-RMOND-SMI::rmondVeDiskTotal	Total space of the VM disk.
VIRTUOZZO-RMOND-SMI::rmondVeDiskUsage	Used space of the VM disk.
VIRTUOZZO-RMOND-SMI::rmondVeDiskReadRequests	Read requests rate of the VM disk.

Continued on next page

Table 10.1.3.2 -- continued from previous page

Object	Description
VIRTUOZZO-RMOND-SMI::rmondVeDiskWriteRequests	Write requests rate of the VM disk.
VIRTUOZZO-RMOND-SMI::rmondVeDiskReadBytes	Read rate of the VM disk, in bytes.
VIRTUOZZO-RMOND-SMI::rmondVeDiskWriteBytes	Write rate of the VM disk, in bytes.
VIRTUOZZO-RMOND-SMI::rmondVeDiskHash1	Low-order 32 bits of the VM disk hash.
VIRTUOZZO-RMOND-SMI::rmondVeDiskHash2	High-order 32 bits of the VM disk hash.

Table 10.1.3.3: Objects related to VE network interfaces

Object	Description
VIRTUOZZO-RMOND-SMI::rmondVeNetworkInterface	Network interface name of the virtual environment.
VIRTUOZZO-RMOND-SMI::rmondVeNetworkInBytes	Incoming traffic, in bytes, received through the VE network interface.
VIRTUOZZO-RMOND-SMI::rmondVeNetworkOutBytes	Outgoing traffic, in bytes, sent through the VE network interface.
VIRTUOZZO-RMOND-SMI::rmondVeNetworkInPackets	Incoming traffic, in packets, received through the VE network interface.
VIRTUOZZO-RMOND-SMI::rmondVeNetworkOutPackets	Outgoing traffic, in packets, sent through the VE network interface.
VIRTUOZZO-RMOND-SMI::rmondVeNetworkMacAddress	MAC address of the VE network interface.

Table 10.1.3.4: Objects related to attributes

Object	Description
VIRTUOZZO-RMOND-SMI::rmondVeVCpuOrdinal	Virtual CPU ordinal numbers inside the virtual environment.
VIRTUOZZO-RMOND-SMI::rmondVeVCpuTime	Virtual CPU execution time inside the virtual environment, in nanoseconds.
VIRTUOZZO-RMOND-SMI::rmondLocalVeNumber	Number of virtual environments on the host.
VIRTUOZZO-RMOND-SMI::rmondVeLimit	Allowed number of virtual environments on the host defined by the user.

Continued on next page

Table 10.1.3.4 -- continued from previous page

Object	Description
VIRTUOZZO-RMOND-SMI::rmondLicenseVeNumber	Allowed number of virtual environments on the host defined by the license.
VIRTUOZZO-RMOND-SMI::rmondLicenseCtNumber	Allowed number of containers on the host defined by the license.
VIRTUOZZO-RMOND-SMI::rmondLicenseVmNumber	Allowed number of virtual machines on the host defined by the license.
VIRTUOZZO-RMOND-SMI::rmondLicenseCtUsage	Allowed usage of containers defined by the license.
VIRTUOZZO-RMOND-SMI::rmondLicenseVmUsage	Allowed usage of virtual machines defined by the license.

10.2 Monitoring Nodes and Virtual Environments via Prometheus

Note: The collected statistics are only intended for monitoring and are not suitable for billing purposes.

You can monitor nodes running Virtuozzo Hybrid Server 7.5 and newer as well as VEs hosted on them via Prometheus. A typical list of required components includes:

- Prometheus, a service that grabs statistics from exporters and stores it in a time series database.
- Alertmanager, a service that receives alerts from Prometheus and handles their delivery via various communication channels.
- Grafana, a service that provides a web panel with flexible dashboards and supports Prometheus as a data source.
- Exporters, services that are installed on Virtuozzo Hybrid Server nodes and export metrics via a simple HTTP server.

This guide describes how to install the exporters, configure an existing Prometheus service, and import the dashboards to an existing Grafana panel. For details on installing Prometheus, Alertmanager, and Grafana, see the respective documentation:

- <https://prometheus.io/docs/prometheus/latest/installation/>
- <https://prometheus.io/docs/alerting/latest/configuration/>
- <https://grafana.com/docs/grafana/latest/>

10.2.1 Installing the Exporters

Perform these steps on each Virtuozzo Hybrid Server 7.5 node that you want to monitor.

1. Install exporter packages:

```
# yum install node_exporter libvirt_exporter
```

2. Configure the firewall:

```
# firewall-cmd --permanent --zone=public --add-rich-rule='\
rule family="ipv4" \
source address="<prom_IP>/32" \
port protocol="tcp" port="9177" accept'
# firewall-cmd --permanent --zone=public --add-rich-rule='\
rule family="ipv4" \
source address="<prom_IP>/32" \
port protocol="tcp" port="9100" accept'
# firewall-cmd --reload
```

Where <prom_IP> is the Prometheus IP address, port 9177 is used by the libvirt exporter, and port 9100 is used by the node exporter.

It is recommended to expose the metrics only to the Prometheus server. Unrestricted access to the metrics can be a security and stability risk.

To be able to monitor Virtuozzo Storage clients, open another port. For example:

```
# firewall-cmd --permanent --zone=public --add-rich-rule='\
rule family="ipv4"\
source address="<prom_IP>/32"\
port protocol="tcp" port="9999" accept'
# firewall-cmd --reload
```

3. Launch the exporters:

```
# systemctl start node_exporter
# systemctl start libvirt-exporter
```

After setting up the exporters, on any Virtuozzo Hybrid Server 7.5 node, obtain the sample configuration, rules, and alerts for Prometheus and dashboards for Grafana:


```
# yum install vz-prometheus-cfg
```

The files will be placed in `/usr/share/vz-prometheus-cfg/`. For example:

```
# tree /usr/share/vz-prometheus-cfg/
/usr/share/vz-prometheus-cfg/
├── alerts
│   ├── vstorage-alerts.yml
│   └── vz-alerts.yml
├── dashboards
│   ├── grafana_hn_dashboard.json
│   ├── grafana_ve_dashboard.json
│   ├── grafana_win_ct_hn_dashboard.json
│   └── grafana_win_ct_ve_dashboard.json
├── prometheus-example.yml
├── rules
│   ├── vstorage-rules.yml
│   ├── vz-rules.yml
│   └── win_ct-rules.yml
└── targets
    ├── targets-example.yml
    └── vstorage-targets-example.yml
```

10.2.2 Configuring Prometheus

You will need to configure Prometheus so it can start collecting metrics from Virtuozzo Hybrid Server nodes. To do this, modify `prometheus.yml` based on the sample `prometheus-example.yml` shipped with `vz-prometheus-cfg`.

1. Copy the rule and alert files to the Prometheus server and set their paths in `rule_files` in `prometheus.yml` (see the example further).
2. Create target files that contain information about exporters you want to scrape. By using multiple target files you can group nodes by attributes like `datacenter`, `cluster`, and such. The following examples create a server group **cluster1** populated with five nodes and scrape their node and libvirt exporters, respectively:

```
# cat cluster1-libvirt.yml
- labels:
  group: cluster1
  targets:
  - hn01.cluster1.tld:9177
  - hn02.cluster1.tld:9177
  - hn03.cluster1.tld:9177
  - hn04.cluster1.tld:9177
  - hn05.cluster1.tld:9177
```

```
# cat cluster1-nodes.yml
- labels:
  group: cluster1
  targets:
  - hn01.cluster1.tld:9100
  - hn02.cluster1.tld:9100
  - hn03.cluster1.tld:9100
  - hn04.cluster1.tld:9100
  - hn05.cluster1.tld:9100
```

If these nodes are in the Virtuozzo Storage cluster, create a dedicated target file to be able to monitor the Virtuozzo Storage clients as well. For example:

```
# cat cluster1.yml
- labels:
  group: cluster1
  targets:
  - hn01.cluster1.tld:9999
  - hn02.cluster1.tld:9999
  - hn03.cluster1.tld:9999
  - hn04.cluster1.tld:9999
  - hn05.cluster1.tld:9999
```

3. Set paths to target files in `scrape_configs` in `prometheus.yml` (see the example further).

The Virtuozzo Storage job must be named `fused`.

A complete Prometheus configuration file may look like this:

```
# cat prometheus.yml
global:
  scrape_interval: 1m
  evaluation_interval: 1m
alerting:
  alertmanagers:
  - static_configs:
    - targets:
      - localhost:9093
rule_files:
  - /prometheus-<version>linux-amd64/rules/vz-rules.yml
  - /prometheus-<version>linux-amd64/rules/vstorage-rules.yml
  - /prometheus-<version>linux-amd64/alerts/vz-alerts.yml
  - /prometheus-<version>linux-amd64/alerts/vstorage-alerts.yml
scrape_configs:
  - job_name: 'prometheus'
    static_configs:
    - targets: ['localhost:9090']
  - job_name: libvirt
    relabel_configs:
    - source_labels: [__address__]
      target_label: instance
```

```

    regex: (.*)[:].+
file_sd_configs:
  - files:
    - /prometheus-<version>linux-amd64/targets/cluster1-libvirt.yml
- job_name: node
  relabel_configs:
  - source_labels: [__address__]
    target_label: instance
    regex: (.*)[:].+
  file_sd_configs:
  - files:
    - /prometheus-<version>linux-amd64/targets/cluster1-nodes.yml
- job_name: fused
  relabel_configs:
  - source_labels: [__address__]
    target_label: instance
    regex: (.*)[:].+
  file_sd_configs:
  - files:
    - /prometheus-<version>linux-amd64/targets/cluster1.yml

```

After editing the Prometheus configuration file, restart the prometheus and alertmanager services.

To enable monitoring of Virtuozzo Storage clients listed in the target file:

1. Adjust fstab on each node, using the previously chosen port. For example:

```

# cat /etc/fstab | grep ^vstorage
vstorage://cluster1 /vstorage/cluster1 fuse.vstorage defaults,_netdev,prometheus=0.0.0.0:9999 0 0

```

1. Stop all virtual environments and re-mount the Virtuozzo Storage file system on each node, one after another, so only one node is down at any given moment.

```

# umount /vstorage/stor1
# mount /vstorage/stor1

```

Then start all virtual environments once again.

Alternatively, reboot each node one after another, so only one node is down at any given moment.

Virtual environments are set to start on node reboots by default.

10.2.3 Configuring Grafana

To see the data collected from the nodes in Grafana, do the following in the Grafana web panel:

1. If you have not already done so, add the configured Prometheus server as a data source:

- 1.1. Navigate to **Configuration** -> **Data Sources**.

- 1.2. Click **Add data source**, select **Prometheus**.
 - 1.3. Enter a name for the data source.
 - 1.4. Specify the Prometheus IP address and port.
 - 1.5. Click **Save & Test**.
2. Import Virtuozzo Hybrid Server dashboards. Perform these steps for each JSON file shipped with vz-prometheus-cfg.
 - 2.1. Navigate to **Dashboards -> Manage**.
 - 2.2. Click **Import** and **Upload JSON file**. Select a JSON file with a Grafana dashboard.
 - 2.3. Select the previously configured Prometheus data source.
 - 2.4. Click **Import**.

The Virtuozzo Hybrid Server dashboards are now available in **Dashboards Home**.

10.2.4 Supported Alerts

The following alerts are supported for Virtuozzo Hybrid Server.

Alert	Severity	Description	What to do
nodeTcpListenDrops	Error	A large amount TCP packets have been dropped on the node.	Inspect the network traffic for issues and fix them.
nodeTcpRetransSegs	Error	A large amount of TCP packets have been retransmitted on the node.	
nodeOutOfMemory	Error	The node has run out of memory.	Find out what has been consuming memory on the node and fix it. If the node is overloaded due to overselling, migrate some of its virtual machines to other nodes.
nodeOutOfSwap	Critical	The node has run out of swap memory.	

Table 10.2.4.1 -- continued from previous page

Alert	Severity	Description	What to do
nodeHighMemoryAllocationLatency	Warning	Node's memory allocation latency is too high. The node may be overloaded, resulting in unpredictable delays in operations.	
nodeRxChecksummingDisabled	Error	The rx-checksumming feature is disabled on the network interface.	These features are enabled by default. If they have been manually disabled, re-enable them for the network interface.
nodeTxChecksummingDisabled	Error	The tx-checksumming feature is disabled on the network interface.	
nodeScatterGatherDisabled	Error	The scatter-gather feature is disabled on the network interface.	
nodeTCPSegmentationOffloadDisabled	Error	The tcp-segmentation-offload feature is disabled on the network interface.	
nodeGenericSegmentationOffloadDisabled	Error	The generic-segmentation-offload feature is disabled on the network interface.	
nodeVzLicenseInactive	Critical	Node's Virtuozzo license is inactive.	Check and update the license.

Continued on next page

Table 10.2.4.1 -- continued from previous page

Alert	Severity	Description	What to do
guestPausedEIO	Critical	A virtual machine has been paused due to a disk I/O error.	Make sure that the node's partition where the VM's disks are stored has not run out of space. If it has, free up more space for the VM. If there is enough free space, evacuate the virtual machine (and any other critical data) from the physical disk it is stored on. Replace the physical disk as it is about to fail.
guestOsCrashed	Critical	A virtual machine has crashed after a BSOD or kernel panic in the guest OS. Collected for VMs only.	Find out the reasons for the crash, fix the VM, and restart any services that will not do so automatically.
nodeSMARTDiskError	Critical	A S.M.A.R.T counter for a node's disk is below threshold.	The node's disk is about to fail. Replace it as soon as possible.
nodeSMARTDiskWarning	Warning	A S.M.A.R.T counter for a node's disk is greater than zero.	Inspect the health of node's disk. You may need to replace it soon.
highCPUusage	Warning	Virtual environments's CPU usage has been over 90% for the last 10 minutes. The alert is disabled by default.	Check the virtual environment for potential problems, including software issues or malware. These alerts are disabled by default. To enable any of them, uncomment them in <code>vz-alerts.yml</code> and restart Prometheus.

Continued on next page

Table 10.2.4.1 -- continued from previous page

Alert	Severity	Description	What to do
highMemUsage	Warning	Virtual environments's memory usage has been over 95% for the last 10 minutes. The alert is disabled by default.	
cpuUsageIncrease	Warning	Virtual environments's CPU usage has greatly increased compared to the previous week. The alert is disabled by default.	
memUsageIncrease	Warning	Virtual environments's memory usage has greatly increased compared to the previous week. The alert is disabled by default.	
nodeHighDiskWriteLatency	Warning	Write operation latency for a node's disk has been too high for the last 10 seconds.	Find and fix the reason why the I/O requests have been taking so long. Reasons can be high I/O load or deterioration of the disk's health.
nodeHighDiskReadLatency	Warning	Read operation latency for a node's disk has been too high for the last 10 seconds.	
pendingKernelReboot	Warning	The node has been updated but not rebooted to the latest kernel.	Reboot the node and switch to the latest kernel.

Continued on next page

Table 10.2.4.1 -- continued from previous page

Alert	Severity	Description	What to do
lowPageCache	Warning	The node has high load average and very small page cache. The node is overloaded, possibly due to memory overcommitment.	Find out why the node is overloaded. If the node is overloaded due to overselling, migrate some of its virtual environments to other nodes.
highPfcacheUsage	Warning	Node's pfcache disk is 90% full and may run out of space.	Add more space to the pfcache disk or clean it up. For details, see the Knowledge Base .
highVstorageMountLatency	Warning	The latency of vstorage-mount requests on a node has been too high for the last 10 seconds.	Check the health of the Virtuozzo Storage and its components. Fix the found issues.
slowIoRequest	Info	Virtual machine's I/O requests have been taking longer than 10 seconds.	Inspect the storage, be it local disks or Virtuozzo Storage, for potential issues and fix them.
unresponsiveBalloonDriver	Info	Virtual machine's balloon driver is not responding. The VM has stopped reporting its memory usage statistics and is not releasing node's memory automatically.	Find out what has happened to the virtio_balloon kernel module inside the VM. Reload the module.

10.3 Monitoring Nodes and Virtual Environments via Zabbix

Note: The collected statistics are only intended for monitoring and are not suitable for billing purposes.

You can monitor nodes with Virtuozzo Hybrid Server 7.5 and newer as well as running VEs on them via Zabbix.

This guide describes how to install the Zabbix agent on a Virtuozzo Hybrid Server node and configure an existing Zabbix server. For details on installing Zabbix, see its [documentation](#).

10.3.1 Installing the Zabbix Agent

Perform these steps on each Virtuozzo Hybrid Server 7.5 node that you want to monitor.

1. Install the Zabbix repository. For example:

```
# rpm -Uvh https://repo.zabbix.com/zabbix/5.0/rhel/7/x86_64/\
zabbix-release-5.0-1.el7.noarch.rpm
# yum clean all
```

2. Install the agent package:

```
# yum install zabbix-agent
```

3. Configure the Zabbix agent by editing `/etc/zabbix/zabbix_agentd.conf` as follows:

```
Server=<zabbix_server_IP>
ListenIP=<listen_IP>
ServerActive=<zabbix_server_IP>
Hostname=<hostname>
Timeout=30
AllowRoot=1
```

Where:

- `Server` and `ServerActive` are the Zabbix server IP address.
- `ListenIP` is a node's IP address that will listen for incoming connections from the Zabbix server.
- `Hostname` is a unique name that must match the name you will use to register the host in the Zabbix web panel. It is recommended to use node's actual hostname.
- `Timeout` is increased to allow a heavily loaded node enough time to send the statistics data to the Zabbix server.
- `AllowRoot=1` is required to let the user `zabbix` collect statistics from the node.

4. Add the user `zabbix` to `sudoers`. For example:

```
# visudo
zabbix ALL=(ALL) NOPASSWD: ALL
```

5. Configure node's firewall:

```
# firewall-cmd --zone=public --permanent --add-port=10050-10051/tcp
# firewall-cmd --reload
```

6. Install the Virtuozzo module and template:

```
# yum install vz-zabbix-agent
```

The package will supply the following Zabbix XML template:

```
# rpm -ql vz-zabbix-agent | grep xml
/etc/zabbix/templates/zbx_virtuozzo_template.xml
```

You will need to import it to the Zabbix web panel later.

7. Enable and start the Zabbix agent:

```
# systemctl enable zabbix-agent
# systemctl start zabbix-agent
```

10.3.2 Configuring Zabbix Server

To see the data collected from the nodes, do the following in the Zabbix web panel:

1. Navigate to **Configuration** -> **Templates**. Click **Import** and import `zbx_virtuozzo_template.xml` obtained in the previous section.
2. Navigate to **Configuration** -> **Hosts** and click **Create Host**. In **Host name** enter the same name as specified in `Hostname` in `zabbix_agentd.conf`. Next, select a **Group** and provide agent network interface details in **Interfaces**. Specify other details if needed and click **Add**.
3. Navigate to **Configuration** -> **Hosts** -> <host> -> **Templates**. Find **Template Virtuozzo** in **Link new templates** and click **Update**.

If you also want to use S.M.A.R.T. triggers provided by Virtuozzo, instead of or together with the standard ones, link **Template Virtuozzo SMART** as well.

The statistics for the host will become available in the Zabbix web panel in a short while.

Optionally, you can filter out unnecessary metrics by excluding ploop mounts from mounted file system discovery. Zabbix performs it every hour by default in its Linux server template. Moreover, you can safely

filter out **vme** network interfaces of virtual machines. Do the following:

1. Navigate to **Configuration -> Templates -> Template OS Linux by Zabbix agent -> Macros -> Inherited and template macros**.

2. Locate the macro `{$VFS.FS.FSNAME.NOT_MATCHES}`. Add `/vz/root/|` to its start. For example:

```
^(/vz/root/|/dev|/sys|/run|/proc|.+/shm$)
```

Newly discovered plops will be filtered out from now on. The statistics for the already discovered plops will be dropped when the grace period is over. The grace period is 30 days by default. You can change it in the file system discovery rule.

3. Locate the macro `{$NET.IF.IFNAME.NOT_MATCHES}` and add `^vme[0-9a-z]+|` to its start. For example:

```
(^vme[0-9a-z]+|^Software Loopback Interface|^NULL[0-9.]*$|^[Ll]o[0-9.]*$|^
[ss]ystem$|^Nu[0-9.]*$|^veth[0-9a-z]+|^docker[0-9]+|^br-[a-z0-9]{12})
```

Now newly discovered **vme** interfaces will be filtered out as well.

