Virtuozzo 6

Storage Administrator's Guide

February 28, 2017

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

Contents

CHAPTER 1

Introduction

This chapter provides basic information about the Virtuozzo Storage solution and this guide.

In This Chapter

About This Guide

The guide is intended for system administrators interested in deploying Virtuozzo Storage in their networks.

The document assumes that you have good knowledge of the Linux command-line interface and extensive experience working with local networks.

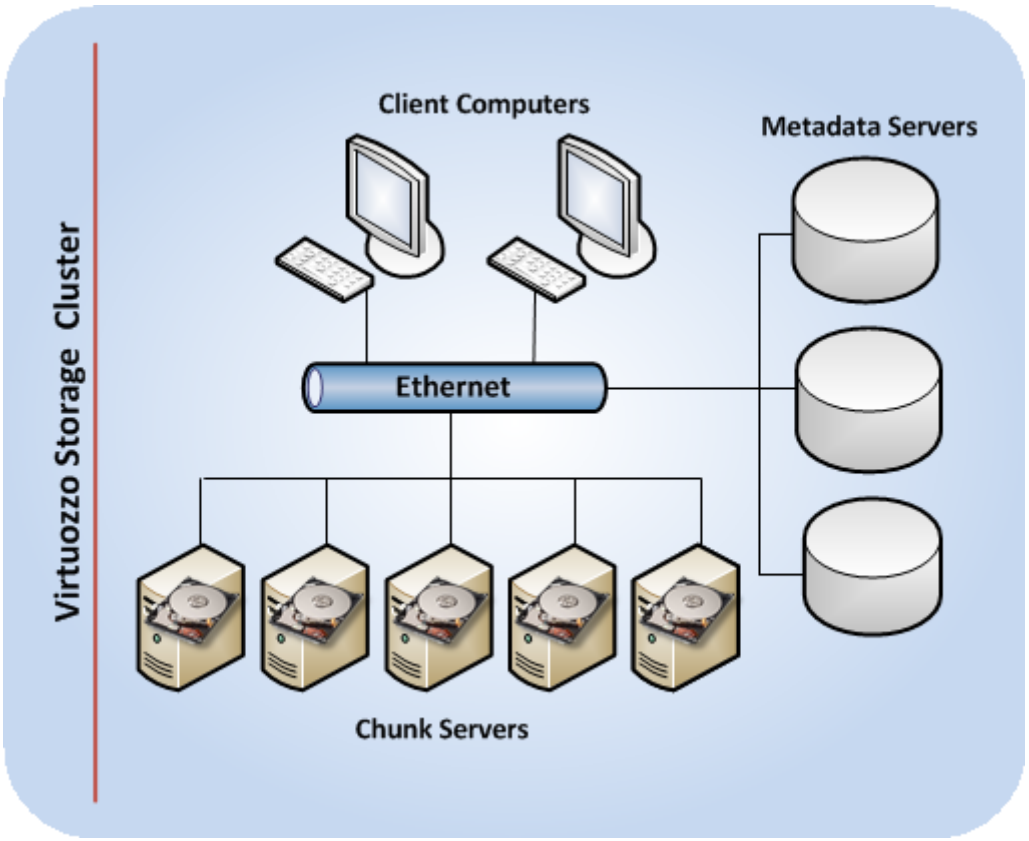
About Virtuozzo Storage

Virtuozzo Storage is a solution allowing you to quickly and easily transform low-cost commodity storage hardware and network equipment into a protected enterprise-level storage, like SAN (Storage Area Network) and NAS (Network Attached Storage).

Virtuozzo Storage is optimized for storing large amounts of data and provides replication, high-availability, and self-healing features for your data. Using Virtuozzo Storage, you can safely store and run virtual machines and Containers, migrate them with zero downtime across physical hosts, provide high availability for your Virtuozzo installations, and much more.

Virtuozzo Storage Architecture

Before starting the deployment process, you should have a clear idea of the Virtuozzo Storage infrastructure. A typical Virtuozzo Storage is shown below.



The basic component of Virtuozzo Storage is a cluster. The cluster is a group of physical computers connected to the same Ethernet network and performing the following roles:

- *chunk servers*
- *metadata servers*
- *client computers* (or *clients*)

Brief Overview

All data in a Virtuozzo Storage cluster, including virtual machine and Container disk images, is stored in the form of fixed-size chunks on chunk servers. The cluster automatically replicates the chunks and distributes them across the available chunk servers to provide high availability of data.

To keep track of data chunks and their replicas, the cluster stores metadata about them on metadata (MDS) servers. The central MDS server, called the *master MDS server*, monitors all cluster activity and keeps metadata current.

Clients manipulate data stored in the cluster by sending different types of file requests, such as modifying an existing file or creating a new one.

Chunk Servers

Chunk servers store all the data, including the contents of virtual machines and Containers, in the form of fixed-size chunks and provide access to these chunks. All data chunks are replicated and the replicas are kept on different chunk servers to achieve high availability. If one of the chunk servers goes down, the other chunk servers will continue providing the data chunks that were stored on the failed server.

Metadata Servers

Metadata (MDS) servers store metadata about chunk servers and control how files keeping the contents of virtual machines and Containers are split into chunks and where these chunks are located. MDS servers also ensure that a cluster has enough chunk replicas and store a global log of all important events that happen in the cluster.

To provide high availability for a Virtuozzo Storage cluster, you need to set up several MDS servers in the cluster. In this case, if one MDS server goes offline, another MDS server will continue keeping control over the cluster.

Note: MDS servers deal with processing metadata only and do not normally participate in any read/write operations related to data chunks.

Clients

Clients are computers with Virtuozzo 6 from where you run virtual machines and Containers stored in a Virtuozzo Storage cluster.

Notes:

1. You can set up any computer in the cluster to perform the role of a metadata server, chunk server, or client. You can also assign two or all three roles to one and the same computer. For example, you can configure a computer to act as a client by installing Virtuozzo 6 on it and running virtual machines and Containers from the computer. At the same time, if you want this computer to allocate its local disk space to the cluster, you can set it up as a chunk server.

2. Though Virtuozzo Storage can be mounted as a file system, it is not a POSIX-compliant file system and lacks some POSIX features like ACL, user and group credentials, hardlinks, and some other.

Virtuozzo Storage Configurations

This section provides information on two Virtuozzo Storage configurations:

- **Minimum configuration** (p. 10). You can create the minimum configuration for evaluating the Virtuozzo Storage functionality. This configuration, however, is not recommended for use in production environments.

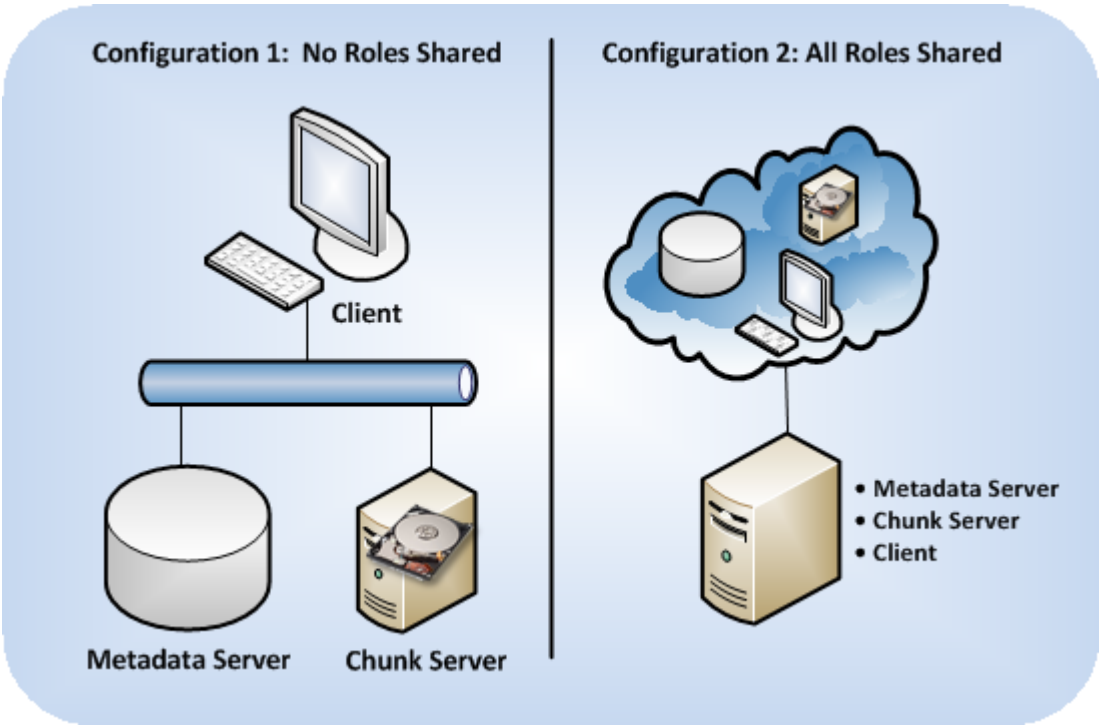
- **Recommended configuration** (p. 11). You can use the recommended Virtuozzo Storage configuration in a production environment "as is" or adapt it to your needs.