10.3.3 Supported Triggers

The following triggers (i.e. alerts) are supported for Virtuozzo Hybrid Server.

Trigger	Priority	Related metric	What to do
License is not active.	Disaster	virtuozzo.host.license_status	Check and update the license.
A S.M.A.R.T. metric is critically low. The disk is about to fail. S.M.A.R.T. metrics watched: <ul style="list-style-type: none"> • Command_Timeout • Current_Pending_Sector • End-To-End_Error • Offline_Uncorrectable • Reallo- cated_Event_Count • Reallocated_Sector_Ct • Spin_Retry_Count 	Disaster	virtuozzo.smart.discovery	The node's disk is about to fail. Replace it as soon as possible.

Continued on next page

Table 10.3.3.1 -- continued from previous page

Trigger	Priority	Related metric	What to do
<p>A S.M.A.R.T. metric is greater than zero. The disk health deteriorates.</p> <p>S.M.A.R.T. metrics watched:</p> <ul style="list-style-type: none"> • Current_Pending_Sector • Offline_Uncorrectable • Reallocated_Event_Count • Reallocated_Sector_Ct 	Warning		Inspect the health of node's disk. You may need to replace it soon.
Virtual machine's or container's memory usage has been over 95% for the last 1 hour.	Warning	virtuozzo.ct.memory_used virtuozzo.vm.memory_used	Check the virtual environment for problems, including software issues or malware.
Guest OS crashed in a virtual machine.	High	virtuozzo.vm.status	Find out the reasons for the crash, fix the virtual machine, and restart any services that will not do so automatically.
Virtual machine's balloon driver is not responding. The VM has stopped reporting its memory usage statistics and is not releasing node's memory automatically.	Info	virtuozzo.vm.memory-_upd_timestamp	Find out what has happened to the virtio_balloon kernel module inside the VM. Reload the module.

10.3.4 Managing Disk I/O Parameters

Note that I/O and IOPS limiting only works for supported I/O schedulers. The scheduler type is stored in `/sys/block/<device>/queue/scheduler`. For example:

```
# cat /sys/block/sda/queue/scheduler
noop deadline [cfq]
```

The I/O scheduler used is marked by square brackets. In this example, it is `cfq`.

The following I/O schedulers are supported:

- CFQ. A separate block device line with counters is added to the `iostat` file. For example:

```
# cat /proc/bc/100/iostat
flush 100 . 0 0 0 0 0 7389 1893968 0 0
fuse 100 . 0 0 0 0 0 0 0 0 0
sda 100 . 0 0 0 9000 1843380 245216 55845488 245028 188
```

- Deadline. The counters are added to the values in the `flush` line.

I/O and IOPS limiting does not work for devices with the `noop` I/O scheduler or without one (e.g., logical devices, CEPH RBD devices, and such).

CHAPTER 11

Customer Experience Program

This chapter explains the use of the Customer Experience Program (CEP) and provides instructions on how to install it. The program helps improve Virtuozzo Hybrid Server products to fit your needs better.

11.1 Participating in Customer Experience Program

By participating in the Customer Experience Program (CEP) you agree to send to Virtuozzo Hybrid Server information about configuration of your physical server and virtual environments, the way you use Virtuozzo Hybrid Server products, and technical issues that you encounter.

Note: No private information like your name, e-mail address, phone number, or keyboard input will be collected.

The program is voluntary and helps improve Virtuozzo Hybrid Server products to better fit your needs.

When installing Virtuozzo Hybrid Server in the attended mode, you automatically join CEP. You can, however, opt out of the program at any time by stopping and disabling the `disp-helper` service:

```
# systemctl stop disp-helper
# systemctl disable disp-helper
```

By default, Virtuozzo Hybrid Server will collect information once a week, although you can change this interval. For example, to have the data collected every two weeks, do as follows:

1. In the configuration file `/etc/vz/disp_helper.json`, change the `report_period` value to `14d`:

```
"report_period": "14d",
```

2. Restart disp-helper to apply changes:

```
# systemctl restart disp-helper
```

To fine-tune which information about your physical server or virtual environments is collected, you can disable or enable corresponding .py scripts in the /usr/share/virtuozzo/cep-scripts directory. For example, to prevent collection of data about your virtual environments, you can disable the libvirt_guests.py script as follows:

```
# chmod a-x libvirt_guests.py
```

To re-enable the script, run

```
# chmod a+x libvirt_guests.py
```

When installing Virtuozzo Hybrid Server in the unattended mode, you can specify the cep parameter in the kickstart file. For more details, see [Standard Kickstart Options](#).

CHAPTER 12

Advanced Tasks

This chapter collects miscellaneous configuration and management tasks, some of which require a deeper knowledge of Linux and Virtuozzo Hybrid Server and should be performed with caution.

12.1 Configuring Automatic Memory Management Policies

Virtuozzo Hybrid Server can automatically manage VM and container memory. It uses the memory management service VCMMD that does the following:

- Manages kernel same-page merging (KSM) for VMs
- Tunes huge pages transparently
- Balances containers and VMs between NUMA nodes
- Adjusts memory limits and guarantees for containers, VMs, and Virtuozzo Storage services

All of these optimizations are applied by means of memory management policies. Setting the correct policy may provide performance and density benefits to your Virtuozzo Hybrid Server installation.

Two policies are available:

- **density** (default): A universal policy whether memory is overcommitted or not. It provides better density than the **performance** policy. It also provides higher performance if memory is overcommitted or the same performance if it is not. This policy does the following:
 - Sets VM memory guarantees to 40%, allowing more memory to be overcommitted (see *Configuring Virtual Machine Memory Guarantees* on page 130). The container memory guarantees remain at 0%

(see *Configuring Container Memory Guarantees* on page 126).

- Binds VMs and containers to server's NUMA nodes. Rebalances them based on their memory and CPU usage when the difference in loads on the busiest and freest nodes exceeds 50%.

Note: This policy is set automatically if a license is not installed or has expired.

- **performance:** Best for servers with virtual environments that can fit into NUMA nodes. This policy does the following:
 - Sets VM memory guarantees to 80% (see *Configuring Virtual Machine Memory Guarantees* on page 130). The container memory guarantees remain at 0% (see *Configuring Container Memory Guarantees* on page 126).
 - Binds VMs and containers to server's NUMA nodes. Rebalances them between NUMA nodes every 5 minutes, based on each node's free RAM and CPU load as well as each VE's CPU and RAM usage. Doing so enables the benefits of the NUMA architecture.

Important: This policy does not allow overcommitting memory.

You can switch policies on a server as follows:

Note: It is recommended that you stop all virtual environments on the server or temporarily live-migrate them to another server for all KSM settings to take effect.

1. Change the policy and check the result using the following commands:

```
# prlsrvctl set --vcmmmd-policy density
# prlsrvctl info | grep "policy"
Vcmmmd policy: density config=density
```

2. If you stopped the virtual environments or migrated them to another server, start the virtual environments or migrate them back to the server.

12.1.1 Restarting VCMMD

In certain situations, you may need to restart the VCMMD daemon. For example:

- When you activate a license after installing Virtuozzo Hybrid Server or update an expired license.
- To permanently apply custom limits and guarantees for services residing in memory slices. See *Managing Host Services with VCMMD* on page 245 and *Managing Virtuozzo Storage Services with VCMMD* on page 248.
- If you deploy Virtuozzo Storage on top of Virtuozzo Hybrid Server and start its services manually without rebooting the server. See *Managing Virtuozzo Storage Services with VCMMD* on page 248.
- To apply changes after tweaking the VCMMD configuration file. See *Managing Virtuozzo Storage Services with VCMMD* on page 248.

If virtual environments are running on the server, do the following:

1. Stop all virtual environments on the server or temporarily live-migrate them to another server.
2. Restart `vcmmmd`:

```
# systemctl restart vcmmmd
```

The default **performance** policy will be enabled.

3. Start the virtual environments or live-migrate them back to the server, depending on what you did in step 1.

If no virtual environments are running on the server, just restarting the service is enough.

12.1.2 Optimizing Virtual Machine Memory Usage with Kernel Same-Page Merging

To optimize memory usage by virtual machines, Virtuozzo Hybrid Server uses a Linux feature called kernel same-page merging (KSM). The KSM daemon periodically scans memory for pages with identical content and merges those into a single page. That page is marked as copy-on-write (COW). When a virtual machine needs to change that page's contents, the kernel creates a copy of the page for that VM and changes it.

KSM enables the server to:

- Avoid swapping

- Run more virtual machines
- Overcommit virtual machine memory
- Speed up RAM and hence certain applications and guest operating systems

In Virtuozzo Hybrid Server 7, pages are not merged by default. In general, merging starts when the amount of free RAM drops below 20% and stops when free RAM goes above this threshold again. It works differently, however, depending on the policy:

- With the **density** policy enabled, merging works as described on server's entire RAM, disregarding NUMA nodes.
- With the **performance** policy enabled, pages are merged independently within each NUMA node. This slows KSM by as many times as there are NUMA nodes on the server (e.g., by four times if there are four NUMA nodes). Unlike **density**, however, pages are unmerged as soon as they can fit into RAM again, leaving 20% of it free. This is done to avoid COW operations and increase server performance.

KSM is enabled by default. If you need to disable it for some reason, set KSM to `false` in `/etc/vz/vcmmmd.conf`. Add the parameter if needed.

```
"LoadManager": {
  "Controllers": {
    "KSM": false,
    <...>
  },
}
```

Restart `vcmmmd` to apply changes. See [Restarting VCMMD](#) on page 244.

12.1.3 Managing Host Services with VCMMD

You can set memory limits and guarantees managed by `vcmmmd` for host services. The daemon will take them into account when allocating resources for virtual environments.

For example, to have a service named `service` managed by `vcmmmd`, do as follows:

1. Place the service processes in `/sys/fs/cgroup/memory/service.slice` directory.
2. Register `service.slice` in `vcmmmd`, specifying the desired guarantee and limit in bytes (you can also use `K` for kibibytes, `M` for mebibytes, or `G` for gibibytes). For example:

```
# vcmmctl register SRVC service.slice --guarantee 100000000 --limit 100000000 --swap 100000
```

These settings will last until a server reboot. A way to set permanent limits is provided further in this

section.

3. Activate the service:

```
# vcmmctl activate service.slice
```

List memory slices to check that the service is managed with `vcmm`. For example:

```
# vcmmctl list
name                type  active  guarantee  limit  swap
<...>
service.slice      SRVC   yes     97656      97656  97
```

Note: This command outputs values in kibibytes.

You can also temporarily change the service memory parameters at runtime using the `vcmmctl update` command.

For example, to change the `service.slice` parameters, run:

```
# vcmmctl update service.slice --guarantee 200000000 --limit 200000000
```

List memory slices to check that the parameters have been applied. For example:

```
# vcmmctl list
name                type  active  guarantee  limit  swap
<...>
service.slice      SRVC   yes     195312     195312  max
```

To unregister a service from `vcmm` at runtime, use the `vcmmctl unregister` command.

Guarantees and limits set using the instructions above will be removed on host reboot. You can, however, set permanent limits and guarantees for services in `system` and `user` memory slices. For example, to make sure that you can still access the host via SSH if it becomes heavily overcommitted. Do as follows:

1. If needed, dump the current values to the VCMMD configuration file:

```
# vcmmctl config -f > /etc/vz/vcmm.conf
```

2. Specify the minimum and maximum values in bytes in `/etc/vz/vcmm.conf` (-1 stands for unlimited memory). For example:

```
{
  "VStorage": {
    <...>
  },
}
```

```

"System": {
  "Path": "system.slice",
  "Limit": {
    "Max": -1,
    "Min": 0,
    "Share": 0.7
  },
  "Guarantee": {
    "Max": 1073741824,
    "Min": 536870912,
    "Share": 0.25
  },
  "Swap": {
    "Max": 0,
    "Min": 0,
    "Share": 0
  }
},
"User": {
  "Path": "user.slice",
  "Limit": {
    "Max": -1,
    "Min": 0,
    "Share": 0.7
  },
  "Guarantee": {
    "Max": 536870912,
    "Min": 268435456,
    "Share": 0.25
  },
  "Swap": {
    "Max": 0,
    "Min": 0,
    "Share": 0
  }
}
}
}

```

Note: The file `/etc/vz/vstorage-limits.conf` is deprecated and will be dropped in the coming releases.

3. Restart `vcmmmd` to apply changes. See [Restarting VCMMD](#) on page 244.
4. Check that the parameters were applied. The actual guarantees and limits are calculated based on the minimum and maximum values specified in `/etc/vz/vcmmmd.conf`.

```

# vcmmctl list
name                type active guarantee  limit  swap
system.slice        SRVC  yes   1048576 5605796  0
user.slice           SRVC  yes   524288  5605796  0

```

```
<...>
```

Note: This command outputs values in kibibytes.

12.1.3.1 Adjusting the Page Cache Limit for `user.slice`

In addition to the memory limits and guarantees for host services, you can set the overall page cache limit for `user.slice`. If the node's file system is actively used, doing so may increase read/write performance at the expense of virtual environments' RAM speed. The caveat is that the optimal value has to be determined experimentally.

By default, the `user.slice` page cache limit is set to whichever value is smaller: 10% of RAM or 10 GB.

To set a different limit, change the `UserCacheLimitTotal` parameter in `/etc/vz/vcmmmd.conf`. It will adjust the `/sys/fs/cgroup/memory/user.slice/memory.cache.limit_in_bytes` value.

Do the following:

1. Dump the current values to the VCMMD configuration file:

```
# vcmmddctl config -f > /etc/vz/vcmmmd.conf
```

2. Add `UserCacheLimitTotal` to the `LoadManager` section. For example, to set it to 20 GiB, in bytes:

```
"LoadManager": {
  <...>
  "UserCacheLimitTotal": 21474836480
},
```

3. Restart `vcmmmd` to apply changes. See [Restarting VCMMD](#) on page 244.

12.1.4 Managing Virtuozzo Storage Services with VCMMD

On servers running Virtuozzo Storage, `vstorage.slice/vstorage-services.slice` is automatically created in the memory cgroup on server start.

If, however, you manually deployed Virtuozzo Storage on top of Virtuozzo Hybrid Server and started the `vstorage*` services without rebooting the server, restart `vcmmmd` to create this memory slice. Consult [Restarting VCMMD](#) on page 244.

List memory slices to check that Virtuozzo Storage services are managed by VCMMD. For example:

```
# vcmmctl list
name                type  active  guarantee  limit  swap
<...>
vstorage.slice/vstorage-services.slice  SRVC   yes    4194304 103854973  0
```

Note: This command outputs values in kibibytes.

You can temporarily change memory limits and guarantees for Virtuozzo Storage services at runtime using the `vcmmctl update` command. For example:

```
# vcmmctl update vstorage.slice/vstorage-services.slice --guarantee 100000000 --limit 100000000
```

The parameters will be reset to the default values on the next `vcmm` restart or node reboot.

To configure limits and guarantees for Virtuozzo Storage services permanently, do as follows:

1. If needed, dump the current values to the VCMMD configuration file:

```
# vcmmctl config -f > /etc/vz/vcmm.conf
```

2. Specify the values in bytes in `/etc/vz/vcmm.conf` (-1 stands for unlimited memory). For example:

```
{
  "VStorage": {
    "Path": "vstorage.slice/vstorage-services.slice",
    "Limit": {
      "Max": -1,
      "Min": 0,
      "Share": 0.7
    },
    "Guarantee": {
      "Max": 1073741824,
      "Min": 536870912,
      "Share": 0.25
    },
    "Swap": {
      "Max": 0,
      "Min": 0,
      "Share": 0
    }
  },
  "System": {
    <...>
  },
  "User": {
    <...>
  }
}
```

```
}
}
```

- Restart `vcmmmd` to apply changes. See *Restarting VCMMD* on page 244.

List memory slices to check that the parameters were applied. The actual guarantees and limits are calculated based on the minimum and maximum values specified in `/etc/vz/vcmmmd.conf`.

```
# vcmmmdctl list
name                               type active guarantee  limit  swap
<...>
vstorage.slice/vstorage-services.slice  SRVC  yes  1048576  5605796  0
```

Note: This command outputs values in kibibytes.

If you use large CS journals, you may also attempt to improve storage write performance by tweaking the VCMMD configuration.

Initially, storage services can use two thirds of host's memory. As soon as you create a virtual environment, the memory available to storage services becomes limited to 512MB per CS, which may throttle their performance. You can, however, let these services use more of host's RAM by adjusting the `StorageCacheLimitTotal` value. In general, set it to at least 4 GB per CS with a journal.

For example, if you have 10 CSEs with journals, set the parameter to 40 GiB by adding it to the `LoadManager` section of `/etc/vz/vcmmmd.conf`:

```
"LoadManager": {
  "Controllers": {
    <...>
    "StorageCacheLimitTotal": 42949672960,
    <...>
  },
  <...>
},
```

Restart `vcmmmd` to apply changes. See *Restarting VCMMD* on page 244.

Check that the changes were applied:

```
# vcmmmdctl config | grep StorageCacheLimitTotal
"StorageCacheLimitTotal": 42949672960,
```

Other parameters that you may see in the `Controllers` section of `/etc/vz/vcmmmd.conf` include:

- NUMA, which switches balancing of virtual environments between NUMA nodes. It is only used when the

performance policy is enabled.

- `StoragePolicy`, which is enabled if the node is in a Virtuozzo Storage cluster. It adjusts swappiness, manages memory limits for Virtuozzo Storage services mentioned above, and protects Virtuozzo Storage services from the OOM killer.

Both of these parameters are set automatically and need not be tweaked.

12.2 Managing Dynamic Mitigation of Intel CPU Vulnerabilities

Starting from Virtuozzo Hybrid Server 7.5, mitigations for a number of Intel CPU vulnerabilities are enabled and disabled in real time based on host workload. The vulnerabilities include the ones that, if left unmitigated, may allow a malicious actor inside a VM or container to run arbitrary code on the host or in other virtual environments on that host. As these mitigations may reduce host performance, it makes sense to disable them when virtual environments are not running or when the host is only used for Virtuozzo Storage and VEs are not needed at all.

In general, mitigations are managed by kernel flags provided by microcode updates or kernel patches. For a mitigation to be enabled, one or more flags need to be set to certain values that depend on a CPU.

Virtuozzo Hybrid Server comes with all the applicable microcode updates and kernel patches. All supported mitigations are enabled by default. Dynamic mitigation management is also enabled by default (see the end of this section). It works as follows:

1. When a host boots, the memory management daemon `vcmmmd` saves the state of enabled mitigations by saving the values of these kernel flags from `/sys/kernel/debug/x86/:`
 - `pti_enabled`, Page Table Isolation, addresses [Meltdown](#)
 - `ibrs_enabled`, Indirect Branch Restricted Speculation, addresses [Spectre-V2](#)
 - `retp_enabled`, Retpolines, addresses [Spectre-V2](#)
 - `ibpb_enabled`, Indirect Branch Prediction Barriers, addresses [Spectre-V2](#)

Note: This flag is now read-only. It is set when either `ibrs_enabled` or `retp_enabled` is set.

- `ssbd_enabled`, Speculative Store Bypass Disable, addresses [Spectre-NG-V4](#)
2. If no virtual environments are running, `vcmmmd` sets the flags to 0 to disable mitigations and speed up the node.
 3. As soon as the first VM or container starts, `vcmmmd` restores the original state of the flags to enable mitigations.
 4. As soon as the last VM or container stops, `vcmmmd` sets the flags to 0 to disable mitigations and speed up the node.

Dynamic mitigations management is controlled by the `EnableMitigationsManagement` parameter in `/etc/vz/vcmmmd.conf`, which is set to `true` by default:

```
# vcmmmdctl config -f > /etc/vz/vcmmmd.conf
# cat /etc/vz/vcmmmd.conf | grep EnableMitigationsManagement
"EnableMitigationsManagement": true,
```

You can disable this feature by setting the parameter to `false` and restarting VCMMD (see [Restarting VCMMD](#) on page 244).

12.3 Creating Customized Containers

If you wish to use custom applications in multiple identical containers, you can create containers with necessary applications already preinstalled and tuned to meet your demands.

Virtuozzo Hybrid Server offers several ways to create customized containers with preinstalled applications:

- From a golden image (an OS EZ template cache with preinstalled application templates).
- From a custom OS EZ template that specifies a custom application package list.
- From a custom configuration sample file that specifies custom application EZ templates.