Minimum Configuration

The minimum hardware configuration for deploying a Virtuozzo Storage cluster is given below:

Server Role	Number of Servers
Metadata Server	1 (can be shared with chunk servers and clients)
Chunk Server	1 (can be shared with metadata servers and clients)
Client	1 (can be shared with chunk and metadata servers)
Total number of servers	with role sharing: **1** without role sharing: **3**

Graphically, the minimum configuration can be represented as follows:



For a Virtuozzo Storage cluster to function, it must have at least one MDS server, one chunk server, and one client. The minimum configuration has two main limitations:

1. The cluster has one metadata server, which presents a single point of failure. If the metadata server fails, the entire Virtuozzo Storage cluster will become non-operational.
2. The cluster has one chunk server that can store only one chunk replica. If the chunk server fails, the cluster will suspend all operations with chunks until a new chunk server is added to the cluster.

Recommended Configuration

The table below lists two of the recommended configurations for deploying Virtuozzo Storage clusters.

Metadata Server	Chunk Server	Total Number of Servers
3 (can be shared with chunk servers and clients)	**5-9** (can be shared with metadata servers and clients)	**5 or more** (depending on the number of clients and chunk servers and whether they share roles)
5 (can be shared with chunk servers and clients)	**10 or more** (can be shared with metadata servers and clients)	**5 or more** (depending on the number of clients and chunk servers and whether they share roles)
Clients	**1 or more** You can include any number of clients in the cluster. For example, if you have 5 servers with Virtuozzo, you can configure them all to act as clients. You can share servers acting as clients with chunk and metadata servers. For example, you can have 5 physical servers and configure each of them to simultaneously act as an MDS server, a chunk server, and a client.	

Even though new clusters are configured to have 1 replica for each data chunk by default, you need to configure each data chunk to have at least 3 replicas to provide high availability for your data.

In total, at least 9 machines running Virtuozzo Storage are recommended per cluster. Smaller clusters will work as fine but will not provide the significant performance advantages over direct-attached storage (DAS) or improved recovery times.

Notes:

1. For large clusters, it is critically important to configure proper failure domains to improve data availability. For more information, see **Configuring Failure Domains** (p. 37).

2. In small and medium clusters, MDS servers consume little resources and do not require being set up on dedicated Hardware Nodes.

3. A small cluster is 3 to 5 machines, a medium cluster is 6 to 15-20 machines, and a large cluster is 15-20 machines and more.

4. Time should be synchronized on all servers in the cluster via NTP. Doing so will make it easier for the support department to understand cluster logs (migrations, failovers, etc.). For more details, see http://kb.virtuozzo.com/en/3197.

System Requirements

Before setting up a Virtuozzo Storage cluster, make sure you have all the necessary equipment at hand. You are also recommended to:

- Consult **Using SSD Drives** (p. 93) to learn how you can increase cluster performance by using solid-state drives for write journaling and data caching, and how many SSDs you may need depending on the number of HDDs in your cluster.
- Check **Troubleshooting** (p. 102) for common hardware issues and misconfigurations that may affect your cluster performance and lead to data inconsistency and corruption.

General

- Each service (be it MDS, CS or client) requires 1.5 GB of free space on root partition for logs. For example, to run 1 metadata server, 1 client, and 12 chunk servers on a host, you will need 21 GB of free space on the root partition.

Metadata Servers

- Virtuozzo 6 or newer,
- 1 CPU core,
- 1 GB of free RAM per 100 TB of data in the cluster,
- 3 GB of free disk space per 100 TB of data in the cluster,
- 1 or more Ethernet adapters (1 Gbit/s or faster).

Note: It is recommended to place the MDS journal on SSD, either dedicated or shared with CS and client caches, or at least on a dedicated HDD which has no CS services on it.

Chunk Servers

- Virtuozzo 6 or newer,
- 1/8 of a CPU core (e.g., 1 CPU core per 8 CS),
- 256 MB of free RAM,
- 100 GB or more of free disk space,
- 1 or more Ethernet adapters (1 Gbit/s or faster).

Notes:

1. On using local RAID with Virtuozzo Storage, consult **Exploring Possible Disk Drive Configurations (p. 89).**

2. Using a shared JBOD array across multiple nodes running CS services may introduce a single point of failure and make the cluster unavailable if all data replicas happen to be allocated and stored on the failed JBOD. For more information, see **Configuring Failure Domains** (p. 37).

3. For large clusters (see **Recommended Configuration** (p. 11)), it is critically important to configure proper failure domains to improve data availability. For more information, see **Configuring Failure Domains** (p. 37).

4. Do not place chunk servers on disks already used in other I/O workloads, e.g., system or swap. Sharing disks between CS and other sources of I/O will result in severe performance loss and high I/O latencies.

Clients

- Virtuozzo 6 or newer,
- 1 CPU core per 30,000 IOPS,
- 1 GB of RAM,
- 1 or more Ethernet adapters (1 Gbit/s or faster).

Note: For hard disk requirements and the recommended partitioning scheme for servers that run Virtuozzo and participate in clusters, see the **Hardware Compatibility** section in the *Virtuozzo 6 Installation Guide*.

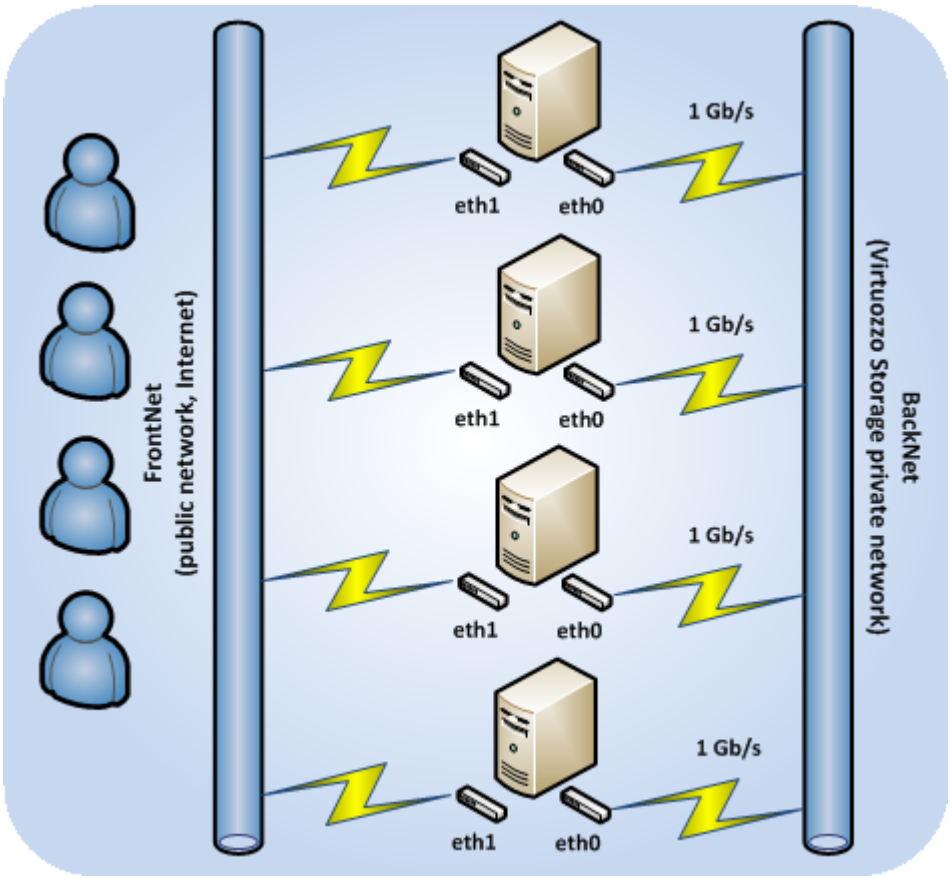
Network Requirements

When planning your network, make sure that it

- operates at 1 Gbit/s or faster (for more details, see **Using 1 GbE and 10 GbE Networks** (p. 91)),
- has non-blocking Ethernet switches.

You should use separate networks and Ethernet adapters for user and cluster traffic. This will prevent possible I/O performance degradation in your cluster by external traffic. Besides, if a cluster is accessible from public networks (e.g., from the Internet), it may become a target of Denial-of-Service attacks, and the entire cluster I/O subsystem may get stuck.

The figure below shows a sample network configuration for Virtuozzo Storage.



In this network configuration:

- *BackNet* is a private network used solely for interconnection and intercommunication of servers in the cluster and is not available from the public network. All servers in the cluster are connected to this network via one of their network cards.
- *FrontNet* is a public network customers use to access their virtual machines and Containers in the Virtuozzo Storage cluster.

Notes:

1. Network switches are a very common point of failure, so it is critically important to configure proper failure domains to improve data availability. For more information, see **Configuring Failure Domains** (p. 37).

2. To learn more about Virtuozzo Storage networks (in particular, how to bind chunk servers to specific IP addresses), see **Securing Server Communication in Clusters** (p. 82).

CHAPTER 2

Setting Up a Virtuozzo Storage Cluster

This chapter provides information on setting up a Virtuozzo Storage cluster. It starts with an overview of the Virtuozzo Storage setup and then describes each setup step in detail.

In This Chapter

Setup Overview

Setting up a Virtuozzo Storage cluster includes these steps:

1. Configuring Virtuozzo Storage cluster discovery.

   In this step, you define the way of detecting a cluster name and resolving the detected name into the IP addresses of MDS servers. For more on this step, see **Configuring Virtuozzo Storage Discovery** (p. 16).

2. Checking data flushing.

   In this step, you check that all storage devices (hard disk drives, solid disk drives, RAIDs, etc.) you plan to include in your cluster can successfully flush data to disk when the server power goes off unexpectedly.

3. If required, prepare the additional (second, third, etc.) disks as described in **Preparing Disks for Virtuozzo Storage** (p. 20).

4. Setting up metadata servers.

   In this step, you create and configure metadata servers for storing metadata about chunk servers and data chunks. For more on this step, see **Setting Up the First Metadata Server** (p. 22).

5. Setting up chunk servers.

In this step, you create and configure chunk servers for storing the actual content of virtual machines and Containers in data chunks. For more on this step, see **Setting Up Chunk Servers** (p. 24).

6 Setting up clients.

In this step, you create and configure clients from where you will access the Virtuozzo Storage cluster and run virtual machines and Containers. For more on this step, see **Setting Up Clients** (p. 26).

Note: You can also set up a Virtuozzo Storage using the Virtuozzo 6 installer. It automatically configures your system as a metadata server, chunk server, or client. For detailed information on this, see the *Virtuozzo 6 Installation Guide*.

Configuring Cluster Discovery

Virtuozzo Storage discovery is the process of:

1 Detecting all available cluster names on the network. Each Virtuozzo Storage cluster is identified by a unique name. All cluster tools use this name when performing specific operations on the cluster or monitoring its health and status.

2 Resolving the detected cluster names into the network addresses of MDS servers. MDS servers are the central components of any cluster, so all cluster tools must be able to discover their IP addresses.

To set up cluster discovery in your network, you can use one of the following techniques:

- (Recommended) DNS records (see **Using DNS records** (p. 16)),
- Zeroconf (see **Setting Up Zeroconf** (p. 18)).

You can also manually specify the information about metadata servers when setting up or configuring the cluster (see **Specifying MDS Servers Manually** (p. 18)).

Note: To verify that the Hardware Node can discover the cluster, use the `pstorage discover` command.

Using DNS Records

The recommended way of configuring cluster discovery is to use special DNS records. The process of setting up this type of cluster discovery includes two steps:

1 Announcing the information about running MDS servers in the cluster so that chunk servers, clients, and new MDS servers can automatically obtain this information when necessary. You do this using DNS SRV records.

2 Defining DNS TXT records or enabling DNS zone transfers so that you can discover the unique names of available clusters when necessary.

Announcing the Information About MDS Servers

You can use SRV records to announce the information about running MDS servers in the cluster. The service field of an SRV record pointing to an MDS server must have the following format:

```
_pstorage._tcp.CLUSTER_NAME
```

where

- `_sstorage` is the symbolic name reserved for Virtuozzo Storage.
- `_tcp` denotes that Virtuozzo Storage uses the TCP protocol for communication in the cluster.
- *`CLUSTER_NAME`* is the name of the Virtuozzo Storage cluster described by the record.

The following example shows a DNS zone file that contains records for three MDS servers listening on the default port 2510 and configured for the `stor1` cluster:

```
$ORIGIN stor.test.
$TTL    1H
@       IN SOA ns rname.invalid. (1995032001 5H 10M 1D 3H)
        NS @
        A       192.168.100.1
s1      A       192.168.100.1
s2      A       192.168.100.2
s3      A       192.168.100.3
; SERVICE SECTION
; MDS for the 'stor1' cluster runs on s1.stor.test and listens on port 2510
_pstorage._tcp.stor1 SRV 0 1 2510     s1
; MDS for the 'stor1' cluster runs on s2.stor.test and listens on port 2510
_pstorage._tcp.stor1 SRV 0 1 2510     s2
; MDS for the 'stor1' cluster runs on s3.stor.test and listens on port 2510
_pstorage._tcp.stor1 SRV 0 1 2510     s3
; eof
```

Once you configure DNS SRV records for the `stor1` cluster, you can list them by issuing the following SRV query:

```
# host -t SRV _pstorage._tcp.stor1
_pstorage._tcp.stor1.stor.test has SRV record 0 1 2510 s1.stor.test.
_pstorage._tcp.stor1.stor.test has SRV record 0 1 2510 s2.stor.test.
_pstorage._tcp.stor1.stor.test has SRV record 0 1 2510 s3.stor.test.
```

Discovering Cluster Names

The easiest and most efficient way of discovering the names of clusters in your network is to specify all cluster names in `pstorage_clusters` TXT records of DNS zone files. The following example provides a sample of valid TXT record formats for Virtuozzo Storage clusters:

```
pstorage_clusters   300    IN     TXT    "cluster1,cluster2" "cluster3,cluster4"
pstorage_clusters   300    IN     TXT    "cluster5"
pstorage_clusters   300    IN     TXT    "cluster6" "cluster7"
```

Another way of discovering cluster names in your network is to use DNS zone transfers. Once DNS zone transfers are enabled, cluster tools will be able to retrieve all DNS SRV records from DNS zone files and extract cluster names from these records.

After you set up cluster discovery via DNS TXT records or DNS zone transfers, you can run the `pstorage discover` command on any of the cluster servers to discover the names of all clusters in your network:

```
# pstorage discover
02-10-12 13:16:46.233 Discovering using DNS TXT records: OK
02-10-12 13:16:46.308 Discovering using DNS zone transfer: FAIL
stor1
stor2
stor3
```

The example `pstorage` output shows that:

- Clusters names are discovered via the DNS TXT records.
- Three clusters with the names of `stor1`, `stor2`, and `stor3` are currently set up on the network.

Setting Up Zeroconf

Warning: Zeroconf discovery does not work if services are running in Containers or virtual machines.

Zeroconf is another method of discovering cluster names and resolving the discovered names into the IP addresses of running MDS servers. This method does not require any special configuration efforts on your part, except ensuring that multicasts are supported and enabled on your network.

Note: To verify that the Hardware Node can discover the cluster, use the `pstorage discover` command.

Specifying MDS Servers Manually

If you cannot configure the DNS records in your network, you need to manually specify the IP addresses of all running MDS servers in the cluster each time you do any of the following:

- Set up a new MDS server in a cluster (except for the first MDS server). For details, see **Adding MDS Servers** (p. 30).
- Set up a new chunk server to a cluster. For details, see **Setting Up Chunk Servers** (p. 24).
- Set up a new client for a cluster. For details, see **Setting Up Clients** (p. 26).

To specify the IP address of an MDS server manually, create the `bs.list` file in the `/etc/pstorage/clusters/`*`Cluster_Name`* directory (make this directory if it does not exist) on the server you are configuring for the cluster and specify in it the IP address and port to use for connecting to the MDS server. For example:

```
# echo "10.30.100.101:2510" >> /etc/pstorage/clusters/stor1/bs.list
# echo "10.30.100.102:2510" >> /etc/pstorage/clusters/stor1/bs.list
```

This command:

1. Assumes that you are configuring discovery for the `stor1` cluster (thus, the directory name of `/etc/pstorage/clusters/stor1`).
2. Creates the `/etc/pstorage/clusters/stor1/bs.list` file on the server, if it did not exist before.
3. Adds the information on two MDS servers with IP addresses 10.30.100.101 and 10.30.100.102 to the `bs.list` file.

Checking Data Flushing

Before creating the cluster, you are recommended to check that all storage devices (hard disk drives, solid disk drives, RAIDs, etc.) you plan to include in your cluster can successfully flush data to disk when the server power goes off unexpectedly. Doing so will help you detect possible problems with devices that may lose data stored in their cache in the event of a power failure.

Virtuozzo Storage ships a special tool, `pstorage-hwflush-check`, for checking how a storage device flushes data to disk in an emergency case such as power outage. The tool is implemented as a client/server utility:

- **Client**. The client continuously writes blocks of data to the storage device. When a data block is written, the client increases a special counter and sends it to the server that keeps it.
- **Server**. The server keeps track of the incoming counters from the client so that it always knows the counter number the client will send next. If the server receives the counter that is less than the one already stored on the server (e.g., because the power was turned off and the storage device did not flush the cached data to disk), the server reports an error.

To check that a storage device can successfully flush data to disk when the power fails, follow the procedure below:

On the server part:

1. On some computer running Virtuozzo 6, install the `pstorage-hwflush-check` tool. This tool is part of the `pstorage-ctl` package and can be installed with this command:

```
# yum install pstorage-ctl
```

2. Run the `pstorage-hwflush-check` server:

```
# pstorage-hwflush-check -l
```

On the client part:

1. On the computer hosting a storage device you want to check, install the `pstorage-hwflush-check` tool:

```
# yum install pstorage-ctl
```

2. Run the `pstorage-hwflush-check` client, for example:

```
# pstorage-hwflush-check -s pstorage1.example.com -d /pstorage/stor1-ssd/test -t 50
```

   where

- `-s pstorage1.example.com` is the hostname of the computer where the `pstorage-hwflush-check` server is running.
- `-d /pstorage/stor1-ssd/test` defines the directory to use for testing data flushing. During its execution, the client creates a file in this directory and writes data blocks to it.
- `-t 50` sets the number of threads for the client to write data to disk. Each thread has its own file and counter. You can increase the number of threads (max. 200) to test your system in more stressful conditions.

You can also specify other options when running the client. For more information on available options, see the `pstorage-hwflush-check` man page.