12.3.1 Using Golden Images

The golden image functionality allows you to preinstall application templates to OS EZ template caches to speed up creating multiple containers based on the same set of OS and application templates. Previously, you could either install application templates to each container after creating it or embed them directly into a custom OS template. Golden image is currently the easiest and fastest way to create containers with preinstalled applications.

The best way to create such a cache is:

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

```
# cp /etc/vz/conf/ve-basic.conf-sample \
/etc/vz/conf/ve-centos-6-x86_64-mysql-devel.conf-sample
```

Note: If you already have a custom sample configuration file with application EZ templates specified in it, you can reuse it instead of creating a new one.

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

```
# cd /etc/vz/conf
# echo OSTEMPLATE=".centos-6-x86_64" >> ve-centos-6-x86_64-mysql-devel.conf-sample
# echo TEMPLATES=".mysql .devel" >> ve-centos-6-x86_64-mysql-devel.conf-sample
```

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

```
# vzpkg create appcache --config centos-6-x86_64-mysql-devel
```

Note: If the resulting application cache already exists, it will not be recreated and you will see a corresponding message. To recreate an application cache, use the `vzpkg update appcache` command.

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

```
# vzpkg list appcache
centos-6-x86_64          2012-07-20 16:51:36
  mysql
  devel
```

12.3.1.1 Disabling Golden Images

The golden image functionality is enabled by default in the `/etc/sysconfig/vz/vz.conf` global configuration file. Should you wish to disable it, do one of the following:

- Set the `GOLDEN_IMAGE` option to `no` in the Virtuozzo Hybrid Server global configuration file. The golden

image functionality will be disabled globally.

- Set the `GOLDEN_IMAGE` option to `no` in the container sample configuration file. The golden image functionality will be disabled for commands that use this specific sample configuration file.
- Create a file named `golden_image` containing `no` in the OS EZ template's configuration directory. The golden image functionality will be disabled for this specific OS EZ template.
- Create a file named `golden_image` containing `no` in the application template's configuration directory. The golden image functionality will be disabled for this specific application template, so it will not be preinstalled into any OS EZ template caches.

12.3.2 Using Custom EZ Templates

You can create custom OS and application templates tailored to your needs. In such a template, you only need to specify parameters that differ from those in the default template. All other parameters—that are not explicitly set in the custom template—are inherited from the corresponding default template.

To create a custom template, do the following:

1. If required, install the default OS template on the hardware node. For example:

```
yum install centos-7-x86_64-ez
```

2. Create a directory for your template at the location where the default template directory is. For example, for a custom CentOS 7 64-bit template `mytmpl`, create the directory

```
/vz/template/centos/7/x86_64/config/os/mytmpl.
```

3. If you are creating a custom OS template, specify repositories. For example, copy the file `mirrorlist` from the default template directory to your template directory:

```
# cp /vz/template/centos/7/x86_64/config/os/default/mirrorlist \  
/vz/template/centos/7/x86_64/config/os/mytmpl
```

4. In your template directory, create the file `packages` listing the RPMs you need, one per line. For example,

```
systemd  
yum
```

Note: The minimal list of packages to include in a custom template may vary depending on guest OS. For example, CentOS 7 templates require that `systemd` be specified in the `packages` file for the `pr1ct1`

enter command to work on resulting containers.

5. Optionally, change more template parameters according to your needs (for a description of parameters, see the next section).

Your custom template is ready. In this example, it is an OS template that contains `systemd`, `yum`, and all their prerequisites. You now can create containers based on it. For example:

```
# prlctl create MyCT --vmtype ct --ostemplate centos-7-x86_64-mytmpl
```

If you created an application template, you now can add it to the container configuration file as described in *Using Golden Images* on page 252.

12.3.2.1 Migrating EZ Templates

You can migrate OS and application EZ templates between Virtuozzo Hybrid Server servers with the `prlsrvctl cttemplate copy` command.

- To copy the OS EZ template `centos-7-x86-64` to the remote server `remoteserver.com`, on the local server run:

```
# prlsrvctl cttemplate copy remoteserver.com centos-7-x86-64
```

- To copy an application EZ template, additionally specify the name of the corresponding OS EZ template. For example, to copy the application template `mysql` of the OS template `centos-7-x86-64` to the remote server `remoteserver.com`, run:

```
# prlsrvctl cttemplate copy remoteserver.com mysql centos-7-x86-64
```

Note: The specified OS template must be present on the destination server for migration to be successful.

To skip all validation checks, indicate the `--force` option.

The root account is used to log in to the remote server by default, so you will be asked for the root password. You can also provide different credentials (and port) in the format `[<user>[:<passwd>]@]<server>[:<port>]`.

Note that migrating EZ templates always copies them to a remote server. If the source EZ template is not needed after migration, you can manually delete it with the `prlsrvctl cttemplate remove` command.

After migration, you can check that the migrated EZ templates are present on the destination server with the `prlsrvctl cttemplate list` command:

```
# prlsrvctl cttemplate list
centos-7-x86_64 os x86_64 yes Centos 7 (for Intel EM64T) Virtuozzo Hybrid Server Template
...
mysql app x86_64 - mysql for Centos 7 (for Intel EM64T) Virtuozzo Hybrid Server
```

12.3.2.2 EZ Template Configuration Files

All EZ templates are stored in `/vz/template`, in subdirectories named according to OS name, version, and architecture. For example, `/vz/template/centos/7/x86_64`. Each template includes a set of configuration files stored in the `/config/os/<template_name>` subdirectory (OS templates) or the `/config/app/<template_name>` subdirectory (application templates).

The following files can be in the template configuration subdirectory:

- `description` – Detailed information on the EZ template.
- `distribution` – OS templates only. The name of the Linux distribution for which the EZ template is created.
- `environment` – OS templates only. A list of environment variables set in the form of `<key>=<value>`.
- `mirrorlist` – Links to files with lists of repositories from which the packages in the EZ template are to be downloaded.
- `osrelease` – OS templates only. Contains native CentOS 7 distribution kernel version.
- `package_manager` – OS templates only. Specifies the packaging system used to handle the EZ template.
- `packages` – Contains a list of package names included in the corresponding EZ template.
- `pre-cache`, `post-cache` – OS templates only. Scripts that are executed before and after the packages in the EZ template are installed on the hardware node.
- `pre-install`, `post-install` – Scripts that are executed inside the container before and after the package management transaction.
- `pre-install-hn`, `post-install-hn` – Scripts that are executed on the hardware node before and after the package management transaction.
- `pre-upgrade`, `post-upgrade` – OS templates only. Scripts that are executed before and after updating packages inside the container.

- `pre-remove`, `post-remove` – Scripts that are executed before and after removing the application EZ template or package from the container.
- `release` – Contains template release number.
- `repositories` – Contains a list of repositories where the packages in the EZ template are stored.
- `summary` – A brief summary of the EZ template.
- `upgradable_versions` – OS templates only.
- `version` – Contains template version number.

12.3.3 Creating Customized EZ Template RPMs

To share a custom EZ template between hardware nodes, you can create an RPM package with it as follows:

1. Download the default OS template source from <http://download.openvz.org/virtuozzo/releases/7.0/source/SRPMS>.
2. Edit the template according to your needs, e.g., change OS template parameters, add, change or remove application templates, and such.
3. Build the RPM from the `.spec` file in a clean environment using standard tools. Do not build more than one template at once.

12.4 Installing Applications to Containers from EZ Templates

Aside from creating containers with preinstalled application templates (see *Using Golden Images* on page 252), you can install application templates after containers have already been created and started.

Note the following:

- Both the host and the container require Internet access to download RPM packages.
- The container may need a valid fully qualified domain name (FQDN).
- The container may need a dedicated external IP address. Certain applications, like web panels, may be bound by license to external IP addresses. If the containers on the host are behind a NAT gateway and share an external IP address, you may be limited to installing one such application per host.

- It may not be possible to reinstall, update, or uninstall an application template by means of `vzpkg` or otherwise.
- It may be necessary to open the following ports in the container:
 - 2087 for cPanel
 - 2222 for DirectAdmin
 - 1500 for ISPmanager

For more information on ports, refer to the documentation for each application.

Find the desired application template with `vzpkg list -A`, start the container, and install the template as follows:

```
# vzpkg install <CT_name> <app_template>
```

An application may require further setup after the installation. The following subsections describe additional steps required to install some of the applications.

12.4.1 cPanel

This application template is available for CentOS 7.

The container must have at least 768MB of RAM.

To install this template in a container, do the following:

1. Install the RPM package with the application template:

```
# yum install centos-7-x86_64-ez-panels
```

2. Install the template:

```
# vzpkg install <CT_name> cpanel
```

3. Log in to the control panel and complete the installation.

12.4.2 ISPsystem ISPmanager

This application template is available for CentOS 7 and VzLinux 8.

To install this template in a container, do the following:

1. Depending on the container guest OS, install the RPM package with the application template:

```
# yum install vzlinux-8-x86_64-ez-panels
```

Or

```
# yum install centos-7-x86_64-ez-panels
```

2. (VzLinux 8 only) Remove MariaDB and a number of related packages to avoid conflicts with ISPmanager:

```
# vzpkg remove <CT_name> -p mariadb mariadb-connector-c \  
net-snmp-agent net-snmp-agent-libs net-snmp perl-DBD-MySQL
```

3. Install the template:

```
# vzpkg install <CT_name> ispmanager
```

4. Log in to the control panel and complete the installation.

12.4.3 Plesk

This application template is available for CentOS 7.

To install this template in a container, do the following:

1. Install the RPM package with the application template:

```
# yum install centos-7-x86_64-ez-panels
```

2. Install the template:

```
# vzpkg install <CT_name> plesk
```

3. Log in to the control panel and complete the installation.

12.4.4 JBMC Software DirectAdmin

This application template is available for CentOS 7 and VzLinux 8.

To install this template in a container, do the following:

1. Depending on the container guest OS, install the RPM package with the application template:

```
# yum install vzlinux-8-x86_64-ez-panels
```

Or

```
# yum install centos-7-x86_64-ez-panels
```

2. Install the template:

```
# vzpkg install <CT_name> directadmin
```

3. Note the account information in the console output (shown before the installed packages list). For example:

```
<...>
The following information has been set:

Admin username:  admin
Admin password:  jTvNZfzvW@2FYA9f
Admin email:    admin@<hostname>

Server IP: <IP_addr>
Server Hostname: <hostname>

To login now, follow this link:
http://<IP_addr>:2222
and enter your Admin username and password when prompted.
<...>
```

If you miss the account information in the console output, log in to the container as the root user and set a new password for the user admin:

```
# passwd admin
```

4. Log in to the control panel and complete the installation.

12.5 Migrating PowerPanel Container from CentOS 7 to VzLinux 7

The ppconvert tool allows migrating a PowerPanel container from CentOS 7 to VzLinux 7.

12.5.1 Prerequisites

1. Install the following package:

```
# yum install https://repo.virtuozzo.com/vzlinux/vzdeploy/vzdeploy8.rpm
```

2. Ensure your PowerPanel container is running.

12.5.2 Converting PowerPanel Container

Run the following command:

Note: MyCT is the name of a PowerPanel container running the operating system to be converted. Enter the UUID or name of the container you need to migrate to VzLinux 7.

```
# ppconvert MyCT
```

During migration, the tool creates a backup of the PowerPanel container running the operating system, subject to conversion. The PowerPanel backup container is a container with the description `<original-name>_pp_centos7_bkp` and the name `BACKUP_CTUUID`. The UUID of the backup container is listed in the output, for example:

```
All operations completed successfully for CT 267338ae-6b48-4194-bb9b-903dbdffdf1b with BACKUP_CTUII
```

Important: If a bridged network is used, the system removes the IP address in the backup container.

The conversion may take some time and cause the container to reboot. Upon the conversion, the system saves the backup container on the host in the stopped state.

Note: During the execution of `ppconvert`, you may see the following error message:

```
ERROR: ld:so object '/usr/libexec/vztt_checker_open.so' from LD_PRELOAD cannot be preloaded: ignore
```

This message is expected to appear during the script's operation, which means the tool works normally. You can safely ignore this message.

After the successful conversion, the system starts the converted container and checks the PowerPanel service inside it.

Below is an example output of the successful conversion:

```
[root@vhs75]# ppconvert ct-powerpanel
2025-01-13 11:35:34,299 INFO: Converting ct-powerpanel
2025-01-13 11:35:34,308 INFO: CT name is ct-powerpanel
2025-01-13 11:35:34,316 DEBUG:                               100          263 running   172.29.137
```

```

2025-01-13 11:35:34,317 WARNING: Config file /etc/vz/conf/ct-powerpanel.conf is not a symbolic link
2025-01-13 11:35:34,326 INFO: /etc/vz/conf/100.conf is link to the container config and therefore c
2025-01-13 11:35:34,340 DEBUG: NAME="CentOS Linux"
VERSION="7 (Core)"
ID="centos"
ID_LIKE="rhel fedora"
VERSION_ID="7"
PRETTY_NAME="CentOS Linux 7 (Core)"
ANSI_COLOR="0;31"
CPE_NAME="cpe:/o:centos:centos:7"
HOME_URL="https://www.centos.org/"
BUG_REPORT_URL="https://bugs.centos.org/"

CENTOS_MANTISBT_PROJECT="CentOS-7"
CENTOS_MANTISBT_PROJECT_VERSION="7"
REDHAT_SUPPORT_PRODUCT="centos"
REDHAT_SUPPORT_PRODUCT_VERSION="7"

2025-01-13 11:35:34,393 DEBUG: pp-release-2.0.4-3.v17.noarch

2025-01-13 11:35:34,394 INFO: OS Version and PP check is completed successfully
locking 100
locking 9a9161a0-f523-44cc-af26-20a780e84e37
Copy /vz/private/100/root.hdd/DiskDescriptor.xml /vz/private/9a9161a0-f523-44cc-af26-20a780e84e37/r
Copy /vz/private/100/ /vz/private/9a9161a0-f523-44cc-af26-20a780e84e37
vzctl : Container is not running
vzctl : Failed to exec action script /usr/libexec/libvzctl/dists/scripts/postcreate.sh
/usr/sbin/vzctl exited with code 79
post create action failed for CT 9a9161a0-f523-44cc-af26-20a780e84e37
unlocking 100
unlocking 9a9161a0-f523-44cc-af26-20a780e84e37
WARNING: You are using a deprecated CLI component that won't be installed by default in the next ma
2025-01-13 11:35:55,529 INFO: Backup created successfully with BACKUP_CTUID: 9a9161a0-f523-44cc-af
2025-01-13 11:35:55,542 INFO: CT ct-powerpanel switched successfully to vzlinux distribution.
ERROR: ld.so: object '/usr/libexec/vztt_checker_open.so' from LD_PRELOAD cannot be preloaded: ignor
2025-01-13 11:37:46,681 INFO: CT ct-powerpanel upgraded successfully.
Warning: RPMDB altered outside of yum.
2025-01-13 11:38:17,123 INFO: yum -y distro-sync completed successfully in CT ct-powerpanel.
2025-01-13 11:38:19,799 INFO: Yum shell commands executed successfully in CT ct-powerpanel.
WARNING: You are using a deprecated CLI component that won't be installed by default in the next ma
2025-01-13 11:38:32,441 INFO: CT ct-powerpanel have been restarted.
2025-01-13 11:38:43,945 INFO: PowerPanel vzapi in CT ct-powerpanel is accessible
2025-01-13 11:38:43,945 INFO: CT ct-powerpanel PowerPanel seems to be working
2025-01-13 11:38:44,241 DEBUG: Output of needs-restarting -r for CT ct-powerpanel:
No core libraries or services have been updated.
Reboot is probably not necessary.
2025-01-13 11:38:44,241 WARNING: No restart is necessary for CT ct-powerpanel.
2025-01-13 11:38:44,413 DEBUG: Output of needs-restarting -r for CT ct-powerpanel:
No core libraries or services have been updated.
Reboot is probably not necessary.
2025-01-13 11:38:44,413 WARNING: No restart is necessary for CT ct-powerpanel.

```

```
2025-01-13 11:38:44,413 INFO: All operations completed successfully for CT ct-powerpanel with BACKU
```

If the converted container or PowerPanel service inside it is not running, the system takes the following actions:

1. Stopping the container.
2. Renaming it to <original-name>-convert-failed.
3. Removing network configuration from the failed container.
4. Restoring the backup of the container.
5. Starting the container.
6. Informing about the failed conversion.

Important: If a bridged network is used, assign the removed IP address back to the original container on your own:

```
# prlctl set powerpanel --device-set net0 --ipadd <addr>
```

12.5.3 Manual Reversion to Backup Container

You can manually revert a container to its previous state if required or if automatic reversion to the backup container, applied after conversion issues, fails. To do it:

1. (Optional) Rename the backup container with a unique name (ensure it is other than the source CT UUID, CTID, or name):

```
# prlctl set BACKUP_CTUUID --name UNIQUE_CT_NAME
```

2. If a bridged network is used, assign the removed IP address back to the original container on your own:

```
# prlctl set BACKUP_CTUUID --device-set net0 --ipadd <addr>
```

3. Stop the converted container if it is running:

```
# prlctl stop <original-name>
```

4. Start the backup container:

```
# prlctl start BACKUP_CTUID
```

12.6 Creating Virtual Environments with virt-install

While you typically create virtual machines and containers using `prlctl`, you can also do the same with `virt-install`. You will need to install it from the Virtuozzo official repository with `yum install virt-install`.

You can manage VMs created with `virt-install` using `prlctl`. You will need, however, to manually delete the hard disk images after you delete such VMs with `prlctl delete`.

To shorten the commands and make sure that the required parameters are always supplied, the common options for creating virtual environments are placed in presets. You can list the default presets and their contents as follows:

```
# virt-install --list-presets
vz_vm_windows - Virtuozzo VM Windows
vz_vm_linux   - Virtuozzo VM Linux
vz_ct_windows - Virtuozzo CT Windows
```

```
# virt-install --show-preset vz_vm_linux
```

```
# virt-install --show-preset vz_vm_windows
```

You can also create your own presets and place them in `/etc/virt-manager/presets`. If a preset starts with a commented-out line, it is used to describe the preset in the output of `virt-install --list-presets`.

Otherwise “No description” is shown.

Note: The `vz_ct_windows` preset is meant for the Windows Containers Demo. For more details, see the [Windows Containers Demo Walkthrough](#).

The following sample command creates a blank VM based on the `vz_vm_linux` preset optimized for running CentOS 7:

```
# virt-install \
--import \
--name mylinuxvm \
--ram 2048 \
--vcpus 'sockets=1,cores=2' \
```

```
--disk 'path=/vz/mylinuxvm/harddisk.hdd,bus=scsi,startup_policy=optional,boot_order=1,size=64' \
--disk 'device=cdrom,path=myosdistrib.iso,bus=scsi,boot_order=2' \
--preset vz_vm_linux \
--graphics=vnc,password=mypasswd,listen=0.0.0.0,port=6533
```

The resulting VM has the following properties:

- 2048 MB RAM
- 1 CPU socket and 2 CPU cores
- 64 GB hard disk in the file `/vz/mylinuxvm/harddisk.hdd`, which is created along with the VM (the directory must exist)
- A CD-ROM drive where an existing file `myosdistrib.iso`, e.g., a Linux distribution image, is mounted to boot and install from
- VNC enabled on port 6533 at host's IP address

The next sample command creates a blank VM based on the `vz_vm_windows` preset optimized for running Microsoft Windows Server 2019:

```
# virt-install \
--import \
--name mywindowsvm \
--ram 2048 \
--vcpus 'sockets=1,cores=2' \
--disk 'path=/vz/mywindowsvm/harddisk.hdd,bus=scsi,startup_policy=optional,boot_order=1,size=64' \
--disk 'device=cdrom,path=myosdistrib.iso,bus=scsi,boot_order=2' \
--preset vz_vm_windows \
--graphics=vnc,password=mypasswd,listen=0.0.0.0,port=6534
```

The resulting VM has the following properties:

- 2048 MB RAM
- 1 CPU socket and 2 CPU cores
- 64 GB hard disk in the file `/vz/mywindowsvm/harddisk.hdd`, which is created along with the VM (the directory must exist)
- A CD-ROM drive where an existing file `myosdistrib.iso`, e.g., a Microsoft Windows distribution image, is mounted to boot and install from
- VNC enabled on port 6534 at host's IP address

Finally, the sample commands to create a Linux container based on the CentOS 7 template are:

```
# vzpkg create image centos-7-x86_64 mylinuxct.hdd
# virt-install \
--connect vzct:///system \
--name mylinuxct \
--memory 2048 \
--disk 'path=/vz/mylinuxct/mylinuxct.hdd,boot_order=1'
```

The first command creates a hard disk image for the container. It is made from the same template that is used when creating containers using `prlctl`. The second command creates the container with the specified hard disk.