3 Wait for 10-15 seconds or more and power off the computer where the client is running, and then turn it on again.

> **Note:** The **Reset** button does not turn off the power so you need to press the **Power** button or pull out the power cord to switch off the computer.

4 Restart the client by executing the same command you used to run it for the first time:

```
# pstorage-hwflush-check -s pstorage1.example.com -d /pstorage/stor1-ssd/test -t 50
```

Once launched, the client reads all written data, determines the version of data on the disk, and then restarts the test from the last valid counter. It then sends this valid counter to the server, and the server compares it with the latest counter it has. You may see output like: `id <N>: <counter_on_disk> -> <counter_on_server>` which means one of the following:

- If the counter on disk is lower than the counter on server, it means that the storage device has failed to flush the data to disk. Avoid using this storage device in production—especially for CS or journals—as you risk losing data.
- If the counter on disk is higher than the counter on server, it means that the storage device has flushed the data to disk but the client has failed to report it to the server. The network may be too slow or the storage device may be too fast for the set number of load threads so you may consider increasing it. This storage device can be used in production.
- If both counters are equal, it means the storage device has flushed the data to disk and the client has reported it to the server. This storage device can be used in production.

To be on the safe side, repeat the procedure several times. Once you check your first storage device, continue with all remaining devices you plan to use in the cluster. You need to test all devices you plan to use in the cluster: SSD disks used for client caching and CS journaling, disks used for MDS journals, disks used for chunk servers.

Preparing Disks for Virtuozzo Storage

Each chunk server is a service that handles a single physical disk in the cluster. Although the disk should be used solely by the CS service, technically, you can use it for multiple purposes. E.g., create a small partition for the operating system and leave the rest of disk space for Virtuozzo Storage. If the disk is already partitioned, skip this section and proceed to creating a chunk server. Otherwise follow the instructions in this section to prepare the disk for use in Virtuozzo Storage.

New disks attached to and recognized by the Hardware Node need to be prepared for use in the Virtuozzo Storage cluster by means of the `/usr/libexec/pstorage/prepare_pstorage_drive` tool. The tool does the following:

1. Removes existing partitions from the disk.
2. Creates and formats the required partition(s).

After that, manually add the new partition to `/etc/fstab` and mount it.

Notes:

1. If you do not need the disk to be bootable, run the tool with the `--noboot` option to skip GRUB bootloader installation.

2. For SSD drives, use the `--ssd` option.

3. To have the tool proceed without confirmation prompts, use the `-y` option.

Preparing Disks for Use as Chunk Servers

1. To prepare an HDD or SSD for use as a chunk server, run the tool with the drive name as the option. For example:

```
# /usr/libexec/pstorage/prepare_pstorage_drive /dev/sdb
ALL data on /dev/sdb will be completely destroyed. Are you sure to continue? [y]
y
Zeroing out beginning and end of /dev/sdb...
Partitioning /dev/sdb...
Waiting for kernel...
Formatting /dev/sdb1 partition...
Done!
```

2. Find out partition UUID:

```
#  ls -al /dev/disk/by-uuid/ | grep sdb1
lrwxrwxrwx 1 root root  10 Jun 19 02:41 f3fbcbb8-4224-4a6a-89ed-3c55bbc073e0 ->
../../sdb1
```

3. Add the new partition to `/etc/fstab` by UUID.

   - For `vzkernel` version 2.6.32-042stab108.8 or newer, the `lazytime` mount option is recommended. For example:

```
UUID=f3fbcbb8-4224-4a6a-89ed-3c55bbc073e0 /pstorage/stor1-cs1    ext4
defaults,lazytime   1 2
```

   - For older `vzkernel` versions, use the `defaults` mount option. For example:

```
UUID=f3fbcbb8-4224-4a6a-89ed-3c55bbc073e0 /pstorage/stor1-cs1    ext4    defaults
1 2
```

4. Mount the partition to `/pstorage/<cluster>-cs<N>`, where `<cluster>` is cluster name and `<N>` is the first unused CS index number.

Note: If `/pstorage/<cluster>-cs<N>` does not exist, create it.

Preparing SSDs for Write Journaling or Caching

1 To prepare an SSD for write journaling or caching, run the tool with two options: `--ssd` and drive name. For example:

```
# /usr/libexec/pstorage/prepare_pstorage_drive /dev/sdb --ssd
ALL data on /dev/sdb will be completely destroyed. Are you sure to continue? [y]
y
Zeroing out beginning and end of /dev/sdb...
Partitioning /dev/sdb...
Waiting for kernel...
Formatting /dev/sdb1 partition...
Done!
```

2 Find out partition UUID:

```
#  ls -al /dev/disk/by-uuid/ | grep sdb1
lrwxrwxrwx 1 root root  10 Jun 19 02:41 f3fbcbb8-4224-4a6a-89ed-3c55bbc073e0 ->
../../sdb1
```

3 Add the new partition to `/etc/fstab` by UUID.

- For `vzkernel` version 2.6.32-042stab108.8 or newer, the `lazytime` mount option is recommended. For example:

```
UUID=f3fbcbb8-4224-4a6a-89ed-3c55bbc073e0 /pstorage/stor1-ssd1   ext4
defaults,lazytime   1 2
```

- For older `vzkernel` versions, use the `defaults` mount option. For example:

```
UUID=f3fbcbb8-4224-4a6a-89ed-3c55bbc073e0 /pstorage/stor1-ssd1   ext4    defaults
1 2
```

4 Mount the partition to `/pstorage/<cluster>-ssd<N>`, where `<cluster>` is cluster name and `<N>` is the first unused SSD index number.

Note: If `/pstorage/<cluster>-ssd<N>` does not exist, create it.

Setting Up the First Metadata Server

Setting up the first MDS server is the first step in creating a Virtuozzo Storage cluster. You can add more MDS servers later to provide better availability for your cluster, as described in **Configuring MDS Servers** (p. 29).

The process of setting up the first MDS server (called *master MDS server*) and, thus, creating a cluster includes two stages:

1 Preparing to create the MDS server (p. 22).

2 Creating the MDS server (p. 23).

Stage 1: Preparing to Create the First MDS Server

To prepare for making the first MDS server, do the following:

1 Choose a name for the cluster that will uniquely identify it among other clusters in your network.

A name may contain the characters a-z, A-Z, 0-9, dash (-), and underscore (_). The examples used throughout this guide assume that the cluster name is `stor1`.

> **Note:** When choosing a name for the cluster, make sure it is unique on your network. Also, do not use names that were once assigned to other clusters in your network, even if these clusters do not exist any more. This will help you avoid possible problems with services from previous cluster setups that might still be running and trying to operate on the new cluster. Though such operations will not succeed, they can make your work as a cluster administrator more difficult.

2. Log in to the computer you want to configure as a metadata server as root or as a user with root privileges.
3. Download and install the following RPM packages on the computer: `pstorage-ctl`, `pstorage-libs-shared`, and `pstorage-metadata-server`.

   The packages are available in the Virtuozzo remote repository (this repository is automatically configured for your system when you install Virtuozzo) and can be installed with this command:

```
# yum install pstorage-metadata-server
```

4. Make sure that cluster discovery is configured in your network. For details, see **Configuring Cluster Discovery** (p. 16).

After you complete the steps above, you are ready to create the MDS server.

Stage 2: Creating the First MDS Server

To create the first MDS server, you use the `pstorage make-mds` command, for example:

```
# pstorage -c stor1 make-mds -I -a 10.30.100.101 -r /pstorage/stor1-mds -p
```

This command does the following:

1. Asks you for a password to use for password-based authentication in your cluster. Password-based authentication enhances security, requiring each server to be authenticated before it can be included in the cluster. The password you specify is encrypted and saved into the `/etc/pstorage/clusters/stor1/auth_digest.key` file on the MDS server.
2. Creates a Virtuozzo Storage cluster with the name of `stor1` (the `-I` option tells `pstorage` to create a new cluster).
3. Creates a metadata server and configures the IP address of 10.30.100.101 for communication with this server. By default, Virtuozzo Storage uses ports 2510 and 2511 to communicate with MDS servers. If necessary, you can replace the default ports with your own ones by reserving two unoccupied consecutive ports and specifying the first one after the IP address of your MDS server (e.g., `-a 10.30.100.101:4110` if your custom ports are 4110 and 4111).

   Replace 10.30.100.101 in the example above with the IP address of your own MDS server. The specified IP address must be (1) static (or in the case of using DHCP, mapped to the MAC address of the MDS server) and (2) chosen from the range of IP addresses on the BackNet network dedicated to your Virtuozzo Storage cluster. See **Network Requirements** (p. 13) for details.

4 Creates a journal in the `/pstorage/stor1-mds` directory on the MDS server and adds the information about the `stor1` cluster to it. When choosing a directory for storing the journal, make sure that the partition where the directory is located has at least 10 GB of free disk space.

After you have created the MDS server, start the MDS management service (`pstorage-mdsd`) and configure it to start automatically when the server boots:

```
# service pstorage-mdsd start
# chkconfig pstorage-mdsd on
```

For information on including additional MDS servers in a Virtuozzo Storage cluster, see **Configuring MDS Servers** (p. 29).

Setting Up Chunk Servers

A chunk server stores actual data of virtual machines and Containers and services requests to it. All data is split into chunks and can be stored in a Virtuozzo Storage cluster in multiple copies called *replicas*.

Initially, any cluster is configured to have only one replica per each data chunk, which is sufficient to evaluate the Virtuozzo Storage functionality using one server only. In production, however, to provide high availability for your data, you need to configure the cluster for each data chunk to have at least three replicas. This requires at least three chunk servers to be set up in the cluster. You can modify the default replication parameter using the `pstorage` utility. For details, see **Configuring Replication Parameters** (p. 36).

Notes:

1. Using shared JBOD arrays across multiple nodes running CS services may introduce a single point of failure and make the cluster unavailable if all data replicas happen to be allocated and stored on the failed JBOD. For more information, see **Configuring Failure Domains** (p. 37).

2. Do not place chunk servers on disks already used in other I/O workloads, e.g., system or swap. Sharing disks between CS and other sources of I/O will result in severe performance loss and high I/O latencies.

The process of setting up a chunk server includes two stages:

1 Preparing to create a chunk server (p. 25).

2 Creating the chunk server (p. 25).

Note: You can configure hot plugging so that a chunk server is created automatically when you attach an HDD to the server. For more information, see the `pstorage-hotplugd` man page.

Stage 1: Preparing to Create a Chunk Server

To prepare for creating a chunk server, do the following:

1. Log in to the computer you want to act as a chunk server as root or as a user with root privileges.
2. Download and install the following RPM packages: `pstorage-ctl`, `pstorage-libs-shared`, and `pstorage-chunk-server`.

   These packages are available in the Virtuozzo remote repository (this repository is automatically configured for your system when you install Virtuozzo) and can be installed with this command:

```
# yum install pstorage-chunk-server
```

3. Make sure that cluster discovery is configured for the server. For details, see **Configuring Cluster Discovery** (p. 16).
4. Authenticate the server in the cluster. This step is required only if the server where you are setting up the chunk server has never been authenticated in the cluster before.

   For example, you can skip this step if this is the same server where you set up the first MDS server. Otherwise, run the following command to authenticate the server in the cluster:

```
# pstorage -c stor1 auth-node
Please enter password for cluster:
```

   During its execution, the command asks you for the password to validate the server. Type the password you specified when setting up the first MDS server and press Enter. `pstorage` then compares the provided password with the one stored on the MDS server, and if the passwords match, successfully authenticates the server.
5. If the disk the CS will be created on has not been prepared for Virtuozzo Storage, do so as described in **Preparing Disks for Virtuozzo Storage** (p. 20).
6. Mount the prepared disk.

Stage 2: Creating a Chunk Server

Note: For large clusters (see **Recommended Configuration** (p. 11)), it is critically important to configure proper failure domains to improve data availability. For more information, see **Configuring Failure Domains** (p. 37).

To create the chunk server, use the `pstorage make-cs` command, for example:

```
# pstorage -c stor1 make-cs -r /pstorage/stor1-cs
```

This command:

1. Creates the `/pstorage/stor1-cs` directory if it does not exist and configures it for storing data chunks.
2. Configures your disk as a chunk server and joins it to the `stor1` cluster.
3. Assigns the chunk server to the default storage tier. Storage tiers allow you to keep different kinds of data on different chunk servers. For details, see **Configuring Storage Tiers** (p. 40).

After you have created the chunk server, start the chunk management service (`pstorage-csd`) and configure it to start automatically when the chunk server boots:

```
# service pstorage-csd start
# chkconfig pstorage-csd on
```

Once you set up the first chunk server, proceed with creating other chunk servers.

Creating Host UUIDs for Chunk Servers

Virtuozzo Storage distinguishes hosts the CS services run on by unique host UUIDs generated during installation. If you plan to set up new hosts by deploying a golden image with an OS and preinstalled Virtuozzo Storage packages, you will need to generate new host UUIDs instead of the one inherited from the golden image.

To create a CS on a copy of the host, do the following:

1. Make sure the golden image does not contain any metadata servers, chunk servers, or clients.
2. Deploy the golden image on a clean host.
3. Generate a new UUID for the host to replace the one inherited from the golden image:
   ```
   # /usr/bin/uuidgen -r | tr '-' ' ' | awk '{print $1$2$3}' > /etc/pstorage/host_id
   ```
   Note: For more information on the `uuidgen` utility, see its man page.
4. Create a CS on the host.

Setting Up Clients

The process of setting up a client includes three stages:

1. Preparing to mount the Virtuozzo Storage cluster to the client (p. 26).
2. Mounting the cluster (p. 27).
3. Configuring virtual machines and Containers to be stored in the cluster (p. 27).

Stage 1: Preparing to Mount the Cluster

To prepare for mounting the Virtuozzo Storage cluster to the client:

1. Log in to the server you want to act as a client as root or as a user with root privileges.
2. Download and install the `pstorage-libs-shared` and `pstorage-client` RPM packages.

   These packages are available in the Virtuozzo remote repository (this repository is automatically configured for your system when you install Virtuozzo) and can be installed with this command:
   ```
   # yum install pstorage-client
   ```
3. Create the directory to mount the Virtuozzo Storage cluster to, for example:

```
# mkdir -p /pstorage/stor1
```

4 Make sure that cluster discovery is configured in your network. For details, see **Configuring Cluster Discovery** (p. 16).

5 Authenticate the server in the cluster. This step is required only if the server where you are setting up the client has never been authenticated in the cluster before.

For example, you can skip this step if this is the same server where you set up the first MDS server or some of the chunk servers. Otherwise, run the following command to authenticate the server in the cluster:

```
# pstorage -c stor1 auth-node
Please enter password for cluster:
```

During its execution, the command asks you for the password to validate the server. Type the password you specified when setting up the first MDS server and press Enter. `pstorage` then compares the provided password with the one stored on the MDS server, and if the passwords match, successfully authenticates the server.

Stage 2: Mounting the Cluster

Next, you need to mount the cluster to make it accessible to the client. You can do this with the the `pstorage-mount` command. For example, if your Virtuozzo Storage cluster has the name of `stor1`, you can run this command to mount it to the `/pstorage/stor1` directory on the client:

```
# pstorage-mount -c stor1 /pstorage/stor1
```

You can also configure the client to automatically mount the cluster to the `/pstorage/stor1` directory when the client boots. To do this, add a line like the following to the `/etc/fstab` file:

```
pstorage://stor1 /pstorage/stor1 fuse.pstorage rw,nosuid,nodev 0 0
```

Note: If the cluster is not used for virtualization, you can mount it with the `--fail-on-nospace` option. In this case an `ERR_NO_SPACE` error will be returned if the cluster runs out of free space.

Stage 3: Configuring Virtual Machines and Containers

To configure a server with Virtuozzo to store its virtual machines and Containers in the cluster, do the following:

1 Log in to the server as `root`.

2 Configure Containers for use in the cluster:

a Check the path to the Container private area in the `/etc/vz/vz.conf` file. By default, the path is set to the following:

```
VE_PRIVATE=/vz/private/$VEID
```

b Make a symbolic link from the Container private area to the directory in the Virtuozzo Storage cluster that will store Containers. Assuming that this directory is `/pstorage/stor1/private`, create this directory and run the following command:

```
# ln -s /pstorage/stor1/private/ /vz/private
```

> **Note:** If the `/vz/private` directory already exists on your server, remove it before running the `ln -s` command.

3 Configure virtual machines for use in the cluster:

a Check the default location of virtual machine files:

```
# prlsrvctl info | grep "VM home"
VM home: /vz/vmprivate
```

b Make a symbolic link from the default location to the directory in the Virtuozzo Storage cluster that will store virtual machines. For example, to make a link to the `/pstorage/stor1/vmprivate` directory, create this directory and execute the following command:

```
# ln -s /pstorage/stor1/vmprivate/ /vz/vmprivate
```

> **Note:** If the `/vz/vmprivate` directory already exists on your server, remove it before running the `ln -s` command.

CHAPTER 3

Configuring Virtuozzo Storage Clusters

This chapter describes the ways to configure a Virtuozzo Storage cluster once you create it.

In This Chapter

Configuring MDS Servers

For a Virtuozzo Storage cluster to function, the majority of MDS servers must be up and running in the cluster. So to ensure high availability of a cluster, you need to set up at least three MDS servers for it. This will allow you to survive the loss of one MDS server. By configuring five MDS servers for a cluster, you can ensure that your cluster will continue operating even if two MDS servers go offline.

Notes:

1. When adding and removing MDS servers, make sure that the running MDS servers in the cluster always have a majority.

2. Remove non-functioning MDS servers from the cluster as soon as possible (e.g., right after you replace a broken server with a new one) to ensure that all MDS servers are up and running and the majority is not lost if one more MDS server fails. Let us assume that 3 MDS servers are running in your cluster. 1 MDS server fails so you add a new MDS server to the cluster. Now the total number of MDS servers in the cluster is 4, with one server offline. If one more MDS server fails, the cluster will have only 2 running MDS servers and become unavailable because the majority (3 running MDS servers) cannot be achieved any more.