You can resize the disk with `virsh blockresize <CT_name> <device_name> <size><units>`. The container can be running or stopped. For example:

```
# virsh --connect vzct:///system blockresize mylinuxct vda 16G
# prlctl list -i mylinuxct | grep hdd
hdd0 (+) scsi:0 image='/root/mylinuxct.hdd' type='expanded' 16384Mb <...>
```

12.7 Setting Up Docker and Kubernetes in Virtuozzo Containers

You can run Docker and Kubernetes inside Virtuozzo Hybrid Server system containers.

In the current version of Virtuozzo Hybrid Server, you can use system containers based on CentOS 7 as well as Ubuntu 18.04 and 20.04.

Note: Modules and third party add-ons that depend on operations prohibited in containers (loading of kernel modules, mounting of block devices, direct access to physical hardware) may not work in containers.

Essentially, you need to:

1. Enable and start the `vz-k8s-inside-ct` service that configures the hardware node for running Kubernetes and Docker swarm mode in system containers.
2. Create a system container with a network connection and enough resources to run the Docker containers you need.
3. Install Docker inside the system container according to the [official documentation](#). It is recommended to change the cgroup driver to `systemd`.

4. Install Kubernetes inside the system container according to the [official documentation](#).
5. Create more similar system containers and join them into a Kubernetes cluster or a Docker swarm.

These steps are demonstrated in more detail in the following tutorial:

1. Enable and start the `vz-k8s-inside-ct` service that configures the hardware node for Kubernetes and Docker swarm mode:

```
# systemctl enable vz-k8s-inside-ct && systemctl start vz-k8s-inside-ct
```

Note: Restart running containers to apply these configuration changes to them.

2. Create a system container with a supported guest OS. For example:

```
# prlctl create dockerct1 --vmtype ct --ostemplate centos-7-x86_64
```

3. Provide the system container with enough resources. For example, add more RAM:

```
# prlctl set dockerct1 --memsize=2G
Set the memsize parameter to 2048Mb.
```

4. Disable swap in the system container as [required for Kubernetes](#). For example:

```
# prlctl set dockerct1 --swap=0
Set swappages 0
```

5. Set up network in the system container. For example, add a network adapter connected to a bridged network (by default) and enable DHCP for it:

```
# prlctl set dockerct1 --device-add net --dhcp yes
Enable automatic reconfiguration for this network adapter.
Creating net0 (+) dev='' ifname='eth0' network='Bridged' mac=<...> card=virtio dhcp='yes'
Created net0 (+) dev='veth72d71c9b' ifname='eth0' network='Bridged' mac=<...> dhcp='yes'
```

6. Set a unique hostname for the system container. For example:

```
# prlctl set dockerct1 --hostname dockerct1.example.local
```

Note: Make sure to register the hostname with the DNS server.

7. Set the root password for the system container to be able to log in via SSH:

```
# prlctl set dockerct1 --userpasswd root:<passwd>
```

Where <passwd> is a strong password.

8. Start and log in to the system container:

```
# prlctl start dockerct1
# ssh <dockerct1_IP>
```

Where <dockerct1_IP> is the system container's IP address. If you previously enabled DHCP, you may need to find out the IP address. For example:

```
# prlctl exec dockerct1 ip -4 a | grep inet
<...>
inet 10.194.110.194/16 brd 10.194.255.255 scope global dynamic eth0
```

Perform the next steps inside the system container.

9. Open the ports required by Kubernetes and the Docker swarm:

- For Kubernetes control-plane:

```
# firewall-cmd --permanent --add-port=6443/tcp;\
firewall-cmd --permanent --add-port=2379-2380/tcp;\
firewall-cmd --permanent --add-port=10250/tcp;\
firewall-cmd --permanent --add-port=10251/tcp;\
firewall-cmd --permanent --add-port=10252/tcp;\
firewall-cmd --reload
```

- For the Docker swarm:

```
# firewall-cmd --permanent --add-port=2377/tcp;\
firewall-cmd --permanent --add-port=7946/tcp;\
firewall-cmd --permanent --add-port=7946/udp;\
firewall-cmd --permanent --add-port=4789/udp;\
firewall-cmd --reload
```

10. Install Docker as laid out in the [official guide](#). Before you start the Docker service, however, change the cgroup driver to systemd:

```
# mkdir /etc/docker
# cat > /etc/docker/daemon.json <<EOF
{
  "exec-opts": [
    "native.cgroupdriver=systemd"
  ],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m"
  },
}
```

```

"storage-driver": "overlay2",
"storage-opts": [
  "overlay2.override_kernel_check=true"
]
}
EOF

```

11. Enable and start the Docker service:

```
# systemctl enable docker && systemctl daemon-reload && systemctl start docker
```

Note: If the Docker service fails to start with the non-mandatory option `storage-opts`, remove this option. The final configuration is as follows:

```

# cat > /etc/docker/daemon.json <<EOF
{
  "exec-opts": [
    "native.cgroupdriver=systemd"
  ],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m"
  },
  "storage-driver": "overlay2"
}
EOF

```

12. Verify that Docker Engine works:

```

# docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
0e03bdcc26d7: Pull complete
Digest: sha256:4cf9c47f86df71d48364001ede3a4fcd85ae80ce02ebad74156906caff5378bc
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.
<...>

```

13. Install `kubelet`, `kubeadm`, and `kubectl` packages according to the [official documentation](#).

14. Initialize the Kubernetes control-plane node. For example:

```

# kubeadm init
<...>
Your Kubernetes control-plane has initialized successfully!

```

To start using your cluster, you need to run the following as a regular user:

```
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

You should now deploy a pod network to the cluster.

Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:

<https://kubernetes.io/docs/concepts/cluster-administration/addons/>

Then you can join any number of worker nodes by running the following on each as root:

```
kubeadm join 10.194.110.194:6443 --token qrdi2v.t4rr9yyylpgpnkcw \
--discovery-token-ca-cert-hash sha256:a7d0e65<...>
```

15. As you are the root user, run

```
# export KUBECONFIG=/etc/kubernetes/admin.conf
```

16. Deploy a pod network. For example:

```
# kubectl apply -f "https://cloud.weave.works/k8s/net?k8s-version=$(kubectl version | base64 | tr -d '\n')serviceaccount/weave-net created
clusterrole.rbac.authorization.k8s.io/weave-net created
clusterrolebinding.rbac.authorization.k8s.io/weave-net created
role.rbac.authorization.k8s.io/weave-net created
rolebinding.rbac.authorization.k8s.io/weave-net created
daemonset.apps/weave-net created"
```

17. Check that Kubernetes has been set up correctly. For example:

```
# kubectl get pods --all-namespaces
```

NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE
kube-system	coredns-f9fd979d6-29k27	1/1	Running	0	5m
kube-system	coredns-f9fd979d6-f6xd5	1/1	Running	0	5m
kube-system	etcd-dockerct1.example.local	1/1	Running	0	5m
kube-system	kube-apiserver-dockerct1.example.local	1/1	Running	0	5m
kube-system	kube-controller-manager-dockerct1.example.local	1/1	Running	0	5m
kube-system	kube-proxy-559tc	1/1	Running	0	5m
kube-system	kube-scheduler-dockerct1.example.local	1/1	Running	0	5m
kube-system	weave-net-jzm8p	2/2	Running	0	5m

18. Create a Docker swarm using the system container's IP address. For example:

```
# docker swarm init --advertise-addr 10.194.110.194
Swarm initialized: current node (n1slc7on484f7ak4d3vaz6tk4) is now a manager.
To add a worker to this swarm, run the following command:
docker swarm join --token SWMTKN-1-<...> 10.194.110.194:2377
To add a manager to this swarm, run 'docker swarm join-token manager'
and follow the instructions.
```

19. Create more (e.g., two more) system containers with the same configuration as explained in steps 2 to 8. They will run Kubernetes and Docker swarm workers.

Perform the next steps in each of the new system containers.

20. Open the ports required by Kubernetes and the Docker swarm:

- For the Kubernetes worker:

```
# firewall-cmd --permanent --add-port=10250/tcp;\
firewall-cmd --permanent --add-port=30000-32767/tcp;\
firewall-cmd --reload
```

- For the Docker swarm:

```
# firewall-cmd --permanent --add-port=2377/tcp;\
firewall-cmd --permanent --add-port=7946/tcp;\
firewall-cmd --permanent --add-port=7946/udp;\
firewall-cmd --permanent --add-port=4789/udp;\
firewall-cmd --reload
```

21. Install Docker and Kubernetes as explained in steps 10 to 13.

22. Add workers to the Kubernetes cluster with the command reported by `kubeadm init`. For example:

```
# kubeadm join 10.194.110.194:6443 --token qrDi2v.t4rr9yyylpgpnkcw \
--discovery-token-ca-cert-hash sha256:a7d0e65<...>
```

23. Add workers to the Docker swarm with the command reported by `docker swarm init`. For example:

```
# docker swarm join --token SWMTKN-1-<...> 10.194.110.194:2377
This node joined a swarm as a worker.
```

24. Log in to the system container running the control plane and swarm manager (`dockerct1` in this tutorial) and check that both the Kubernetes cluster and Docker swarm work. For example:

```
# kubectl get nodes
NAME                 STATUS    ROLES    AGE   VERSION
dockerct1.example.local Ready    master   10m   v1.19.2
dockerct2.example.local Ready    <none>   5m    v1.19.2
dockerct3.example.local Ready    <none>   5m    v1.19.2
```

```
# docker info
<...>
Swarm: active
NodeID: n1slc7on484f7ak4d3vaz6tk4
Is Manager: true
ClusterID: lyz9u13zoswtm03jkswqcdzaj
Managers: 1
Nodes: 3
Default Address Pool: 10.0.0.0/8
```

```

SubnetSize: 24
Data Path Port: 4789
Orchestration:
  Task History Retention Limit: 5
Raft:
  Snapshot Interval: 10000
  Number of Old Snapshots to Retain: 0
  Heartbeat Tick: 1
  Election Tick: 10
Dispatcher:
  Heartbeat Period: 5 seconds
CA Configuration:
  Expiry Duration: 3 months
  Force Rotate: 0
Autolock Managers: false
Root Rotation In Progress: false
Node Address: 10.194.110.194
Manager Addresses:
  10.194.110.194:2377
<...>

```

After completing this tutorial, you have three or more Virtuozzo Hybrid Server 7 system containers running Docker and Kubernetes. You have also created a Kubernetes cluster and a Docker swarm.

12.8 Managing Container Virtual Hard Disk Encryption

Virtuozzo Hybrid Server offers container virtual hard disk encryption capabilities based on `dm-crypt` and `cryptsetup`. The current implementation uses the AES-256 encryption algorithm. The encryption mechanism is separated from encryption key management, enabling you to use your own key management system (KMS) to issue and manage encryption keys.

Important: Only the root user must have access to encryption operations on the host.

The overall encryption procedure may be described as follows. An end user requests encryption for their container disk (create a new container with an encrypted disk, encrypt an existing disk, etc.). The KMS stores a secret encryption key and a public encryption key ID assigned to the end user. The administrator (or an automation tool) obtains the end user's encryption key ID from the KMS and passes it to the corresponding `pr1ct1` command. The `pr1ct1` command passes the key ID to the `getkey` executable that accesses the KMS and returns the encryption key which is passed to `cryptsetup`. The `cryptsetup` tool issues a master key

unique for the container disk, encrypts the disk contents with the master key, then encrypts the master key stored in container disk's LUKS header with the encryption key passed from `getkey`.

The double encryption saves host resources in situations when the encryption key has to be changed (e.g., for key rotation purposes). In such cases, only the master key in the LUKS header has to be re-encrypted instead of the entire disk contents.

The only configuration step required to start using container disk encryption capabilities in Virtuozzo Hybrid Server is to set up the encryption key requester as described further to be able to obtain encryption keys by their IDs for corresponding `pr1ct1` commands.

12.8.1 Setting Up Encryption Key Requester

The encryption mechanism communicates with the KMS as follows: executes the file `/usr/libexec/ploop/crypt.d/getkey` with the only string parameter—encryption key ID—and reads the returned encryption key value from the standard output. The `getkey` executable can be a script that calls the KMS binary and passes the specified encryption key ID to it.

On success, the executable is expected to exit with zero code and the key value is expected to be printed to the standard output in the binary form. On failure, the script is expected to exit with non-zero code.

Note: Key value may contain arbitrary bytes (e.g., `\x00`, `\n`, and such) that may be treated differently by various scripting languages. For example, assigning the key value to a Bash variable would strip the zero bytes from it, e.g., `key_val\x00lue` would become `key_value`.

To set up the encryption key requester on a Virtuozzo Hybrid Server host, place the script to `/usr/libexec/ploop/crypt.d/getkey` and make it executable and only accessible by the root user:

```
# chown root:root /usr/libexec/ploop/crypt.d/getkey
# chmod 700 /usr/libexec/ploop/crypt.d/getkey
```

12.8.2 Encrypting and Decrypting Container Virtual Hard Disks

Following is a list of encryption-related operations you can perform on container virtual hard disks. All operations except `decrypt` require an encryption key ID obtained from your KMS.

Note: PFCache is disabled for encrypted disks.

1. Create a container with an encrypted root disk.

```
# prlctl create <CT_name|CT_UUID> --vmtype ct --encryption-keyid <key_id>
```

2. Add a new encrypted disk to a container.

```
# prlctl set <CT_name|CT_UUID> --device-add hdd --encryption-keyid <key_id>
```

3. Encrypt an unencrypted disk. During this operation, a new empty encrypted disk is created, the data is moved to it from the unencrypted disk which is then securely erased with `shred` (unless `--no-wipe` is added). For this operation, the host needs free disk space equal to the size of the container disk being encrypted.

```
# prlctl set <CT_name|CT_UUID> --device-set hdd0 --encrypt --encryption-keyid <key_id> \  
[--no-wipe]
```

4. Change the encryption key ID of an encrypted disk with or without re-encrypting the entire disk contents. By default, only the LUKS header of an encrypted disk is re-encrypted to save host resources. The entire disk contents can be re-encrypted by adding `--reencrypt` to the command. During this operation, a new empty encrypted disk is created, the data is moved to it from the old encrypted disk which is then securely erased with `shred` (unless `--no-wipe` is added). For this operation, the host needs free disk space equal to the size of the container disk being re-encrypted.

```
# prlctl set <CT_name|CT_UUID> --device-set hdd0 --encryption-keyid <key_id> [--reencrypt] \  
[--no-wipe]
```

5. Decrypt an encrypted container disk. During this operation, a new empty unencrypted disk is created and the data is moved to it from the encrypted disk. If the operation completes successfully, the encrypted disk is deleted. If the operation fails, the partially decrypted data is securely erased with `shred` (unless `--no-wipe` is added) and the encrypted disk remains intact. For this operation, the host needs free disk space equal to the size of the container disk being decrypted.

```
# prlctl set <CT_name|CT_UUID> --device-set hdd0 --decrypt [--no-wipe]
```


12.8.3 Encrypting System Swap

Even if containers are encrypted, their memory may still be swapped to an unencrypted system swap partition. On nodes intended to run encrypted containers, consider encrypting the swap partition as well.

To encrypt a swap partition with a random encryption key, do the following:

1. Identify swap partitions in the system:

```
# swapon -s
Filename      Type      Size      Used      Priority
/dev/<partition> partition XXXXXXXX 0         -1
```

2. Add a swap partition's entry to the `/etc/crypttab` file. For example:

```
swap /dev/<partition> /dev/urandom swap,noearly
```

Once the encryption is initiated, this will map the `/dev/<partition>` to `/dev/mapper/swap` as an encrypted swap partition.

Note: To create an encryption key, you may use `/dev/random` instead of `/dev/urandom`. However, in this case, the system may take a long time to boot.

3. Find out the UUID of the swap partition:

```
# blkid /dev/<partition>
/dev/<partition>: UUID="XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX" TYPE="swap"
```

4. Comment the swap partition's entry in the `/etc/fstab` file and add an entry for the mapped `/dev/mapper/swap` file. For example:

```
#UUID=<partition_UUID> swap swap defaults 0 0
/dev/mapper/swap none swap sw 0 0
```

5. Reboot to initiate the encryption.
6. Run the `lsblk` command to ensure the swap is encrypted. In the output, the swap partition's TYPE should be `crypt`.

```
# lsblk
NAME      MAJ:MIN  RM  SIZE RO TYPE  MOUNTPOINT
...
  <swap_partition>  X:X     0   XG   0  crypt [SWAP]
...
```

The output of `swapon -s` will show a different filename:

```
# swapon -s
Filename                                Type      Size      Used      Priority
/dev/dm-X                               partition XXXXXXXX 0         -1
```

12.9 Connecting to Virtual Machines and Containers via VNC

You can use your favorite VNC clients to connect to and manage containers and virtual machines. To do this, you need to complete these steps:

1. (Recommended) Secure VNC connections on the node with SSL.
2. Enable VNC access in the desired virtual machine or container.
3. Connect to the virtual machine or container with a VNC client.

The sections below describe these steps in detail.

12.9.1 Securing VNC Connections with SSL

To set up SSL for all VNC connections on the node, do the following:

1. Acquire an SSL certificate and key from a trusted certificate authority.

Note: The key for an SSL certificate should not be protected by a passphrase.

2. Configure the VNC server to use the certificate and key:

```
# prlsvctl set --vnc-ssl-certificate <path_to_cert_file> --vnc-ssl-key <path_to_key_file>
```

The certificate will protect VNC connections to virtual machines started after executing this command.

If you are replacing an expired certificate, you can apply the new one to running VMs by restarting the dispatcher service:

```
# systemctl restart prl-disp
```

To disable VNC encryption, specify empty arguments. For example:

```
# prlsrvctl set --vnc-ssl-certificate '' --vnc-ssl-key ''
```

12.9.1.1 Using a Certificate Chain to Encrypt VNC Connections

If you acquire an SSL certificate from an intermediate certificate authority (CA), you will get an end-user certificate along with a CA bundle that contains the root and intermediate certificates. To be able to use these certificates for VNC encryption, you need to merge them into a chain first. A certificate chain includes the end-user certificate, the certificates of intermediate CAs, and the certificate of a trusted root CA. In this case, an SSL certificate can only be trusted if every certificate in the chain is properly issued and valid.

For example, if you have an end-user certificate, two intermediate CA certificates, and a root CA certificate, you can encrypt all VNC connections on the node as follows:

1. Create a new certificate file and include all certificates in it in the following order:

```
# End-user certificate issued by the intermediate CA 1
-----BEGIN CERTIFICATE-----
MIICiDCCAg2gAwIBAgIQNfwmXNmET8k9Jj1X<...>
-----END CERTIFICATE-----
# Intermediate CA 1 certificate issued by the intermediate CA 2
-----BEGIN CERTIFICATE-----
MIEIDCCAwigAwIBAgIQNE7VVyDV7exJ90N9<...>
-----END CERTIFICATE-----
# Intermediate CA 2 certificate issued by the root CA
-----BEGIN CERTIFICATE-----
MIIC8jCCAdqgAwIBAgICZngwDQYJKoZIhvcN<...>
-----END CERTIFICATE-----
# Root CA certificate
-----BEGIN CERTIFICATE-----
MIIDODCCAiCgAwIBAgIGIAYFFnACMA0GCSqG<...>
-----END CERTIFICATE-----
```

2. Specify the full paths to the newly created certificate file and the private key for the end-user certificate in the following command:

```
# prlsrvctl set --vnc-ssl-certificate <path_to.crt_chain> --vnc-ssl-key <path_to_key_file>
```

12.9.1.2 Securing Previously Enabled VNC Connections

If you enable VNC in a virtual environment and then install an SSL certificate, that VE will not use encryption until you do the following:

1. Stop the virtual environment:

```
# prlctl stop MyVE
```

2. Disable VNC in the virtual environment:

```
# prlctl set MyVE --vnc-mode off
```

3. Re-enable VNC in the virtual environment as described in the two next sections.
4. Start the virtual environment.