This section explains how to

- add new MDS servers to a cluster (p. 30)
- remove existing MDS servers from a cluster (p. 31)

Adding MDS Servers

The procedure of setting up the first MDS server is described in **Setting Up the First Metadata Server** (p. 22). To configure a second and all subsequent MDS servers for a cluster, follow the steps below:

1. Make sure that you remember the exact name of the Virtuozzo Storage cluster where you want to add an MDS server. The example below uses `stor1` as the cluster name.
2. Log in to the computer you want to configure as an MDS server and add to the cluster as root or as a user with root privileges.
3. Download and install the following RPM packages on the computer: `pstorage-ctl`, `pstorage-libs-shared`, and `pstorage-metadata-server`.

   These packages can be installed with this command:

```
# yum install pstorage-metadata-server
```

4. Make sure that cluster discovery is configured for the server. For details, see **Configuring Cluster Discovery** (p. 16).
5. Authenticate the server in the cluster.

   This step is required only if the physical server where you are setting up the MDS server has never been authenticated in the cluster before. For example, you can skip this step if you have already configured the server as a chunk server or a client. Otherwise, run the following command to authenticate the server in the cluster:

```
# pstorage -c stor1 auth-node
Please enter password for cluster:
```

   During its execution, the command asks you for the password to validate the server. Enter the password you specified when setting up the first MDS server and press Enter. `pstorage` then compares the provided password with the one stored on the MDS server, and if the passwords match, successfully authenticates the server.
6. Create the MDS server and add it to the cluster:

```
# pstorage -c stor1 make-mds -a 10.30.100.102 -r /pstorage/stor1-mds
```

   In the command above:

   - `stor1` is the name of the cluster you are adding the MDS server to.
   - `10.30.100.102` is the IP address of the new MDS server.

     Replace 10.30.100.102 in the example above with the IP address of your own MDS server. The specified IP address must be (1) static (or in the case of using DHCP, mapped to the MAC address of the MDS server) and (2) chosen from the range of IP addresses on the BackNet network dedicated to your Virtuozzo Storage cluster. See **Network Requirements** (p. 13) for details.
   - `/pstorage/stor1-mds` is the path to a journal that will store the information about the `stor1` cluster. When choosing a directory for storing the journal, make sure that the partition where the directory is located has at least 10 GB of free disk space.

If the DNS records or Zeroconf discovery is not configured in your network, you need to additionally use the `-b` option and specify the IP addresses of the first MDS server (and all other MDS servers, if you have more than one in your cluster) when running the command:

```
# pstorage -c stor1 make-mds -a 10.30.100.102:2510 -r /pstorage/stor1-mds -b
10.30.100.101
```

7 Start the MDS management service (`pstorage-mdsd`) and configure it to start automatically on the MDS server boot:

```
# service pstorage-mdsd start
# chkconfig pstorage-mdsd on
```

For instructions on how to check that the MDS server has been successfully configured for your cluster, see **Monitoring Virtuozzo Storage Clusters** (p. 66).

Removing MDS Servers

Sometimes, you may need to remove an MDS server from a Virtuozzo Storage cluster, for example, to upgrade the server or to perform some maintenance tasks on it. To do this:

1 Configure a new MDS server to replace the one you plan to remove from the cluster. For instructions, see **Adding MDS Servers** (p. 30).

2 Find out the index number of the MDS server to remove by running the following command on some of your MDS servers or clients:

```
# pstorage -c stor1 top
```

This will display detailed information about the cluster. Locate the section with the information about MDS servers, for example:

```
...
MDSID STATUS   %CTIME   COMMITS   %CPU    MEM   UPTIME HOST
M    1 avail      0.0%       0/s   0.0%    64m  17d  6h 10.30.17.38
     2 avail      0.0%       0/s   0.0%    50m  12d  3h 10.30.45.12
     3 avail      0.0%       0/s   0.0%    57m   7d  1h 10.30.10.15
...
```

The index number is displayed in the **MDSID** column. In the output above, three MDS servers are configured for the `stor1` cluster. They have index numbers of 1, 2, and 3.

3 Remove the MDS server from the cluster. For example, to remove the MDS server with index number 3, run this command:

```
# pstorage -c stor1 rm-mds 3
```

Configuring Chunk Servers

This section explains how to complete the following tasks:

- Increase disk space in a cluster by adding new chunk servers to it (p. 32).
- Remove chunk servers from a cluster for performing maintenance tasks on them (p. 32).

Note: You can configure hot plugging so that a chunk server is created automatically when you attach an HDD to the server. For more information, see the `pstorage-hotplugd` man page.

Adding New Chunk Servers to Increase Disk Space

You can increase the amount of disk space in a Virtuozzo Storage cluster simply by adding new chunk servers to it. For details, see **Setting Up Chunk Servers** (p. 24).

Note: Virtuozzo Storage can scale to support at least 8 PB of effective available disk space, which means up to 24 PB of physical disk space in the case of mirroring with 3 copies.

Removing Chunk Servers

If you need to remove a chunk server from the cluster, for example, to perform some maintenance tasks on it, do the following:

1 Make sure that

- The number of chunk servers configured for your cluster is enough to store the required number of data chunks (that is, equals or exceeds the current replication value).
- The chunk servers have enough disk space to store the chunks.

For instructions on obtaining this information, see **Monitoring Chunk Servers** (p. 70). If you need to add a new chunk server to the cluster before removing an active one, see **Setting Up Chunk Servers** (p. 24).

2 Find out the index number of the chunk server you want to remove by running the following command on some of your cluster servers:

```
# pstorage -c stor1 top
```

This will display detailed information about the `stor1` cluster. Locate the section with the information about chunk servers, for example:

```
...
 CSID  STATUS     SPACE   FREE REPLICAS IOWAIT IOLAT(ms) QDEPTH HOST
 1025  active     105GB   88GB       40     0%       0/0    0.0 10.30.17.38
 1026  active     105GB   88GB       40     0%       0/0    0.0 10.30.18.40
 1027  active     105GB   99GB       40     0%       0/0    0.0 10.30.21.30
 1028  active     107GB  101GB       40     0%       0/0    0.0 10.30.16.38
...
```

The **CSID** column displays the index number of chunk servers. In the output above, four chunk servers are configured for the `stor1` cluster. They have index numbers of 1025, 1026, 1027, and 1028.

3 Remove the chunk server from the cluster. For example, to delete the chunk server with index number 1028 from the `stor1` cluster, run this command:

```
# pstorage -c stor1 rm-cs --wait 1028
```

Once you initiate the delete operation, the cluster starts replicating data chunks that were stored on the removed server and placing them on the remaining chunk servers in the cluster. The `--wait` option, when specified, tells the command to wait until the operation is complete (which may take a long time).

Notes:

1. If the CS's disk has disappeared from the OS and the CS has no access to its repository, data chunks which need to be replicated will not be accessible and CS removal will not complete. You will need to forcibly remove the CS from the cluster with the `-f` option. **Warning:** Do so only if the CS is irreversibly lost. Never remove active chunk servers with the `-f` option.

2. When deletion is in progress, you can cancel it with the command `pstorage -c stor1 rm-cs --cancel 1028`. This might be useful, for example, if you specified a wrong ID of the chunk server to remove.

To add a removed chunk server back to the cluster, set it up from scratch by following the steps in **Setting Up Chunk Servers (p. 24).**

Configuring HDD Hot Plugging

Maintaining a large cluster (see **Recommended Configuration** (p. 11)) with many hosts and multiple HDDs per host requires a simple automated solution to easily add new and replace failed HDDs.

Linux has had support for handling such tasks for a long time. In Linux, each time an HDD is inserted or removed, the kernel generates an event which is passed to the `udev` device manager that allows the system to process the event further.

`pstorage-hotplugd` is a set of scripts which use `udev` and enable you to automatically create and start a CS on an HDD hot plug event.

Warning: Special considerations apply to sharing SSD write cache between running CSes. This functionality is not implemented yet, so chunk servers with SSD journals will not be started automatically on disk hotplug.

To enable and configure hotplugging, set the parameters in the configuration file as described in **Setting Hotplugging Parameters** (p. 34).

Disk Hot Plug Sequence

When you connect an HDD while hotplug is enabled, the following happens:

1. The new HDD is detected. If it does not have a partition table or has an empty partition table, a single partition spanning the entire HDD is created, formatted to ext4, and labelled **pstorage-hotplug**. Otherwise the HDD is ignored.
2. The partition labelled **pstorage-hotplug** is detected, mounted to a unique location, and checked for an existing CS on it.

   Note: Such a partition can be created either automatically (see the first step above) or manually as described in **Partitioning the Disk Manually** (p. 34).

3 If a CS is found, it is added to the list of CSes on this host and started. Otherwise, a new CS is created for the specified cluster and started.

Notes:

1. You cannot manage metadata servers with the `pstorage-hotplugd` scripts.

2. Hot plugging currently works for HDDs connected to any HBA (recommended) or JBOD controller. Such HDDs must work in the passthrough mode.

3. As hot unplugging is not supported, you can unplug a disk as follows: 1) remove the CS as described in **Removing Chunk Servers** (p. 32), 2) unmount the disk, 3) unplug the disk.

Setting Hotplugging Parameters

You can configure how hotplug events are handled by setting the following parameters in `/etc/pstorage/hotplug.config`:

Parameter	Description
ENABLE_HOTPLUG	Mandatory. Enables or disables hotplug support. To enable, set to `yes`.
HOTPLUG_CLUSTER	Mandatory. Name of the cluster for which the new chunk servers will be configured.
HOTPLUG_TIER	Sets the tier for new chunk servers.
HOTPLUG_LOG_FILE	Hotplug event log.
HOTPLUG_MOUNT_PATH	Hotplug mount path.

Note: Restart the `pstorage-hotplugd` daemon each time you change the `ENABLE_HOTPLUG` variable in the configuration file.

Partitioning the Disk Manually

To partition a disk manually, do the following (in our example, `/dev/sdb` is the target disk):

1 Create a new partition table:

```
# parted /dev/sdb mklabel gpt
```

2 Create a new partition:

```
# parted /dev/sdb mkpart primary 0 -0
```

3 Format the new partition to ext4 with the required label **pstorage-hotplug**:

```
# mkfs.ext4 -L pstorage-hotplug /dev/sdb1
```

4 Configure a mount point for this partition in `/etc/fstab`:

```
# /dev/sdb1 /mnt ext4 defaults 0 0
```

5 Mount the partition:

```
# mount /dev/sdb1 /mnt
```

6 Create a CS at `/mnt/cs`, or move an existing CS, or leave it to automatic CS creation on next replug. Example:

```
# pstorage -c stor1 make-cs -r /mnt/cs
```

Configuring Clients

This section explains how to complete the following tasks:

- Add new clients to clusters (p. 35).
- Update clients (p. 35).
- Remove clients from clusters (p. 35).

Adding Clients

The process of including additional clients in a Virtuozzo Storage cluster does not differ from that of setting up the first client and is described in **Setting Up Clients** (p. 26) in detail.

Once you configure the client, you can run different commands on it to administer the cluster. For example, you can monitor the cluster health and status using the `pstorage top` command. For more details on monitoring Virtuozzo Storage clusters, see **Monitoring Virtuozzo Storage Clusters** (p. 66).

Updating Clients

The process of updating software on clients that participate in clusters does not differ from that of updating software on standalone servers, except for updating the `pstorage-client` package. When updating this package, pay attention to the following:

- If no cluster is mounted to the client, the client starts using the updated package right away.
- If at least one cluster is mounted to the client, the updated package is installed, but the client starts using it only after you remount the cluster or reboot the client.

Removing Clients

Removing a client from a Virtuozzo Storage cluster simply means unmounting the directory under which the cluster is mounted on the client. Assuming that the cluster is mounted under `/pstorage/stor1`, you can unmount it as follows:

1. Make sure that all virtual machines and Containers in the cluster are stopped.
2. Unmount the cluster:

```
# umount /pstorage/stor1
```

3. If your cluster is configured to be automatically mounted when the client boots, comment out the cluster entry in the `/etc/fstab` file on the client:

```
# pstorage://stor1 /pstorage/stor1 fuse.pstorage rw,nosuid,nodev 0 0
```

Configuring High Availability

High Availability keeps virtual machines, Containers, and iSCSI targets operational even if the Hardware Node they are hosted on fails. In such cases, the affected virtual environments will either be relocated to healthy Hardware Nodes in the cluster in the round-robin manner (default) or to a special backup Hardware Node.

For information on how to configure High Availability for entire Hardware Nodes or specific virtual machines and Containers, consult *Virtuozzo 6 User's Guide*.

Managing Cluster Parameters

This section explains what cluster parameters are and how you can configure them with the `pstorage` utility.

Cluster Parameters Overview

The cluster parameters control creating, locating, and managing replicas for data chunks in a Virtuozzo Storage cluster. All parameters can be divided into two main groups:

- *replication parameters*
- *location parameters*

The table below briefly describes each of the cluster parameters. For more information on the parameters and how to configure them, see the following sections.

Parameter	Description
Replication Parameters	
`Normal Replicas`	The number of replicas to create for a data chunk, from 1 to 15. Recommended: 3.
`Minimum Replicas`	The minimum number of replicas for a data chunk, from 1 to 15. Recommended: 2.
Location Parameters	
`Failure Domain`	A placement policy for replicas, can be `room`, `row`, `rack`, `host` (default), or `disk` (`CS`).
`Tier`	Storage tiers, from 0 to 3 (0 by default).

Configuring Replication Parameters

The cluster replication parameters define

- *The normal number of replicas of a data chunk*. When a new data chunk is created, Virtuozzo Storage automatically replicates it until the normal number of replicas is reached.

- *The minimum number of replicas of a data chunk (optional)*. During the life cycle of a data chunk, the number of its replicas may vary. If a lot of chunk servers go down it may fall below the defined minimum. In such a case, all write operations to the affected replicas are suspended until their number reaches the minimum value.

To check the current replication parameters applied to a cluster, you can use the `pstorage get-attr` command. For example, if your cluster is mounted to the `/pstorage/stor1` directory, you can run the following command:

```
# pstorage get-attr /pstorage/stor1
connected to MDS#1
File: '/pstorage/stor1'
Attributes:
  ...
  replicas=1:1
  ...
```

As you can see, the normal and minimum numbers of chunk replicas are set to 1.

Initially, any cluster is configured to have only 1 replica per each data chunk, which is sufficient to evaluate the Virtuozzo Storage functionality using one server only. In production, however, to provide high availability for your data, you are recommended to

- configure each data chunk to have at least 3 replicas,
- set the minimum number of replicas to 2.

Such a configuration requires at least 3 chunk servers to be set up in the cluster.

To configure the current replication parameters so that they apply to all virtual machines and Containers in your cluster, you can run the `pstorage set-attr` command on the directory to which the cluster is mounted. For example, to set the recommended replication values to the `stor1` cluster mounted to `/pstorage/stor1`, set the normal number of replicas for the cluster to 3:

```
# pstorage set-attr -R /pstorage/stor1 replicas=3
```

The minimum number of replicas will be automatically set to 2 by default.

Note: For information on how the minimum number of replicas is calculated, see the `pstorage-set-attr` man page.

Along with applying replication parameters to the entire contents of your cluster, you can also configure them for specific directories and files. For example:

```
# pstorage set-attr -R /pstorage/stor1/private/101 replicas=3
```

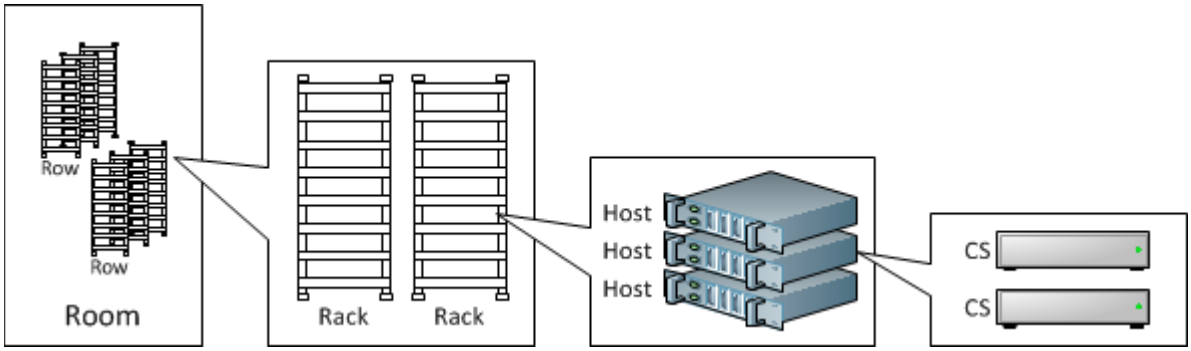
Configuring Failure Domains

A failure domain is a set of services which can fail in a correlated manner. Due to correlated failures it is very critical to scatter data replicas across different failure domains for data availability. Failure domain examples include:

- A single disk (the smallest possible failure domain). For this reason, Virtuozzo Storage never places more than 1 data replica per disk or chunk server (CS).
- A single host running multiple CS services. When such a host fails (e.g., due to a power outage or network disconnect), all CS services on it become unavailable at once. For this reason, Virtuozzo Storage is configured by default to make sure that a single host never stores more than 1 chunk replica (see **Defining Failure Domains** below).
- Larger-scale multi-rack cluster setups introduce additional points of failure like per-rack switches or per-rack power units. In this case, it is important to configure Virtuozzo Storage to store data replicas across such failure domains to prevent data unavailability on massive correlated failures of a single domain.

Failure Domain Topology

Every Virtuozzo Storage service component has topology information assigned to it. Topology paths define a logical tree of components' physical locations consisting of 5 identifiers: `room.row.rack.host_ID.cs_ID`.



The first 3 topology path components, `room.row.rack`, can be configured in the `/etc/pstorage/location` configuration files (for more information, see the man page for `pstorage-config-files`). The last 2 components, `host_ID.cs_ID`, are generated automatically:

- `host_ID` is a unique, randomly generated host identifier created during installation and located at `/etc/pstorage/host_id`.
- `cs_ID` is a unique service identifier generated at CS creation.

Note: To view the current services topology and disk space available per location, run the `pstorage top` command and press **w**.

Defining Failure Domains

Based on the levels of hierarchy described above, you can use the `pstorage set-attr` command to define failure domains for proper file replica allocation:

```
# pstorage -c <cluster_name> set-attr -R -p /failure-domain=<disk|host|rack|row|room>
```

where

- `disk` means that only 1 replica is allowed per disk or chunk server,
- `host` means that only 1 replica is allowed per host (default),
- `rack` means that only 1 replica is allowed per rack,
- `row` means that only 1 replica is allowed per row,
- `room` means that only 1 replica is allowed per room.