```
# prlctl start MyVE
```

Now SSL encryption is applied to the VNC traffic to and from the virtual environment.

12.9.2 Enabling VNC Access to Virtual Machines and Containers

To enable VNC access to a virtual machine or container, do the following:

- Select a VNC mode: auto or manual.
- Set a password to secure your VNC connection or choose to disable it.
- Specify a TCP port number on the host that will be used to listen to VNC connections.

The port number must be unique. In the auto mode, unique port numbers are assigned automatically. In the manual mode, you need to make sure that port numbers are unique.

A VNC port can be set for both running and stopped virtual machines and containers. The VNC mode and password can only be set for stopped virtual environments.

You can perform all these operations with a single command. For example:

```
# prlctl set MyVM --vnc-mode manual --vnc-passwd XXXXXXXX --vnc-port 5901
```

Or

```
# prlctl set MyVM --vnc-mode auto --vnc-passwd XXXXXXXX
```

12.9.3 Connecting with a VNC Client

After you have enabled VNC access to the virtual machine or container, you can connect to it with your favorite VNC client. To do this, you need to pass the following parameters to the VNC client:

- IP address of the server where the virtual machine or container is hosted.

- Port number and password you specified when enabling VNC access.
- Valid user account in the virtual machine or container.

You can set a VNC IP address to newly created VMs. For this, do the following:

```
# prlsrvctl set --vnc-default-address <ip>
```

Also, we can set a VNC IP address for already existing VMs with the following command:

```
# prlctl set MyVM --vnc-address <address>
```

You can copy and paste the text into your virtual environment. To enable or disable the option of copying to the clipboard, use the following command:

```
# prlsrvctl set --vnc-clipboard <on|off>
```

To be able to copy and paste the text into a virtual environment, do the following:

Note: Please note that after the dispatcher config edit, changes will only apply to newly created VMs. If you already have a VM that requires VNC copy-and-paste functionality:

1. Execute `virsh edit`.
2. For Windows or Linux, in the dispatcher config editor, add the following at the end of the file before the `</domain>` entry:

```
<qemu:commandline>
  <qemu:arg value='-chardev' />
  <qemu:arg value='qemu-vdagent,id=vdagent,clipboard=on' />
  <qemu:arg value='-device' />
  <qemu:arg value='virtserialport,chardev=vdagent,name=com.redhat.spice.0' />
</qemu:commandline>
```

1. Stop the dispatcher on the node:

```
# systemctl stop prl-disp.service
```

Note: While the dispatcher is stopped, you cannot manage and collect stats of VMs and containers. Running VMs and containers are not stopped.

2. In `/etc/vz/dispatcher.xml`, set `EnableClipboard` to 1.
3. Start the dispatcher:

```
# systemctl start pr1-disp.service
```

4. Install the [SPICE agent](#) into the virtual environment.

On Linux, you can install the `spice-vdagent` package. On Windows, you can install a corresponding [SPICE agent binary](#).

If you deploy virtual environments from ready images, consider adding the SPICE agent to them to have this functionality out of the box.

Note the following:

- With copying and pasting enabled, virtual environments cannot be migrated live to older versions of Virtuozzo Hybrid Server.
- The SPICE agent requires a graphical user interface (GUI), e.g., X Window System on Linux.
- Not all VNC clients fully support copy and paste functionality.

12.10 Managing iptables Modules

This section describes how to manage `iptables` modules for both physical servers and containers.

12.10.1 Using iptables Modules in Virtuozzo Hybrid Server

Filtering network packets on hardware nodes running Virtuozzo Hybrid Server does not differ from doing so on a typical Linux server. You can use the standard `iptables` tool to control how network packets enter, move through, and exit the network stack within the Virtuozzo Hybrid Server kernel.

Connection tracking on the hardware node is disabled by default. Setting `iptables` rules that require `conntrack` functionality enables tracking of new connections and makes the node vulnerable to DoS attacks, since the number of `conntrack` slots is limited. However, setting such rules for particular virtual machines and containers (e.g., for NAT) leaves other containers, virtual machines and the hardware node reachable in case of a DoS attack.

Note: Once `conntrack` is enabled for a container, it cannot be disabled until the restart of the hardware node or the container.

To detect active connections tracked on the hardware node, check if the `/proc/net/nf_contrack` file contains any entries:

```
# cat /proc/net/nf_contrack
```

For your reference, below are several resources you can consult to get detailed information on using iptables on Linux servers:

- [Red Hat Enterprise Linux 7 Security Guide](#) contains a section focusing on packet filtering basics and explaining various options available for iptables.
- [iptables Tutorial 1.2.2](#) explains in great detail how iptables is structured and works.

12.10.2 Using iptables Modules in Containers

Using iptables modules in containers requires additional configuration on your part.

12.10.2.1 Configuring iptables Modules

To set the state of iptables modules for backup/restore or live migration, use the `prlctl set --netfilter` command. If some of the iptables modules allowed for a container are not loaded on the hardware node where that container has been restored or migrated, they will be automatically loaded when that container starts. For example, the command

```
# prlctl set MyCT --netfilter stateful
```

will make sure that all modules except NAT-related will be allowed and loaded for the container `MyCT` (if required) on a hardware node where it has been restored or migrated.

Note: The default setting is `full`, which allows all modules.

12.10.2.2 Using conntrack Rules and NAT Tables

To limit the maximum number of conntrack slots available for each container on the hardware node, set the `net.netfilter.nf_conntrack_max` variable. For example:

```
# sysctl -w net.netfilter.nf_conntrack_max=50000
```

The value of `net.netfilter.nf_conntrack_max` cannot exceed the value of `net.nf_conntrack_max`.

Even if a container is under a DoS attack and all its `conntrack` slots are in use, other containers will not be affected and will still be able to create as many connections as set in `net.netfilter.nf_conntrack_max`.

12.11 Using SCTP in Containers and Virtual Machines

Virtuozzo Hybrid Server supports the Stream Control Transmission Protocol (SCTP) in both containers and virtual machines. In virtual machines, SCTP can be set up and used like on any physical machine. In containers, SCTP support is disabled by default. To enable SCTP in containers, load the `sctp` kernel module on the hardware node with

```
# modprobe sctp
```

After the module is loaded, you can create and use SCTP sockets inside containers with standard Linux tools.

If required, add an `sctp` entry to the `/etc/modules-load.d/vz.conf` file to automatically load the `sctp` kernel module on hardware node start.

Note: Live migration via SCTP is not supported.

12.12 Creating Configuration Files for New Linux Distributions

Distribution configuration files are used to distinguish among containers running different Linux versions and to determine what scripts should be executed when performing the relevant container-related operations (e.g., assigning a new IP address to the container).

All Linux distributions shipped with Virtuozzo Hybrid Server have their own configuration files located in the `/usr/libexec/libvzctl/dists/` directory on the hardware node. However, you may wish to create your own distribution configuration files to support new Linux versions released. Let us assume that you wish your containers to run the CentOS 7 Linux distribution and, therefore, have to make the `centos-7.conf` distribution configuration file to define what scripts are to be executed while performing major tasks with containers

running this Linux version. To do this:

1. In the container configuration file (with the name of `/etc/vz/conf/<UUID>.conf`), specify `centos-7` as the value of the `DISTRIBUTION` variable (for example, `DISTRIBUTION="centos-7"`).
2. Create the `centos-7.conf` configuration file in the `/usr/libexec/libvzctl/dists/` directory. The easiest way to do it is copy one of the existing configuration files by executing the following command in the `/usr/libexec/libvzctl/dists/` directory:

```
# cp fedora.conf centos-7.conf
```

In the example above, we assume that the `fedora.conf` file is present in the `/usr/libexec/libvzctl/dists/` directory on the hardware node. In case it is not, you may use any other distribution configuration file available on your server.

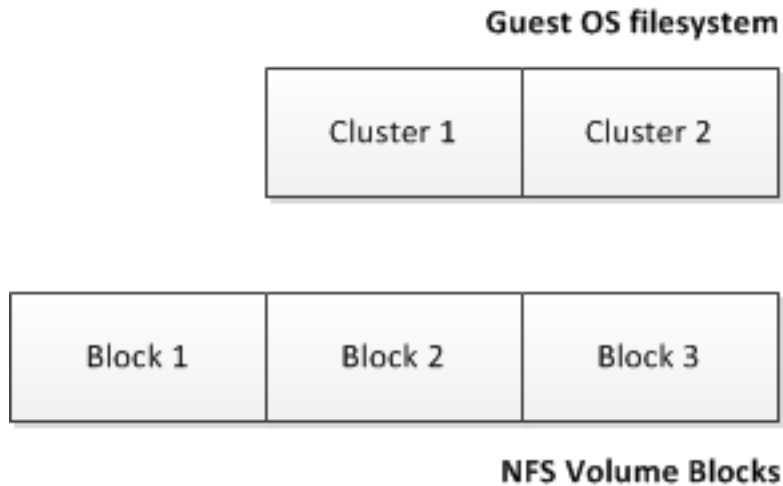
3. Open the `centos.conf` file for editing, go to the first entry and, in the right part of the entry, specify the name of the script you wish to be run on issuing the `prlctl` command with the parameter specified in the left part of the entry. For example, if you wish the script to be executed while assigning a new IP address to your container and the script has the `my_centos_script` name, your entry should look as follows:

```
ADD_IP=my_centos_script-add_ip.sh
```

4. Repeat Step 3 for all entries in the file.
5. Place the scripts for the new Linux distribution to the `/usr/libexec/libvzctl/dists/scripts` directory on the Node. Make sure the names of these scripts coincide with those specified in the `centos-7.conf` file.

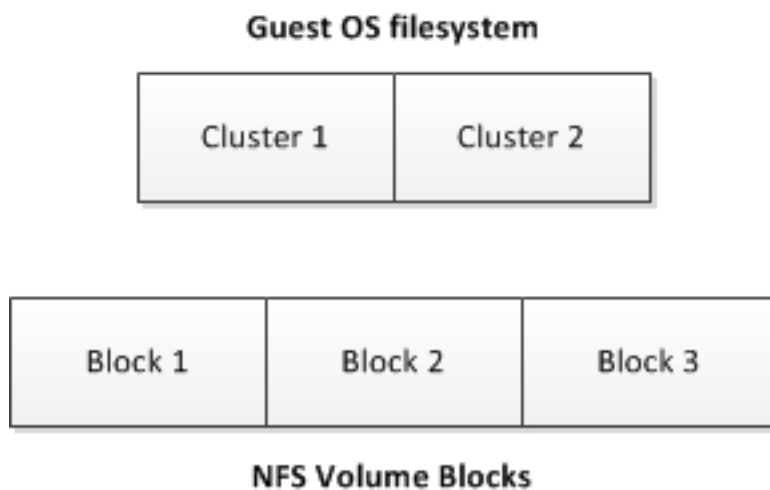
12.13 Aligning Disks and Partitions in Virtual Machines

Most of the modern operating systems automatically align partitions when they are installed in virtual machines. For example, Windows Server 2008 creates a default partition offset of 1024 KB to satisfy the partition alignment requirements. The following figure shows an example of correct partition alignment:



In this example, any cluster (the smallest unit of data) in the guest OS file system is aligned with the boundaries of an NFS block, and reading from or writing to a cluster requires only access to one NFS block. For example, reading from Cluster 1 causes only a read from Block 1.

At the same time, virtual machines running non-modern systems (for example, Windows Server 2008 or Red Hat Enterprise Linux 5) do usually have misaligned partitions, which is shown in the figure below:



In this example, clusters of the guest OS file system do not match the boundaries of NFS blocks, and reading from or writing to a cluster requires access to several NFS blocks. For example, reading from Cluster 1 causes two reads: from Block 1 and from Block 2. This results in a slower read time as compared to properly aligned partitions and leads to performance degradation.

12.13.1 Aligning Partitions

Basically, to align disks and partitions in virtual machines, you need to set an offset so that clusters in the guest OS file system match the volume block size on your NFS storage. Usually, the block size of most network storages is 512 bytes or a multiple of 512 bytes. As an example, the following sections describe the procedure of aligning disks and partitions for Linux and Windows virtual machines assuming that the size of your NFS blocks is 512 bytes.

When deciding on aligning disks and partitions, take into account that this process destroys all data on these disks and partitions. So if you want to have a correctly aligned system partition, you need to align your disks and partitions before creating a virtual machine and installing a guest operating system in it. If you do not want an aligned system partition, you can first create a virtual machine and install a guest OS in it, and then align your data disks from inside the virtual machine.

The sections below demonstrate how to align disks and partitions before you start installing a guest OS. You can, however, use a similar procedure to align data disks and partitions from inside your virtual machines.

12.13.2 Checking Partition Alignment in Existing Virtual Machines

First of all, you may wish to know how you can check that the partitions of a virtual machine are not aligned. Depending on the operating system installed in the virtual machine, you can do the following.

12.13.2.1 Linux Virtual Machines

To check the partition alignment in a Linux virtual machine, log in to this virtual machine and run the following command:

```
# fdisk -l -u /dev/device_name
```

For example, to check the partition alignment on the sdc device, you can run this command:

```
# fdisk -l -u /dev/sdc
Disk /dev/sdc: 73.0 GB, 73014444032 bytes
255 heads, 63 sectors/track, 8876 cylinders, total 142606336 sectors
Units = sectors of 1 * 512 = 512 bytes
   Device   Boot    Start        End    Blocks   Id  System
/dev/sdc1   *         63      208844    104391   83   Linux
/dev/sdc2             208845  142592939  71192047+  8e   Linux LVM
```

Pay attention to the number of sectors in the Start column. Usually, a sector contains 512 bytes, which

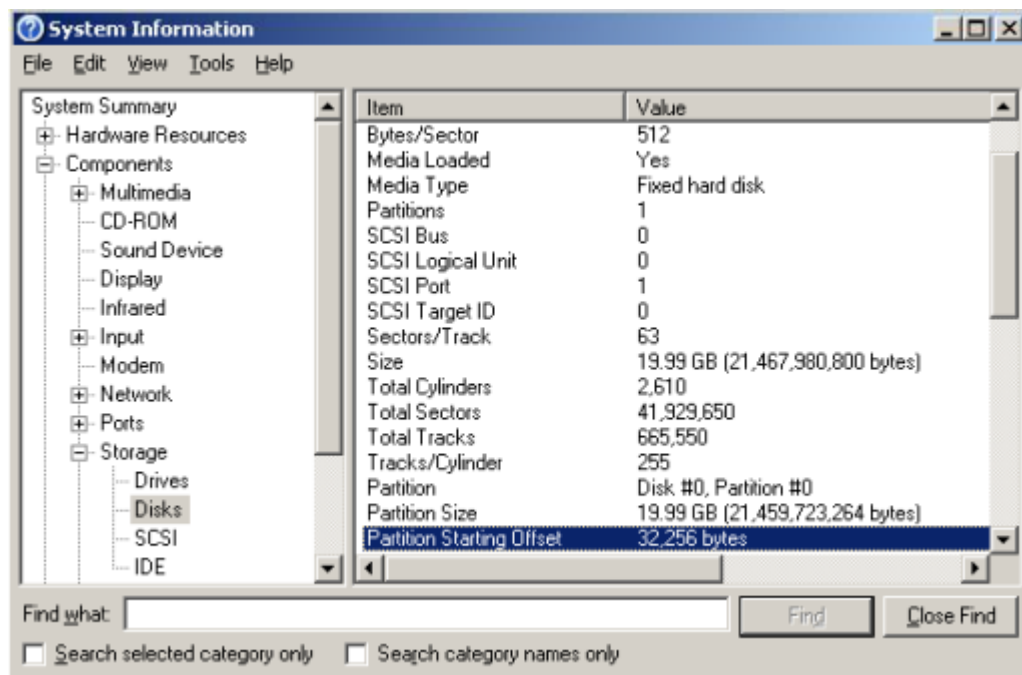
makes up 32256 bytes for 63 sectors for the `/dev/sdc1` partition and 26105625 bytes for 208845 for the `/dev/sdc2` partition. For a partition to be properly aligned, it must align with 4096 byte boundaries (assuming that the block size of your storage is 4 KB). As 32256 and 106928640 is not a multiple of 4096, the partitions `/dev/sdc1` and `/dev/sdc2` are not aligned properly. To align them, you should offset

- the `/dev/sdc1` partition by 1 sector so that it starts at 64. In this case, 64 sectors each containing 512 bytes make up 32768 that is a multiple of 4096.
- the `/dev/sdc2` partition by 3 sectors so that it starts at 208848. In this case, 208848 sectors each containing 512 bytes make up 106930176 that is a multiple of 4096.

12.13.2.2 Windows Virtual Machines

To check the partition alignment in a Windows virtual machine, do the following:

1. Click **Start** > **Run**, type `msinfo32.exe`, and press Enter to open System Information.
2. Navigate to Components > Storage > Disks, and look for the Partition Starting Offset field in the right part of the window.



To find out if the partition is aligned properly, use the method described above for Linux virtual machines.

12.13.3 Aligning Disks for Linux Virtual Machines

To align partitions for use in a Linux virtual machine, you need a working Linux virtual machine. Once you have it at hand, follow the steps below:

1. Create a new disk for the virtual machine. On this disk, you will create aligned partitions. Then you will connect the disk to a new virtual machine and install your Linux guest OS on this disk.
2. Start the virtual machine and log in to it using SSH.
3. Run the `fdisk` utility for the disk you want to align.
4. Create a primary partition, and set the starting block number for the created partition.
5. Repeat steps 3-4 to create and align all partitions you plan to have in your new virtual machine.

The following example creates partition #1 with the size of 1 GB on the `/dev/sda` device and uses the offset of 64 KB.

```
# fdisk /dev/sda
Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel
Building a new DOS disklabel. Changes will remain in memory only,
until you decide to write them. After that, of course, the previous
content won't be recoverable.
The number of cylinders for this disk is set to 1044.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
 1) software that runs at boot time (e.g., old versions of LILO)
 2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)
Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First sector (63-16777215, default 63): 64
Last sector or +size or +sizeM or +sizeK (64-16777215, default 16777215): 208848
Command (m for help): w
The partition table has been altered!
Calling ioctl() to re-read partition table.
Syncing disks.
```

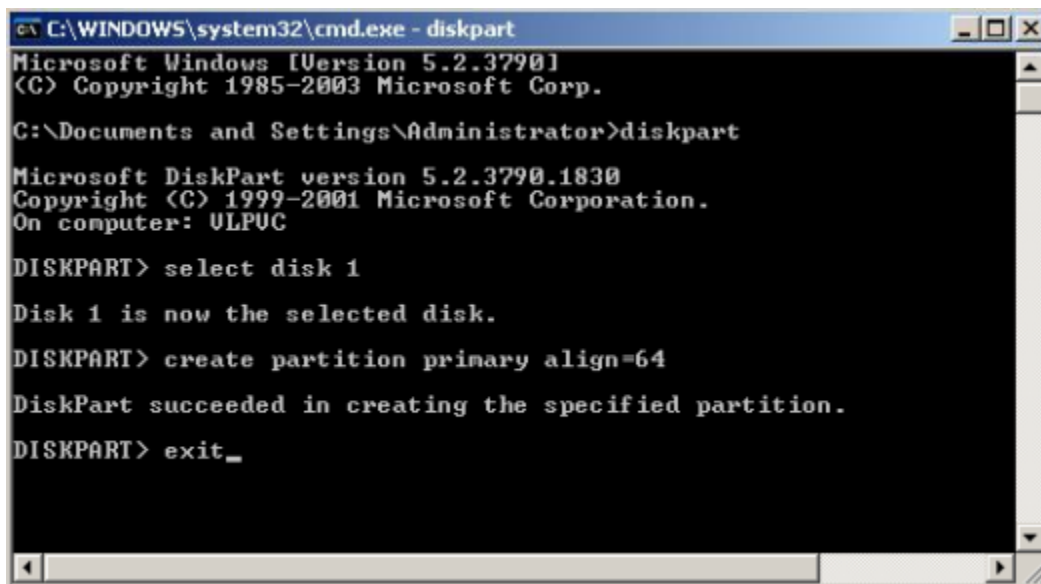
Once you align all the necessary partitions, disconnect the disk from the virtual machine. When creating a new virtual machine, choose this disk for use with this virtual machine.

12.13.4 Aligning Partitions for Windows Virtual Machines

To align a disk for a Windows virtual machine, you need a working Windows virtual machine. Once you have it at hand, you can use the `diskpart` or `diskpar` utility (depending on your operating system) to align the disk:

1. Create a new disk for the virtual machine. On this disk, you will create aligned partitions. Then you will connect the disk to a new virtual machine and install your Windows guest OS on this disk.
2. Open the command-line prompt, and run the `diskpart` or `diskpar` utility.
3. Select the disk you want to align.
4. Create the primary partition on the disk, and align it.
5. Exit the `diskpart` or `diskpar` utility, and close the command-line prompt.

The following example demonstrates how to use the `diskpart` utility to align disk 1 by setting the offset of 64 for it:

A screenshot of a Windows command prompt window titled "C:\WINDOWS\system32\cmd.exe - diskpart". The window shows the following text:

```
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\Documents and Settings\Administrator>diskpart

Microsoft DiskPart version 5.2.3790.1830
Copyright (C) 1999-2001 Microsoft Corporation.
On computer: ULPUC

DISKPART> select disk 1

Disk 1 is now the selected disk.

DISKPART> create partition primary align=64

DiskPart succeeded in creating the specified partition.

DISKPART> exit_
```

Once you align the virtual disk, disconnect it from the virtual machine. When creating a new virtual machine, choose this disk for use with this virtual machine.

12.13.5 Creating a Template of a Virtual Machine with Aligned Partitions

To facilitate the procedure of creating virtual machines that have aligned system partitions, you can create a template of the aligned virtual machine and deploy new virtual machines from this template.

For example, if you align a disk by following the steps in *Aligning Partitions for Windows Virtual Machines* on page 288, then create a new virtual machine that uses this disk, and then install Windows Server 2008 operating system in the virtual machine, you will have a clean Windows Server 2008 installation on the correctly aligned disk. Now you can create a template of this virtual machine and use this template each time you need to deploy a new virtual machine with Windows Server 2008.

12.14 Uninstalling Virtuozzo Guest Tools from Virtual Machines

If you find out that Virtuozzo guest tools are incompatible with some software inside a virtual machine, you can uninstall them from that VM. The steps you need to perform to remove guest tools differ depending on the guest operating system and are described in the sections below.

Note: Running virtual machines that do not have Virtuozzo guest tools installed cannot be configured from the physical server.

12.14.1 Uninstalling Guest Tools from Linux Virtual Machines

To uninstall Virtuozzo guest tools from a Linux guest, log in to the virtual machine and do as follows:

1. Remove the packages:

- On RPM-based systems (CentOS and other):

```
# yum remove dkms-vzvirtio_balloon prl_nettool qemu-guest-agent-vz vz-guest-udev
```

- On DEB-based systems (Debian and Ubuntu):

```
# apt-get remove vzvirtio-balloon-dkms prl-nettool qemu-guest-agent-vz vz-guest-udev
```

If any of the packages listed above are not installed on your system, the command will fail. In this case, exclude these packages from the command and run it again.

2. Remove the files:

```
# rm -f /usr/bin/prl_backup /usr/share/qemu-ga/VERSION /usr/bin/install-tools \  
/etc/udev/rules.d/90-guest_iso.rules /usr/local/bin/fstrim-static /etc/cron.weekly/fstrim
```

3. Reload the udev rules:

```
# udevadm control --reload
```

After removing Virtuozzo guest tools, restart the virtual machine.

Note: After Virtuozzo guest tools are removed from a virtual machine, their state is shown as possibly installed.

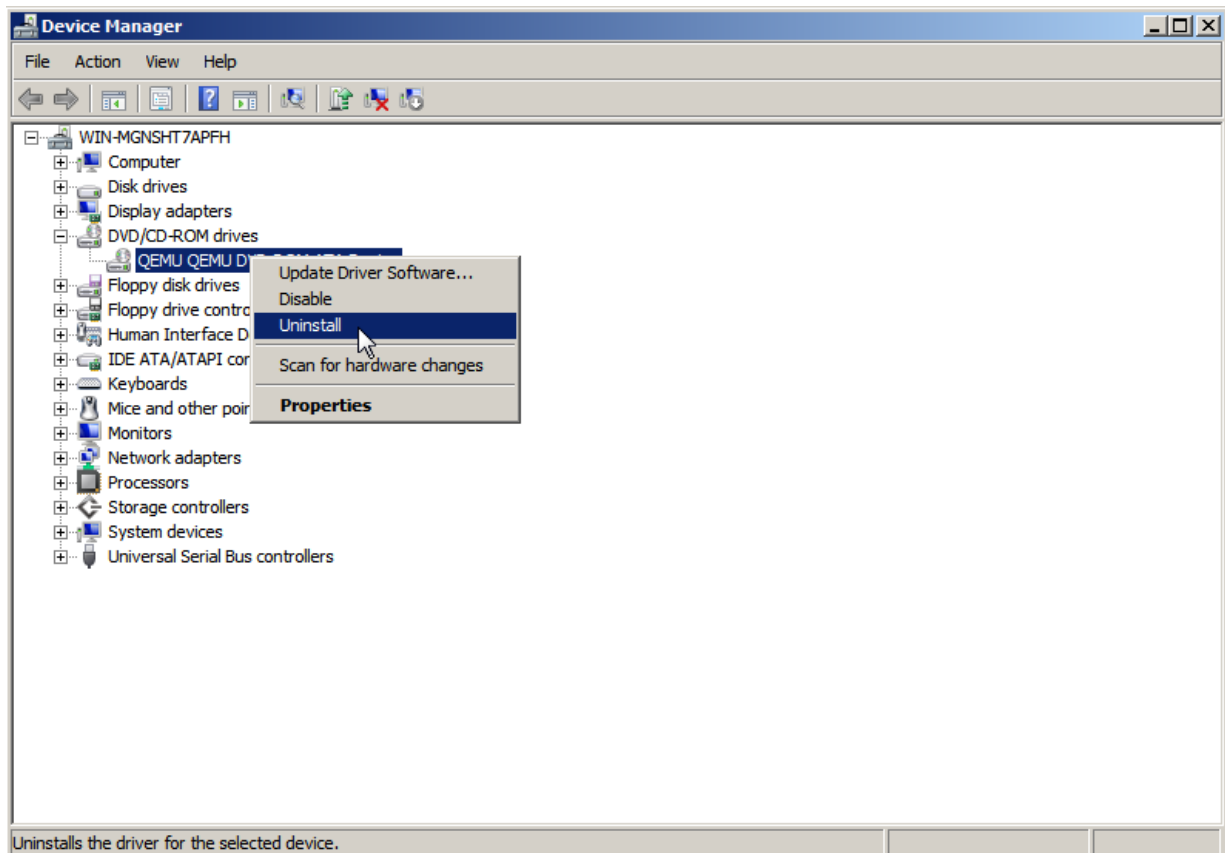
12.14.2 Uninstalling Guest Tools from Windows Virtual Machines

To uninstall Virtuozzo guest tools for Windows, log in to the virtual machine and do as follows:

1. Remove virtualized device drivers:

Important: Do not remove the VirtIO/SCSI hard disk driver and NetKVM network driver. Without the former, the virtual machine will not boot; without the latter, it will lose network connectivity.

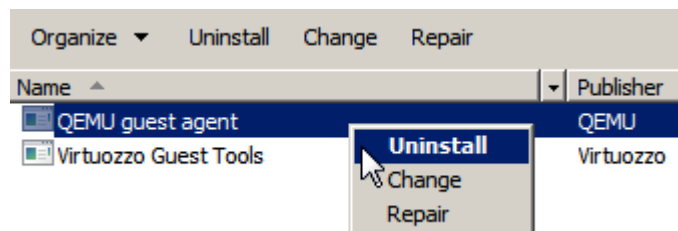
- 1.1. Open the **Device Manager**.
- 1.2. Double-click the device to expand the list of installed drivers.
- 1.3. Right-click the driver to be removed and select **Uninstall** from the drop-down menu.



2. Uninstall QEMU guest agent and Virtuozzo Guest Tools:

2.1. From the Windows **Start** menu, open **Control Panel > Programs > Programs and Features**.

2.2. Right-click **QEMU guest agent** and select **Uninstall** from the drop-down menu.



2.3. Right-click **Virtuozzo Guest Tools** and select **Uninstall** from the drop-down menu.

3. Stop and delete Guest Tools Monitor:

```
> sc stop VzGuestToolsMonitor
> sc delete VzGuestToolsMonitor
```

4. Unregister Guest Tools Monitor from Event Log:

```
> reg delete HKLM\SYSTEM\CurrentControlSet\services\eventlog\Application\VzGuestToolsMonitor
```

5. Delete the autorun registry key for RebootNotifier:

```
> reg delete HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run /v VzRebootNotifier
```

6. Delete the C:\Program Files\Qemu-ga\ directory.

If VzGuestToolsMonitor.exe is locked, close all the Event Viewer windows. If it remains locked, restart the eventlog service:

```
> sc stop eventlog
> sc start eventlog
```

After removing Virtuozzo guest tools, restart the virtual machine.

Note: After Virtuozzo guest tools are removed from a virtual machine, their state is shown as possibly installed.

12.15 Enabling Legacy VM Debug Mode

The legacy VM debug mode makes sure that legacy VMs that failed to convert to the Virtuozzo Hybrid Server 7 format during migration or restoration from backup are not deleted from the destination Virtuozzo Hybrid Server 7 server after failed conversion. With the debug mode enabled, such virtual machines remain stopped or running with disabled network to let the technical support team study the memory dump and find out the reason for failure.

If you are trying to migrate or restore backup of a legacy virtual machine to a Virtuozzo Hybrid Server 7 server and conversion stage fails, you can enable the debug mode on the destination Virtuozzo Hybrid Server 7 server, make another migration or restore attempt, send the problem report, and contact the technical support team.

To enable the legacy VM debug mode on the destination Virtuozzo Hybrid Server 7 server:

1. Stop the dispatcher:

```
# systemctl stop prl-disp.service
```

Note: While the dispatcher is stopped, you cannot manage and collect stats of VMs and containers.

Running VMs and containers are not stopped.

2. In `/etc/vz/dispatcher.xml`, set `LegacyVmUpgrade` to 1.
3. Start the dispatcher:

```
# systemctl start prl-disp.service
```

12.16 Installing Optional Packages

Virtuozzo Hybrid Server comes with everything you may need already installed. However, you can also install optional Virtuozzo Hybrid Server packages from remote repositories by means of the `yum` command.

Note: For more information on using `yum` in Virtuozzo Hybrid Server, see [Updating Manually](#) on page 186 and the `yum` manual page.

12.17 Enabling Nested Virtualization in Virtual Machines

Warning: This feature is experimental and tested only on Linux guests. Operation of nested virtual machines may be unstable.

Virtuozzo Hybrid Server supports nested Intel VT-x and AMD-V virtualization in virtual machines.

To permanently enable nested virtualization on the host, do the following:

1. Stop all running or paused virtual machines on the node.
2. Unload the required module from the kernel:

```
# rmmmod kvm_intel
```

Or

```
# rmmmod kvm_amd
```

3. Uncomment the line options `<module> nested=1` in `/etc/modprobe.d/kvm.conf` corresponding to the module.

4. Load the module again:

```
# modprobe kvm_intel
```

Or

```
# modprobe kvm_amd
```

5. Restart the libvirt daemon:

```
# systemctl restart libvirtd
```

6. Enable nested virtualization in the virtual machine:

```
# prlctl set MyVM --nested-virt on
```

Take note of the following:

1. A guest operating system in a nested virtual machine will not be able to obtain an IP address via DHCP if `ipfilter`, `macfilter`, and `preventpromisc` parameters of the host VM's bridged network adapter are set to `no`.
2. You cannot change CPU features mask for nested virtual machines using the `prlsrvctl set --cpu-features-mask` command.

12.18 Changing Server-wide Backup Configuration

You can turn off backup compression and connection tunneling globally for a Virtuozzo Hybrid Server 7 node. By default, these options are always turned on.

To turn off the compression and connection tunneling for a backup globally, run the following:

```
prlsrvctl set --backup-tunnel off
prlsrvctl set --backup-compression off
```

Running `prlsrvctl set --backup-tunnel off` turns off the connection tunneling for a backup and does not provide secure data transmission. Using `prlsrvctl set --backup-compression off` makes the compression of a backup image unavailable.

You can set both options independently. They are transparent to the application that creates a backup (for example, CLI or Virtuozzo Automator). These settings will apply to all backups launched from the application you use to create the backups.

We do not recommend turning off the backup compression and connection tunneling for a backup globally unless you need to make them unavailable by default. Setting these options rests entirely at your own risk.

Note: To avoid failure to create remote backups with disabled connection tunneling, turn off a firewall on the node involved in the backup operation:

```
# systemctl stop firewalld
# systemctl disable firewalld
```

The last command prevents the firewall from starting again the next time the node boots up.

12.19 Customizing the Message of the Day (MOTD)

Virtuozzo Hybrid Server can display an informative message, known as the message of the day (MOTD), when a user logs in to the server. As a root user, you can customize the MOTD according to your needs, for example, to show a greeting, important system information, current resource consumption, or something else.

The `vs-motd` service generates the message of the day from the contents of the `/etc/motd_user` and `/etc/motd` files. The steps to customize these files differ depending on whether you want the MOTD to show static or dynamic information. They are described in the following sections.

12.19.1 Setting a Custom Static MOTD

To set the MOTD to display static information, for example, a greeting, do the following:

1. Add the desired text to the `/etc/motd_user` file. For example:

```
echo 'Welcome to Virtuozzo Hybrid Server' >> /etc/motd_user
```

2. Restart the Virtuozzo Hybrid Server MOTD service:

```
systemctl restart vz-motd
```

On the next successful login, you will see the following static message of the day:

```
Welcome to Virtuozzo Hybrid Server
```

12.19.2 Setting a Custom Dynamic MOTD

To set the MOTD to display dynamic information, for example, the current Virtuozzo Hybrid Server release and kernel versions as well as license status, do as follows:

1. Create the `/etc/vz/ufup.d` directory and a shell script with a custom name in it, for example, `motd.sh`:

```
mkdir /etc/vz/ufup.d
vi /etc/vz/ufup.d/motd.sh
```

2. Add the following lines to the script:

```
#!/bin/bash

echo -e "
Welcome to `cat /etc/virtuozzo-release`
vzkernel: `uname -r`
License:
`vzlicview | grep -E "(status|expiration)"`
" >> /etc/motd
chmod 644 /etc/motd
```

3. Make the script executable:

```
chmod 755 /etc/vz/ufup.d/motd.sh
```

4. Restart the Virtuozzo Hybrid Server MOTD service:

```
systemctl restart vz-motd
```

On the next successful login, you can see a message of the day similar to the following:

```
Welcome to Virtuozzo Hybrid Server release 7.0.9 (469)
vzkernel: 3.10.0-862.20.2.vz7.73.25
License:
  status="ACTIVE"
  expiration="02/15/2019 02:59:59"
```

12.20 Setting Up RSA Authentication Between Nodes

Some operations on virtual environments, e.g., migration and backup to other nodes, require authenticating on remote servers running Virtuozzo Hybrid Server. It is typically done by means of passwords. If entering passwords is not an option, nodes can be authenticated with RSA keys.

Do the following:

1. Generate an identity for the user that will run commands on the remote node. An identity is a pair of key files: the private key `id_rsa` and the public key `id_rsa.pub`. Identities are stored in `<user_dir>/.vz/keys/`. For example, to generate an identity for the current user (typically root):

```
# mkdir -p ~/.vz/keys/
# ssh-keygen -f ~/.vz/keys/id_rsa -m pem
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.vz/keys/id_rsa.
Your public key has been saved in /root/.vz/keys/id_rsa.pub.
The key fingerprint is:
SHA256:Jf+I5HjipRFy1AxcG945ZrSVZCMkauhru40r1u9XEHQ root@node1.example.local
The key's randomart image is:
+---[RSA 2048]-----+
|  o.*oAo E          |
|.o * B.*.          |
|o.. 0 . = .        |
|o o o . +          |
| . . o G .         |
| . o = o o         |
| o. + * . .        |
|o+. . B           |
|o=0.oo+           |
+----[SHA256]-----+
```

2. Add the contents of the public key file `id_rsa.pub` to `<user_dir>/.vz/authorized_keys` on the remote node. For example, for the root user:

```
# cat ~/.vz/keys/id_rsa.pub | \
ssh <user>@<remote_node> 'mkdir -p ~/.vz/keys/ && cat >> ~/.vz/keys/authorized_keys'
```

Where `<remote_node>` is the remote node's IP address.

Note: To use SSH keys as well, generate them in the user's home directory on the local node and authorize them in the user's directory on the remote node same as RSA keys, e.g., `cat`

```
~/ .ssh/id_rsa.pub >> ~/ .ssh/authorized_keys.
```

Now the user authorized on the remote node can manage virtual environments on that node without having to enter the password. Some of the operations are:

- Creating and managing virtual environments remotely. For example:

```
# prlctl create remote_ve -l <user>@<remote_node>
# prlctl list remote_ve -l <user>@<remote_node>
UUID          STATUS      IP_ADDR      T  NAME
{6f18d2ce-<...>} stopped    -            VM remote_ve
```

- Backing up local virtual environments to remote nodes. For example:

```
# prlctl backup local_ve -s <user>@<remote_node>
# prlctl backup-list -s <user>@<remote_node>
ID          Backup_ID      Node          Date          Type          Size
{fc0106cb-<...>} {b1edf157-<...>} remote_node  02/28/2021
```

- Setting the default backup location on a remote node. For example:

```
# prlsrvctl set --backup-storage <user>@<remote_node>
```

Note: The use of passwords in the `prlsrvctl set --backup-storage` command is dropped in Virtuozzo Hybrid Server 7.5 Update 1 to avoid keeping passwords in the file system.

- Backing up remote virtual environments to the local node. For example:

```
# cat ~/.vz/keys/id_rsa.pub >> ~/.vz/keys/authorized_keys
# prlctl backup remote_ve -l <user>@<remote_node> -s <user>@<local_node>
```

Where `<local_node>` is the local node's IP address. It must be reachable from the remote node.

Adding the local user's public key to the local node's authorized keys is required to avoid being asked for the password, because the local node's IP address is used instead of `localhost`.

- Migrating virtual environments. For example:

```
# prlctl migrate local_ve <user>@<remote_node>
# prlctl migrate <user>@<remote_node>/remote_ve <user>@<local_node>
```


12.21 Switching Ploop I/O Engines

A ploop I/O engine is a driver that maps block device requests (REQ_XXX) to operations on an image file.

Virtuozzo Hybrid Server supports two ploop I/O engines: **io_kaio** and **io_direct**. The former, **io_kaio**, is used by default for virtual disk images on both local ext4 file systems and Virtuozzo Storage FUSE file systems. The latter, **io_direct**, is deprecated and will be dropped in the future versions of Virtuozzo Hybrid Server.

The advantage of **io_kaio** compared to **io_direct** is in the better usage of storage space on the host file system achieved by the default discard granularity that is a factor of 4 KB versus the default 1 MB of **io_direct**. In other words, by default **io_kaio** makes it possible to free up blocks smaller than 1 MB from mounted virtual disks.

The disadvantage of **io_kaio** compared to **io_direct** is in the increasing defragmentation of host physical disks. It reduces disk write speeds, as each request needs to be split into smaller fragments. It also leads to higher memory usage, because fragment data is cached, and the more fragments, the more memory is needed for the cache. This disadvantage affects HDDs much more than SSDs.

For the **io_kaio** ploop engine, you can manually switch the discard granularity between 4 KB and 1 MB for ploop images on both local ext4 and Virtuozzo Storage FUSE file systems. In addition to the pros and cons of each discard granularity mentioned early in this section, it is recommended to use 4 KB for ploop images residing on SSD drives and 1 MB for ploops on HDD drives.

To change the discard granularity:

1. Set the desired value in `/etc/vz/ploop.conf`. For example:

```
EXT4_DISCARD_GRANULARITY=4096
FUSE_DISCARD_GRANULARITY=1048576
```

Warning: Only use the values 4096 and 1048576.

2. As this parameter is read at ploop mount, re-mount the ploop images, e.g., by restarting the virtual environments.

The newly created and started virtual environments will use the new discard granularity as well.

The discard granularity currently used by a mounted ploop image is reported in `/sys/block/ploop<N>/ptune/discard_granularity`. For example:

```
# cat /sys/block/ploop64959/ptune/discard_granularity
1048576
```

If you need to switch to the **io_direct** ploop engine, do the following:

1. Set `USE_KAIO_FOR_EXT4=no` in `/etc/vz/ploop.conf`.
2. As this parameter is read at ploop mount, re-mount the ploop images, e.g., by restarting the virtual environments.

The newly created and started virtual environments will use the new ploop I/O engine as well.

To switch back to **io_kaio**, set `USE_KAIO_FOR_EXT4=yes` in `/etc/vz/ploop.conf` and remount the ploop images.

The engine currently used by a mounted ploop image is reported in `/sys/block/ploop<N>/pdelta/<highest_number>/io`. For example:

```
# cat /sys/block/ploop61324/pdelta/0/io
kaio
```

12.22 Managing Windows Containers

Important: This feature is a technical preview. It is not ready for production.

You can use Virtuozzo Hybrid Server to create and manage [Windows containers](#) on Microsoft Windows nodes.

12.22.1 Setting Up the Environment

You will need the following:

- A node running Virtuozzo Hybrid Server 7.5 or newer.
- Three or more Microsoft Windows nodes in a [Storage Spaces Direct](#) cluster. The nodes must be running Microsoft Windows Server 2019 v1809 or newer. It is also implied that the nodes meet the [Storage Spaces Direct hardware requirements](#).
- A network interconnecting the Virtuozzo Hybrid Server node and Windows nodes. All Microsoft Windows nodes will need Internet access for Docker installation.

12.22.1.1 Configuring the Virtuozzo Hybrid Server Node

1. Install the required packages:

```
# yum groupinstall "Windows Containers"
# yum install virt-install
```

2. Create a temporary directory, e.g. /home/tmp/, and generate certificates in it.

- The CA certificate:

```
# /usr/share/winrm-docker-deploy/create-certs.sh -m ca \
-t /home/tmp/ -pw <cert_passwd>
```

Where <cert_passwd> is a password for the certificate.

- The client certificate:

```
# /usr/share/winrm-docker-deploy/create-certs.sh -m client \
-t /home/tmp/ -us <windows_admin> -pw <cert_passwd>
```

Where <windows_admin> is the username of an account that is in the Administrators group on every cluster node.

- A certificate for every node in the Storage Spaces Direct cluster:

```
# /usr/share/winrm-docker-deploy/create-certs.sh -m server \
-h <hostname> -hip <hostip> -t /home/tmp/ -pw <cert_passwd>
```

Where <hostname> and <hostip> are the host name and IP address of a cluster node, respectively.

Run this command for each cluster node.

For example, if you have three nodes in the Storage Spaces Direct cluster: win1.example.com, win2.example.com, and win3.example.com, you will get these certificates as a result:

```
# ls -l /home/tmp/
cakey.pem
ca.pem
ca.srl
cert.pem
key.pem
win1.example.com-https.pfx
win1.example.com-server-cert.pem
win1.example.com-server-key.pem
win2.example.com-https.pfx
win2.example.com-server-cert.pem
win2.example.com-server-key.pem
win3.example.com-https.pfx
win3.example.com-server-cert.pem
```

```
win3.example.com-server-key.pem
```

Note: Non-root users may require additional permissions to read these files.

- Copy the CA certificate as well as the client public and private certificates to `/etc/pki/libvirt/docker/`:

```
# cp /home/tmp/{ca,cert,key}.pem /etc/pki/libvirt/docker/
```

12.22.1.2 Configuring Microsoft Windows Nodes

Perform the following steps on each Windows Server node in the Storage Spaces Direct cluster:

- Install the **Containers** and **Windows Server Backup** features. For example, from a PowerShell session run as administrator:

```
PS C:\> Install-WindowsFeature -Name Containers, Windows-Server-Backup
```

- Reboot the node.

- Download the deployment bundle and unpack it to a temporary directory, e.g.,

`C:\ProgramData\Virtuozzo\docker-bundle\`. For example:

```
PS C:\> (New-Object System.Net.WebClient).DownloadFile( `
  "https://docs.virtuozzo.com/files/winct/docker-deploy-latest.zip", `
  "C:\docker-deploy-latest.zip")
```

```
PS C:\> Expand-Archive -LiteralPath C:\docker-deploy-latest.zip `
  -DestinationPath C:\ProgramData\Virtuozzo\docker-bundle\
```

You will get the following files as a result:

```
PS C:\> ls C:\ProgramData\Virtuozzo\docker-bundle\

Directory: C:\ProgramData\Virtuozzo\docker-bundle\

Mode                LastWriteTime         Length Name
----                -
d-----           11/26/2020    8:03 AM          bindata
d-----           11/26/2020    8:03 AM          certs
d-----           11/26/2020    8:03 AM          vz-windows-backup
-a----           11/25/2020    3:25 PM        38131 auto-deploy.ps1
-a----           11/25/2020    3:25 PM         140 bundle-version.txt
-a----           11/25/2020    3:25 PM        27905 winct-uninstall.ps1
```

- Copy the CA and node's certificates from the Virtuozzo Hybrid Server node to

C:\ProgramData\Virtuozzo\docker-bundle\certs. Remove the hostname from file names. For example:

```
PS C:\> scp root@10.94.106.235:/etc/pki/libvirt/docker/ca.pem `
C:\ProgramData\Virtuozzo\docker-bundle\certs\
root@10.94.106.235's password:
ca.pem      100% 1781      1.7KB/s   00:00
```

```
PS C:\> scp root@10.94.106.235:/home/tmp/$(`
[System.Net.Dns]::GetHostByName($env:COMPUTERNAME).hostname)* `
C:\ProgramData\Virtuozzo\docker-bundle\certs\
root@10.94.106.235's password:
win1.example.com-https.pfx      100% 4021      3.9KB/s   00:00
win1.example.com-server-cert.pem 100% 1777      1.7KB/s   00:00
win1.example.com-server-key.pem 100% 3243      3.2KB/s   00:00
```

```
PS C:\> ls C:\ProgramData\Virtuozzo\docker-bundle\certs\ | Rename-Item -NewName `
{ $_.Name -replace $('([System.Net.Dns]::GetHostByName($env:COMPUTERNAME).hostname+"-"),'' }`
```

You will get the following files as a result:

```
PS C:\> ls C:\ProgramData\Virtuozzo\docker-bundle\certs\

Directory: C:\ProgramData\Virtuozzo\docker-bundle\certs\

Mode                LastWriteTime         Length Name
----                -
-ar---             11/4/2020  11:42 AM           1781 ca.pem
-a---              11/4/2020  11:41 AM           4021 https.pfx
-ar---             11/4/2020  11:41 AM           1777 server-cert.pem
-ar---             11/4/2020  11:41 AM           3243 server-key.pem
```

5. Run the deployment script:

```
PS C:\> cd C:\ProgramData\Virtuozzo\docker-bundle\
PS C:\> .\auto-deploy.ps1 -username:<windows_admin> `
-pwforpfx:<cert_passwd> -password:<windows_admin_passwd>
```

Where <windows_admin> and <windows_admin_passwd> are the username and password of an account that is in the Administrators group on every cluster node and <cert_passwd> is the certificate password. For simplicity, use the same administrator account that you generated the certificates with.

The deployment script will download and install Docker and required components, including a base Nano Server 1809 image. It will also replace the original dockerd.exe with a custom one made by Virtuozzo. If Docker is already installed, the script will update it and replace the executable. The script will also add firewall rules and import the certificates. The log will be saved to C:\ProgramData\Virtuozzo\winctfiles\auto-deploy.log.

Note: If you only need to import or update the certificates, use the deployment script with the `-onlycerts` option. For more options, run `.\auto-deploy.ps1 -help`.

After the script finishes, reboot the node and check that the Docker client and server are running. For example:

```
PS C:\> docker version
Client: Docker Engine - Enterprise
Version:      19.03.12
API version:  1.40
Go version:   go1.13.13
Git commit:   4306744
Built:        08/05/2020 19:27:53
OS/Arch:     windows/amd64
Experimental: false

Server:
Engine:
Version:      19.03.13
API version:  1.40 (minimum version 1.24)
Go version:   go1.13.15
Git commit:   cb7af2e
Built:        Tue Oct 13 12:15:38 2020
OS/Arch:     windows/amd64
Experimental: false
```

If an error is shown for the server, try starting it manually:

```
PS C:\> Start-Service docker
```

12.22.2 Creating Windows Containers

After setting up both the Virtuozzo Hybrid Server and Microsoft Windows Server nodes, you can create Windows containers using `virt-install`. Container base images need to be present on Windows Server nodes. The deployment script only installs the Nano Server 1809 image. Any other base images need to be pulled from the Docker Hub first.

You can create a Windows container from the Virtuozzo Hybrid Server node as follows:

```
# virt-install --connect docker://<hostname> --name <ct_name> --memory 1024 \
--boot init=* --os-variant none --console none --disk bus=virtio,path=<ct_image> \
--network type=network,mac=<ct_MAC>,source=<net_name> \
--filesystem <guest_dir>,<host_dir> --noreboot --noautoconsole
```

Where:

- <hostname> is the hostname of the Windows Server node to create the container on.
- <ct_name> is a unique container name.
- <ct_image> is a base image to create the container from. For example, `mcr.microsoft.com/windows/nanoserver:1809`.
- <ct_MAC> is a unique container MAC address.
- <net_name> is a name of a Docker network to connect the container to. Use `default` for the default NAT network.
- <guest_dir> and <host_dir> are the container guest and host directories located on a virtual disk in the Storage Spaces Direct cluster. Create these directories if they do not exist.

For example:

```
# virt-install --connect docker://win1.example.com --name winct1 --memory 1024 \
--boot init=* --os-variant none --console none \
--disk bus=virtio,path=mcr.microsoft.com/windows/nanoserver:1809 \
--network type=network,mac=00:01:02:03:04:05,source=default \
--filesystem C:/ClusterStorage/disk1/guest_dir/,C:/ClusterStorage/disk1/host_dir/ \
--noreboot --noautoconsole

Starting install...
Domain creation completed.
You can restart your domain by running:
virsh --connect docker://win1.example.com start winct1
```

This command will create a Windows container based on a Nano Server image.

As more examples for this technical preview, you can also create Windows containers with DotNetNuke and WordPress from base images published by Virtuozzo at Docker Hub. These base images include the required database management systems, so you can skip to setting up DotNetNuke and WordPress once you run the container. Do the following:

1. Pull the images from Docker Hub to Windows Server nodes:

```
PS C:\> docker pull virtuozzo/dnn:techpreview
```

Or

```
PS C:\> docker pull virtuozzo/wordpress:techpreview
```

2. Create the Windows container from the Virtuozzo Hybrid Server node. For example:

```
# virt-install --connect docker://win1.example.com --name winct1dnn --memory 2048 \
--boot init=* --os-variant none --console none \
--disk bus=virtio,path=virtuozzo/dnn:techpreview \
--network type=network,mac=00:01:02:03:04:06,source=default \
--filesystem C:/ClusterStorage/disk1/guest_dir/,C:/ClusterStorage/disk1/host_dir/ \
--noreboot --noautoconsole
# virsh --connect docker://win1.example.com start winct1dnn
```

Provide A DotNetNuke container with at least 2GB of memory for the setup to complete.

Or

```
# virt-install --connect docker://win1.example.com --name winct1wp --memory 1024 \
--boot init=* --os-variant none --console none \
--disk bus=virtio,path=virtuozzo/wordpress:techpreview \
--network type=network,mac=00:01:02:03:04:07,source=default \
--filesystem C:/ClusterStorage/disk1/guest_dir/,C:/ClusterStorage/disk1/host_dir/ \
--noreboot --noautoconsole
# virsh --connect docker://win1.example.com start winct1wp
```

3. Find out the container's IP address. For example:

```
# virsh --connect docker://win1.example.com dumpxml winct1dnn | grep "ip address"
<ip address='192.168.175.204' family='ipv4' prefix='24'/>
```

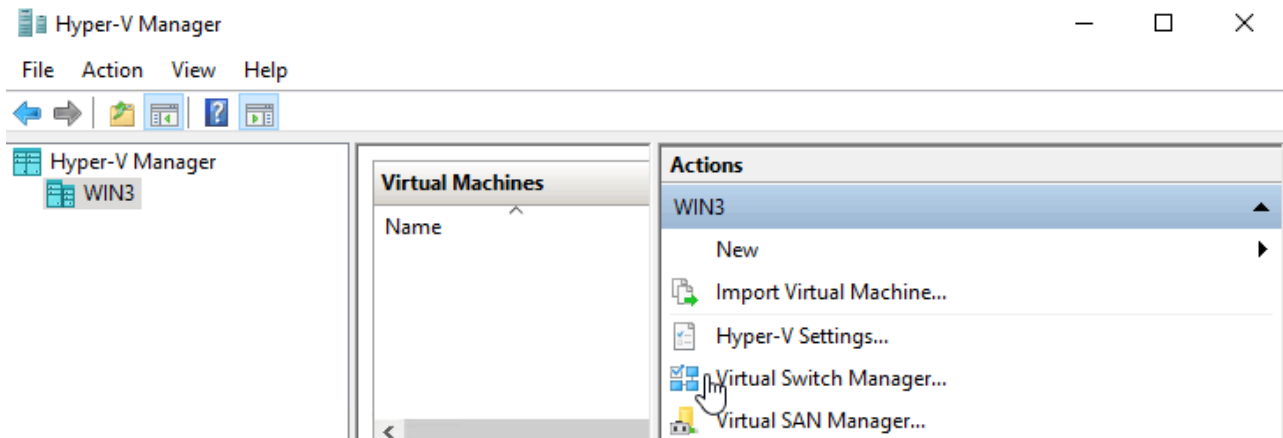
Or

```
# virsh --connect docker://win1.example.com dumpxml winct1wp | grep "ip address"
<ip address='192.168.175.114' family='ipv4' prefix='24'/>
```

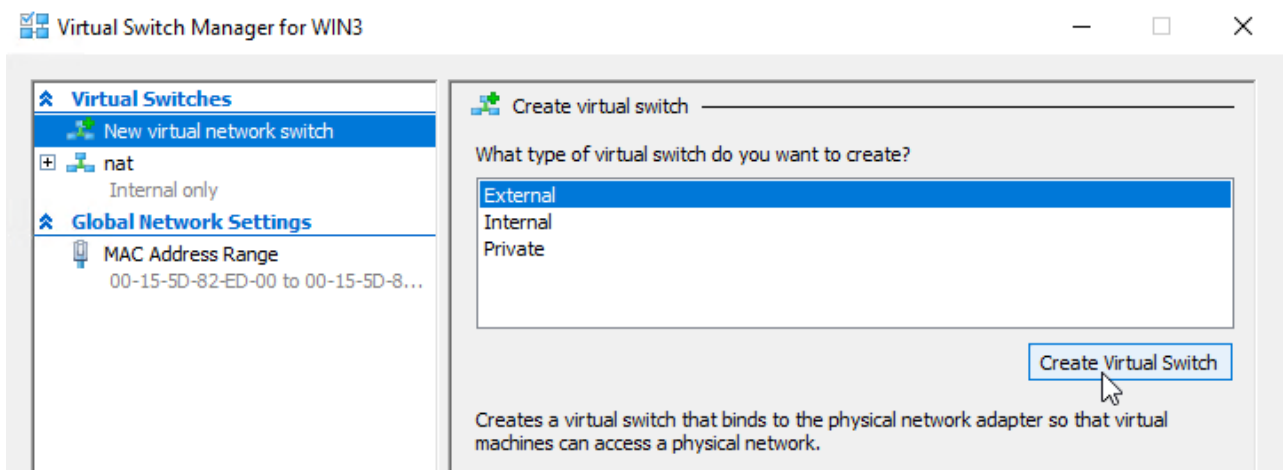
4. As the Windows containers use a NAT network by default, they are only accessible from the Windows Server node the container is running on. From that node, visit the container's IP address (and optionally port) in a web browser to proceed with setup.

To make Windows containers publicly accessible, let them use a virtual switch bound to a node's physical network adapter connected to a public network. To create a virtual switch, do the following:

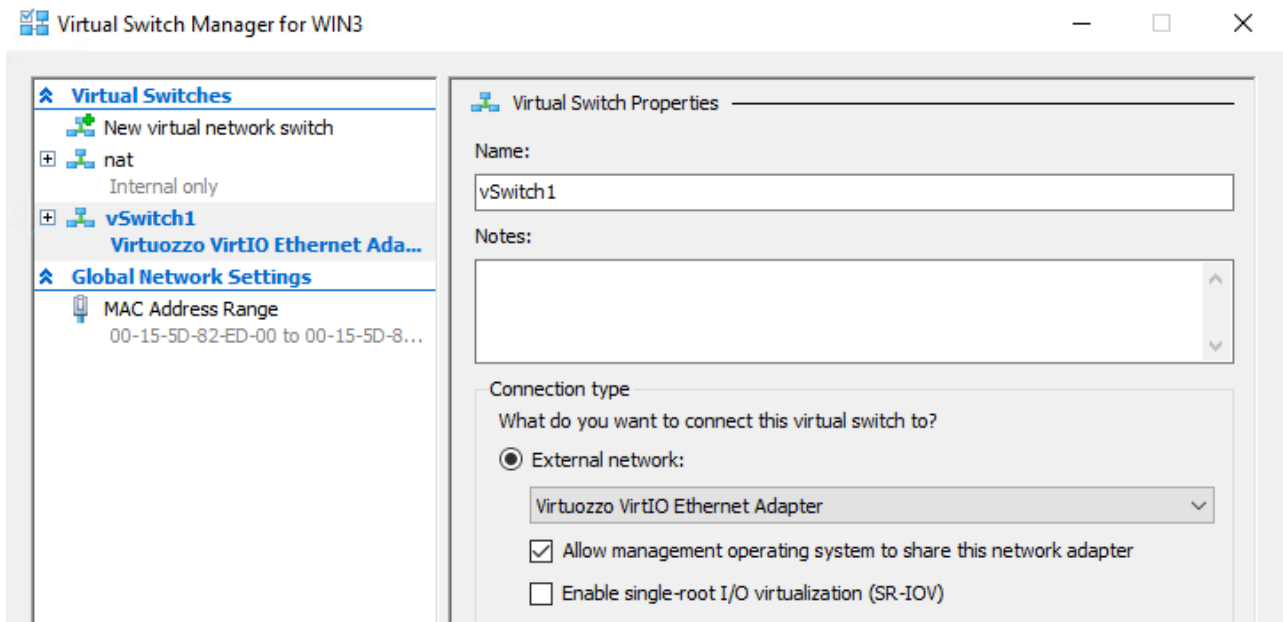
1. Open the Hyper-V Manager. E.g., in Server Manager, click **Tools > Hyper-V Manager** or run `virtmgmt.msc` from the Command Prompt or PowerShell.
2. In the Hyper-V Manager, click **Virtual Switch Manager** in the **Actions** menu.



3. In the Virtual Switch Manager, choose **New virtual network switch** > **External**. Click **Create Virtual Switch**.



4. In **Virtual Switch Properties**, enter a name for the switch, e.g., **vSwitch1**, and select a public physical network adapter in **External network**. Click **OK**.



- Restart Docker:

```
PS C:\> Restart-Service docker
```

- Check that the switch is listed in Docker networks. For example:

```
PS C:\> docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
ad2c3ef4de3e       nat                 nat                 local
ea6c5aa76c42       none               null                local
7ae7bd5c638e       vSwitch1           transparent         local
```

Now you can specify the public network, e.g., `--network ... source=vSwitch1` instead of `--network ... source=default`, when creating Windows containers with `virt-install`.

12.22.3 Starting, Listing, and Stopping Windows Containers

After creating a Windows container, start it as indicated by `virt-install`. For example:

```
# virsh --connect docker://win1.example.com start winct1
Domain winct1 started
```

Using `virsh`, you can also list and shut down Windows containers on the node. For example:

```
# virsh --connect docker://win1.example.com list --all
Id      Name      State
-----
39370   winct1   running
```

```
# virsh --connect docker://win1.example.com shutdown winct1
```

12.22.4 Running Commands in Windows Containers

To attach to a Windows container in this technical preview, use the `docker attach` command on the Windows Server node the container is running on.

To run a command inside a Windows container, use the `docker exec` command as follows:

```
PS C:\> docker exec <ct> <shell> /c "<command>"
```

Where:

- `<ct>` is the container name or ID.
- `<shell>` is the name of the shell bundled with the container image, e.g., `cmd` is Command Prompt and `pwsh` is PowerShell.
- `<command>` is the command to execute in the shell.

For example:

```
PS C:\> docker exec winct1 cmd /c "echo Test > C:\ClusterStorage\disk1\host_dir\test.txt"
```

```
PS C:\> docker exec winct1 cmd /c "type C:\ClusterStorage\disk1\host_dir\test.txt"
Test
```

```
PS C:\> docker exec winct1 cmd /c "dir C:\ClusterStorage\disk1\host_dir\"
Volume in drive C has no label.
Volume Serial Number is 34FA-4299

Directory of C:\ClusterStorage\disk1\host_dir

11/05/2020  03:04 AM    <DIR>          .
11/05/2020  03:04 AM    <DIR>          ..
11/05/2020  03:01 AM                9 test.txt
                1 File(s)          9 bytes
                2 Dir(s)  544,967,794,688 bytes free
```

12.22.5 Configuring Windows Container Parameters

In this technical preview, redefining container configuration (editing its XML scheme) is not supported. You can, however, update container resource limits as follows:

- Change available memory using `virsh setmem`. For example:

```
# virsh -c docker://win1.example.com setmem winct1 512M
# virsh -c docker://win1.example.com dumpxml winct1 | grep -i memory
<memory unit='KiB'>524288</memory>
<currentMemory unit='KiB'>524288</currentMemory>
```

- Edit the CPU shares or quotas for the container using `virsh schedinfo`. For example:

```
# virsh -c docker://win1.example.com schedinfo winct1 cpu_shares=1000
Scheduler      : posix
cpu_shares     : 1000
vcpu_period    : 0
vcpu_quota     : 0
```

Or

```
# virsh -c docker://win1.example.com schedinfo winct1 vcpu_period=4000 vcpu_quota=2000
Scheduler      : posix
cpu_shares     : 0
vcpu_period    : 4000
vcpu_quota     : 2000
```

You cannot set both `cpu_shares` and `vcpu_quota` at the same time.

Set `cpu_shares` or `vcpu_quota` or to `-1` to reset it:

```
# virsh -c docker://win1.example.com schedinfo winct1 cpu_shares=-1
```

Or

```
# virsh -c docker://win1.example.com schedinfo winct1 vcpu_quota=-1
```

- Throttle container disk with `virsh blkdeviotune` by changing the total throughput limit, in bytes per second, as well as the total I/O operations limit per second. For example:

```
# virsh -c docker://win1.example.com blkdeviotune winct1 vda \
--total-bytes-sec=104857600 --total-iops-sec=1000
```

Use `virsh domblklist` to find out container's device names.

12.22.6 Migrating Windows Containers

Take note of the following:

- The source container must be stopped.
- The name and UUID of the migrated container must be unique on the destination node.

- The migrated container keeps the source's UUID.
- The MAC address of the migrated container must be unique within the destination node's network. For example, if all nodes belong to the same network, MAC addresses must be unique across all nodes.
- The data on container's drive C: will be lost after migration. The data on persistent volumes will remain.

To migrate a Windows container, use the `migrate` subcommand of `virsh`. For example:

- Migrate a container and delete the source:

```
# virsh --connect docker://node3.winctpreview.com migrate winct1 --desturi \
docker://node1.winctpreview.com --offline --persistent --undefinesource
```

- Migrate a container under a new name and delete the source:

```
# virsh --connect docker://node3.winctpreview.com migrate winct1 --desturi \
docker://node1.winctpreview.com --offline --persistent --undefinesource \
--dname winct1_migrated
```

- Migrate a container under a new name and keep the source:

```
# virsh --connect docker://node3.winctpreview.com migrate winct1 --desturi \
docker://node1.winctpreview.com --dname winct1_migrated --offline --persistent
```

Note: In this case, the container can only be migrated once. On the second attempt, the UUIDs of the source container and the one migrated earlier will be the same, which will cause a conflict. You can, however, create a container with a new UUID from the source's configuration file and migrate it.

12.22.7 Managing Windows Container Backups

It is recommended to use the Windows Server Backup feature for container backups. You can use the provided PowerShell scripts to partially automate backup-related tasks.

Run the script on each node to find the containers with persistent CSVs and add those volumes to the backup schedule:

```
PS> Config-Docker-Backup.ps1 -Times '<1st> <2nd> <3rd>'
```

The script will configure the Windows Server Backup feature to create incremental backups daily at <1st>, <2nd>, and <3rd>. If you omit the `-Times` parameter, backups will be created at 9:00, 12:00, and 15:00.

Container backups will be listed in the Windows Server Backup snap-in. To open it, run `wbadmin.msc`.

To restore a container backup, run

```
PS> Restore-Docker .ps1 <ct_name>
```

Where <ct_name> is container name.

Restoring will overwrite the backed up container volume(s).

12.22.8 Troubleshooting Windows Containers

In case of issues with Windows containers, you can create a problem report and send it to the technical support team:

```
usage: vz-problem-report [-h] [-v] [--proxy PROXY] [--name NAME]
                        [--email EMAIL] [--description DESC]
                        [--reportfile FILE] [--version]
                        address
```

Where

- address is the hostname or IP address of the Microsoft Windows Server host to collect the report from
- name and email are your name and email address, respectively
- desc is the description of the problem
- reportfile is the name of the report file to save locally

If the node is not connected to the Internet, you can create a local report file with the --reportfile option and manually email it to the support team.

12.22.9 Uninstalling the Windows Containers Feature

To remove the ability to create and manage Windows Containers on Microsoft Windows Server nodes from a Virtuozzo Hybrid Server node, run the uninstallation script:

```
PS C:\> C:\Program Files\Virtuozzo\winctfiles\winct-uninstall.ps1
```

The script will attempt to undo changes made by the deployment script:

- Uninstall Docker if it was not installed before the deployment. If Docker was already installed, leave it with the custom dockerd.exe but restore its service's original startup type (e.g., **Automatic** or **Manual**). Stop the service if the original startup type is **Manual**.

- Uninstall DockerMsftProvider.
- Delete joblib, Prometheus exporter, and backup scripts.
- Remove the path to the bundle from PATH.
- Restore the original startup type of the WinRM service. Stop the service if the original startup type is **Manual**.
- Delete the firewall rules for Docker and WinRM. If rules existed and were changed, restore them.
- Delete the imported certificate files from ProgramDatadockercerts.d. Delete the certificates from the store. Kill the HTTP Listener in WinRM and remove authentication by the imported certificate.

CHAPTER 13

Troubleshooting

This chapter provides the information about problems that may occur during your work with Virtuozzo Hybrid Server and suggests the ways to solve them.

13.1 General Considerations

The general issues to take into consideration when troubleshooting your system are listed below. You should read them carefully before trying to solve more specific problems.

- You should always remember where you are currently located in your terminal. Check it periodically using the `pwd`, `hostname`, `ifconfig`, `cat /proc/vz/veinfo` commands. One and the same command executed inside a virtual machine or container and on the hardware node can lead to very different results. You can also set up the `PS1` environment variable to show the full path in the `bash` prompt. To do this, add these lines to `/root/.bash_profile`:

```
PS1="[\\u@\\h \\w]$ "  
export PS1
```

- If the hardware node slows down, use `vmstat`, `ps` (`ps axfw`), `dmesg`, `htop` (`vztop`) to find out what is happening, never reboot the machine without investigation. If no thinking helps restore the normal operation, use the `Alt+SysRq` sequences to dump the memory (`showMem`) and processes (`showPc`).
- Do not run any binary or script that belongs to a container directly from the hardware node, for example, do not ever do this:

```
# cd /vz/root/99/etc/init.d  
./httpd status
```

Any script inside a container could have been changed to whatever the container owner chooses: it

could have been trojaned, replaced to something like `rm -rf`, etc. You can use only `prlctl exec/prlctl enter` to execute programs inside a container.

- Do not use init scripts on the hardware node. An init script may use `killall` to stop a service, which means that all similar processes will be killed in all containers. You can check `/var/run/Service.pid` and kill the correspondent process explicitly.
- You must be able to detect any rootkit inside a container. It is recommended to use the `chkrootkit` package for detection (you can download the latest version from <http://www.chkrootkit.org>), or at least run

```
# rpm -Va|grep "S.5"
```

To check if the MD5 sum has changed for any RPM file.

You can also run `nmap`, for example:

```
# nmap -p 1-65535 192.168.0.1
Starting nmap V. 2.54BETA22 ( www.insecure.org/nmap/ )
Interesting ports on (192.168.0.1):
(The 65531 ports scanned but not shown below are in
state: closed)
Port      State      Service
21/tcp    open      ftp
22/tcp    open      ssh
80/tcp    open      http
111/tcp   open      sunrpc
Nmap run completed -- 1 IP address (1 host up) scanned
in 169 seconds
```

To check if any ports are open that should normally be closed.

That could however be a problem to remove a rootkit from a container and make sure it is 100% removed. If you're not sure, create a new container for that customer and migrate his/her sites and mail there.

- Check the `/var/log/` directory on the hardware node to find out what is happening on the system. There are a number of log files that are maintained by the system and Virtuozzo Hybrid Server (the `boot.log`, `messages`, etc.), but other services and programs may also put their own log files here depending on your distribution of Linux and the services and applications that you are running. For example, there may be logs associated with running a mail server (the `maillog` file), automatic tasks (the `cron` file), and others. However, the first place to look into when you are troubleshooting is the `/var/log/messages` log file. It contains the boot messages when the system came up as well as other status messages as the system runs. Errors with I/O, networking, and other general system errors are reported in this file. So, we recommend that you read to the `messages` log file first and then proceed

with the other files from the `/var/log/` directory.

- Subscribe to bug tracking lists. You should keep track of new public DoS tools or remote exploits for the software and install them into containers or at hardware nodes.
- When using `iptables`, there is a simple rule for Chains usage to help protect both the hardware node and its containers:
 - Use INPUT, OUTPUT to filter packets that come in/out the hardware node.
 - Use FORWARD to filter packets that are designated for containers

13.2 Kernel Troubleshooting

13.2.1 Using ALT+SYSRQ Keyboard Sequences

Press ALT+SYSRQ+H and check what is printed at the hardware node console, for example:

```
SysRq: unRaw Boot Sync Unmount showPc showTasks showMem loglevel0-8 tErm kIll \  
killalL Calls Oops
```

This output shows you what ALT+SYSRQ sequences you may use for performing this or that command. The capital letters in the command names identify the sequence. Thus, if there are any troubles with the machine and you're about to reboot it, please use the following key sequences before pressing the Power button:

- ALT+SYSRQ+M to dump memory info
- ALT+SYSRQ+P to dump processes states
- ALT+SYSRQ+S to sync disks
- ALT+SYSRQ+U to unmount file systems
- ALT+SYSRQ+L to kill all processes
- ALT+SYSRQ+U try to unmount once again
- ALT+SYSRQ+B to reboot

If the server is not rebooted after that, you can press the Power button.

13.2.2 Saving Kernel Faults (OOPS)

You can use the following command to check for the kernel messages that should be reported to Virtuozzo developers:

```
# grep -E "Call Trace|Code" /var/log/messages*
```

Then, you should find kernel-related lines in the corresponding log file and figure out what kernel was booted when the oops occurred. Search backward for the Linux string, look for strings like:

```
Sep 26 11:41:12 kernel: Linux version 2.6.18-8.1.1.el5.028stab043.1 \
(root@rhel5-32-build) (gcc version 4.1.1 20061011 (Red Hat 4.1.1-30)) \
#1 SMP Wed Aug 29 11:51:58 MSK 2007
```

An oops usually starts with some description of what happened and ends with the Code string. Here is an example:

```
Aug 25 08:27:46 boar BUG: unable to handle kernel NULL pointer dereference at \
virtual address 00000038
Aug 25 08:27:46 boar printing eip:
Aug 25 08:27:46 boar f0ce6507
Aug 25 08:27:46 boar *pde = 00003001
Aug 25 08:27:46 boar Oops: 0000 [#1]
Aug 25 08:27:46 boar SMP
Aug 25 08:27:46 boar last sysfs file:
Aug 25 08:27:46 boar Modules linked in: snapapi26(U) bridge(U) ip_vzredir(U) \
vzredir(U) vzcompat(U) vzrst(U) i
p_nat(U) vzcpt(U) ip_contrack(U) nfnetlink(U) vzlinkdev(U) vzethdev(U) vzevent(U) \
vzlist(U) vznet(U) vzmo
n(U) xt_tcpudp(U) ip_vznetstat(U) vznetstat(U) iptable_mangle(U) iptable_filter(U) \
ip_tables(U) vztable(U) vzquota(U) vzdev(U) autofs4(U) hidp(U) rfcomm(U) l2cap(U) \
bluetooth(U) sunrpc(U) ipv6(U) xt_length(U) ipt_ttl(U) xt_tcpmss(U) ipt_TCPMSS(U) \
xt_multiport(U) xt_limit(U) ipt_tos(U) ipt_REJECT(U) x_tables(U) video(U) sbs(U) \
i2c_ec(U) button(U) battery(U) asus_acpi(U) ac(U) lp(U) floppy(U) sg(U) pcspkr(U) \
i2c_piix4(U) e100(U) parport_pc(U) i2c_core(U) parport(U) cpqphp(U) eepr100(U) \
mii(U) serio_raw(U) ide_cd(U) cdrom(U) ahci(U) libata(U) dm_snapshot
(U) dm_zero(U) dm_mirror(U) dm_mod(U) megaraid(U) sym53c8xx(U) \
scsi_transport_spi(U) sd_mod(U) scsi_mod(U) ext3(U) jbd(U) ehci_hcd(U) ohci_hcd(U) \
uhci_hcd(U)
Aug 25 08:27:46 boar CPU: 1, VCPU: -1.1
Aug 25 08:27:46 boar EIP: 0060:[<f0ce6507>] Tainted: P VLI
Aug 25 08:27:46 boar EFLAGS: 00010246 (2.6.18-028stab043.1-ent #1)
Aug 25 08:27:46 boar EIP is at clone_endio+0x29/0xc6 [dm_mod]
Aug 25 08:27:46 boar eax: 00000010 ebx: 00000001 ecx: 00000000 edx: 00000000
Aug 25 08:27:46 boar esi: 00000000 edi: b6f52920 ebp: c1a8dbc0 esp: 0b483e38
Aug 25 08:27:46 boar ds: 007b es: 007b ss: 0068
Aug 25 08:27:46 boar Process swapper (pid: 0, veid: 0, ti=0b482000 task=05e3f2b0 \
task.ti=0b482000)
Aug 25 08:27:46 boar Stack: 0b52caa0 00000001 00000000 b6f52920 00000000f0ce64de \
```

```

00000000 02478825
Aug 25 08:27:46 boar 00000000 c18a8620 b6f52920 271e1a8c 024ca03800000000 00000000 \
00000000
Aug 25 08:27:46 boar 00000000 00000000 c18a3c00 00000202 c189e89400000006 00000000 \
05cb7200
Aug 25 08:27:46 boar Call Trace:
Aug 25 08:27:46 boar [<f0ce64de>] clone_endio+0x0/0xc6 [dm_mod]
Aug 25 08:27:46 boar [] bio_endio+0x50/0x55
Aug 25 08:27:46 boar [<024ca038>] __end_that_request_first+0x185/0x47c
Aug 25 08:27:46 boar [<f0c711eb>] scsi_end_request+0x1a/0xa9 [scsi_mod]
Aug 25 08:27:46 boar [<02458f04>] mempool_free+0x5f/0x63
Aug 25 08:27:46 boar
Aug 25 08:27:46 boar [<f0c713c3>] scsi_io_completion+0x149/0x2f3 [scsi_mod]
Aug 25 08:27:46 boar [<f0c333b9>] sd_rw_intr+0x1f1/0x21b [sd_mod]
Aug 25 08:27:46 boar [<f0c6d3b9>] scsi_finish_command+0x73/0x77 [scsi_mod]
Aug 25 08:27:46 boar [<024cbfa2>] blk_done_softirq+0x4d/0x58
Aug 25 08:27:46 boar [] __do_softirq+0x84/0x109
Aug 25 08:27:46 boar [<0242650d>] do_softirq+0x36/0x3a
Aug 25 08:27:46 boar [<024050b7>] do_IRQ+0xad/0xb6
Aug 25 08:27:46 boar [<024023fa>] default_idle+0x0/0x59
Aug 25 08:27:46 boar [<0240242b>] default_idle+0x31/0x59
Aug 25 08:27:46 boar [<024024b1>] cpu_idle+0x5e/0x74
Aug 25 08:27:46 boar =====
Aug 25 08:27:46 boar Code: 5d c3 55 57 89 c7 56 89 ce 53 bb 01 00 00 00 83 ec 0c \
8b 68 3c 83 7f 20 00 8b 45 00 8b 00 89 44 24 04 8b 45 04 89 04 24 8b 40 04 <8b> \
40 28 89 44 24 08 0f 85 86 00 00 00 f6 47 10 01 75 0a 85 c9
Aug 25 08:27:46 boar EIP: [<f0ce6507>] clone_endio+0x29/0xc6 [dm_mod] \
SS:ESP0068:0b483e38
Aug 25 08:27:46 boar Kernel panic - not syncing: Fatal exception in interrupt

```

You can save the oops in a file to be able to provide it when asking for technical support.

13.2.3 Finding a Kernel Function That Caused the D Process State

If there are too many processes in the D state and you can't find out what is happening, issue the following command:

```
# objdump -Dr /boot/vmlinux-'uname -r' >/tmp/kernel.dump
```

and then get the process list:

```
# ps axfwln
F UID PID PPID PRI NI VSZ RSS WCHAN STAT TTY TIME COMMAND
100 0 20418 20417 17 0 2588 684 - R ? 0:00 ps axfwln
100 0 1 0 8 0 1388 524 145186 S ? 0:00 init
040 0 8670 1 9 0 1448 960 145186 S ? 0:00 syslogd -m 0
040 0 8713 1 10 0 1616 1140 11ea02 S ? 0:00 crond
```

Look for a number under the WCHAN column for the process in question. Then, open /tmp/kernel.dump in an

editor, find that number in the first column and then scroll backward to the first function name, which can look like this:

```
"c011e910 <sys_nanosleep>:"
```

Then you can tell if the process “lives” or is blocked into the found function.

13.3 Container Management Issues

This section includes recommendations on how to solve certain container issues.

13.3.1 Failure to Start a Container

An attempt to start a container fails.

Solution 1

If there is a message on the system console: IP address is already used, issue the `cat /proc/vz/veinfo` command. The information about the container numeric identifier, container class, number of container’s processes and container IP address shall be displayed for each running container. This shall also demonstrate that your container is up, i.e. it must be running without any IP address assigned. Set its IP address using the command:

```
# prlctl set <CT_name> --ipadd <IP_address>
```

Where `<CT_name>` is the container name and `<IP_address>` is the desired IP address.

Solution 2

The container might be configured incorrectly. Try to validate the container configuration and find out what parameters have caused the error. Set appropriate values using the `prlctl set` command.

Solution 3

The container might have used all its disk quota (disk space). Check the quota (see *Managing Disk Quotas* on page 110) and adjust its parameters if needed.

Solution 4

Run the `prlctl console` utility to log in and get access to the container console. The utility will provide container startup/shutdown output that may be used to pinpoint the problem. For example:

```
# prlctl console MyCT
```

Where MyCT is the container name.

13.3.2 Failure to Access a Container from Network

Solution 1

The IP address assigned to the container might be already in use in your network. Make sure it is not. The problem container address can be checked by issuing the following command:

```
# grep IP_ADDRESS /etc/vz/conf/<UUID>.conf  
IP_ADDRESS="10.0.186.101"
```

The IP addresses of other containers, which are running, can be checked by running

```
# cat /proc/vz/veinfo
```

Solution 2

Make sure the routing to the container is properly configured. Containers can use the default router for your network, or you may configure the hardware node as router for its containers.

13.3.3 Failure to Log In to a Container

The container starts successfully, but you cannot log in.

Solution 1

You are trying to connect via SSH, but access is denied. Probably you have not set the password of the root user yet or there is no such user. In this case, use the `prlctl set --userpasswd` command. For example, for the container MyCT you might issue the following command:

```
# prlctl set MyCT --userpasswd root:secret
```

Solution 2

Check forwarding settings by issuing the following command:

```
# cat /proc/sys/net/ipv4/conf/venet0/forwarding
```

If it is 0 then change it to 1 by issuing the following command:

```
# echo 1 > /proc/sys/net/ipv4/conf/venet0/forwarding
```

13.4 Getting Technical Support

You can get technical support via the mailing list, bug tracker, and support forum. For more information, visit <https://virtuozzo.com/support/>.