You should use the same configuration for all cluster files as it simplifies the analysis and is less error-prone.

As an example, do the following to configure a 5-rack setup:

1. Assign `0.0.1` to `/etc/pstorage/location` on all hosts from the first rack, `0.0.2` on all hosts from second rack, and so on.
2. Create 5 metadata servers: 1 on any host from the first rack, 1 on any host from second rack, and so on.
3. Configure Virtuozzo Storage to have only 1 replica per rack (assuming that the cluster name is `stor1`):

```
# pstorage -c stor1 set-attr -R -p /failure-domain=rack
```

4. Create CS services with the `pstorage-make-cs` command as described in **Setting Up Chunk Servers** (p. 24).

Changing Host Location

Once a host has started running services in the cluster, its topology is cached in the MDS and cannot be changed. All new services created on the host will use that cached information. If you need to modify the host location information, do the following:

1. Kill and remove CS and client services running on the host with the `pstorage-rm-cs` and `umount` commands as described in **Removing Chunk Servers** (p. 32) and **Removing Clients** (p. 35), respectively.
2. Set `/etc/pstorage/host_id` to another unique ID, e.g., generated with `/dev/urandom`.
3. Adjust `/etc/pstorage/location` as required.
4. Recreate the CS and client services: mount the cluster and create new CS instances using the `pstorage-make-cs` command as described in **Setting Up Chunk Servers** (p. 24) and **Setting Up Clients** (p. 26), respectively.

Recommendations on Failure Domains

- **Important:** Do not use failure domain `disk` simultaneously with journaling SSDs. In this case, multiple replicas may happen to be located on disks serviced by the same journaling SSD. If that SSD fails, all replicas that depend on journals located on it will be lost. As a result, your data may be lost.

- For the flexibility of Virtuozzo Storage allocator and rebalancing mechanisms, it is always recommended to have at least 5 failure domains configured in a production setup (hosts, racks, etc.). Reserve enough disk space on each failure domain so if a domain fails it can be recovered to healthy ones.
- When MDS services are created, the topology and failure domains must be taken into account manually. That is, in multi-rack setups, metadata servers should be created in different racks (5 MDSes in total).
- At least 3 replicas are recommended for multi-rack setups.
- Huge failure domains are more sensitive to total disk space imbalance. For example, if a domain has 5 racks, with 10 TB, 20 TB, 30 TB, 100 TB, and 100 TB total disk space, it will not be possible to allocate (10+20+30+100+100)/3 = 86 TB of data in 3 replicas. Instead, only 60 TB will be allocatable, as the low-capacity racks will be exhausted sooner, and no 3 domains will be available for data allocation, while the largest racks (the 100TB ones) will still have free space
- If a huge domain fails and goes offline, Virtuozzo Storage will not perform data recovery by default, because replicating a huge amount of data may take longer than domain repairs. This behavior managed by the global parameter `mds.wd.max_offline_cs_hosts` (configured with `pstorage-config`) which controls the number of failed hosts to be considered as a normal disaster worth recovering in the automatic mode
- Failure domains should be similar in terms of I/O performance to avoid imbalance. For example, avoid setups in which `failure-domain` is set to `rack`, all racks but one have 10 Nodes each and one rack has only 1 Node. Virtuozzo Storage will have to repeatedly save a replica to this single Node, reducing overall performance
- Depending on the global parameter `mds.alloc.strict_failure_domain` (configured with `pstorage-config`), the domain policy can be strict (default) or advisory. Tuning this parameter is highly not recommended unless you are absolutely sure of what you are doing.

Using Storage Tiers

This section describes storage tiers used in Virtuozzo Storage clusters and provides information of how to configure and monitor them.

What Are Storage Tiers

Storage tiers represent a way to organize storage space. You can use them to keep different categories of data on different chunk servers. For example, you can use high-speed solid-state drives to store performance-critical data instead of caching cluster operations.

Configuring Storage Tiers

To assign disk space to a storage tier, do this:

1. Assign all chunk servers with SSD drives to tier 1. You can do this when setting up a chunk server; see **Creating the Chunk Server** (p. 25) for details.

Note: For information on recommended SSD drives, see **Using SSD Drives** (p. 93).

2 Assign the frequently accessed directories and files to tier 1 with the `pstorage set-attr` command. For example:

```
# pstorage set-attr -R /pstorage/stor1/private/MyCT tier=1
```

This command assigns the directory `/pstorage/stor1/private/MyCT` and its contents to tier 1.

When assigning storage to tiers, have in mind that faster storage drives should be assigned to higher tiers. For example, you can use tier 0 for backups and other cold data (CS without SSD journals), tier 1 for virtual environments—a lot of cold data but fast random writes (CS with SSD journals), tier 2 for hot data (CS on SSD), journals, caches, specific virtual machine disks, and such.

This recommendation is related to how Virtuozzo Storage works with storage space. If a storage tier runs out of free space, Virtuozzo Storage will attempt to temporarily use a lower tier. If you add more storage to the original tier later, the data, temporarily stored elsewhere, will be moved to the tier where it should have been stored originally.

For example, if you try to write data to the tier 2 and it is full, Virtuozzo Storage will attempt to write that data to tier 1, then to tier 0. If you add more storage to tier 2 later, the aforementioned data, now stored on the tier 1 or 0, will be moved back to the tier 2 where it was meant to be stored originally.

Automatic Data Balancing

To maximize the I/O performance of chunk servers in a cluster, Virtuozzo Storage automatically balances CS load by moving hot data chunks from hot chunk servers to colder ones.

A chunk server is considered hot if its request queue depth exceeds the cluster-average value by 40% or more (see example below). With data chunks, "hot" means "most requested".

The hotness (i.e. request queue depth) of chunk servers is indicated by the `QDEPTH` parameter shown in the output of `pstorage top` and `pstorage stat` commands. For example:

```
...
IO QDEPTH: 0.1 aver, 1.0 max; 1 out of 1 hot CS balanced     46 sec ago
...
  CSID STATUS      SPACE  AVAIL REPLICAS   UNIQUE IOWAIT IOLAT(ms) QDEPTH HOST
BUILD_VERSION
  1025 active     1007.3 156.8G     7142        0    10%     1/117    0.3 10.31.240.167
6.0.11-10
  1026 active     1007.3 156.8G     7267        0    11%     0/225    0.1 10.31.240.167
6.0.11-10
  1027 active     1007.3 156.8G     7151        0     2%      0/10    0.1 10.31.240.167
6.0.11-10
  1028 active     1007.3 156.8G     7285        0    13%     1/141    1.0 10.31.240.167
6.0.11-10
...
```

In the output, the `IO QDEPTH` line shows the average and maximum request queue depth values in the entire cluster for the last 60 seconds. The `QDEPTH` column shows average request queue depth values for each CS for the last 5 seconds.

Each 60 seconds, the hottest data chunk is moved from a hot CS to one with a shorter request queue.

Monitoring Storage Tiers

You can monitor disk space assigned to each storage tier with the `top` utility in the verbose mode (enabled by pressing `v`). Typical output may look like this:

```
Cluster 'tiers': healthy
Space: [OK] allocatable 3.3TB of 3.5TB, 3.5TB total, 3.5TB free
 tier 0: allocatable 1.6TB of 1.7TB, 1.7TB total, 1.7TB free
 tier 1: allocatable 866GB of 912GB, 912GB total, 912GB free
 tier 3: allocatable 866GB of 912GB, 912GB total, 912GB free
MDS nodes: 1 of 1, epoch uptime:  3 min
CS nodes:  4 of 4 (4 avail, 0 inactive, 0 offline)
Replication:  1 norm,  1 limit
Chunks: [OK] 0 healthy,  0 standby,  0 degraded,  0 urgent,
             0 blocked,  0 pending,  0 offline,  0 replicating,
             0 overcommitted,  0 deleting,  0 void
FS:  0B in 1 files, 0 inodes,  0 file maps,  0 chunks,  0 chunk
IO:       read     0B/s (  0ops/s), write     0B/s (  0ops/s)
IO total: read       0B (    0ops), write       0B (    0ops)
Repl IO:  read     0B/s, write:     0B/s
Sync rate:   0ops/s, datasync rate:   0ops/s
```

Changing Virtuozzo Storage Cluster Network

Before moving your cluster to a new network, consider the following:

- Changing the cluster network results in a brief downtime for the period when more than half of the MDS servers are unavailable.
- It is highly recommended to back up all MDS repositories before changing the cluster network.

To change the Virtuozzo Storage cluster network, do the following on each node in the cluster where an MDS service is running:

1. Stop the MDS service:

```
# service pstorage-mdsd stop
```

2. Specify new IP addresses for all metadata servers in the cluster with the command `pstorage configure-mds -r <MDS_repo> -n <MDS_ID@new_IP_address>[:port] [-n ...]`, where:
   - *<MDS_repo>* is the repository path of the MDS on the current node.
   - Each *<MDS_ID@new_IP_address>* pair is an MDS identifier and a corresponding new IP address.

For example, for a cluster with 5 metadata servers:

```
# pstorage -c stor1 configure-mds -r /pstorage/stor1-cs1/mds/data -n 1@10.10.20.1 -n
2@10.10.20.2 -n 3@10.10.20.3 -n 4@10.10.20.4 -n 5@10.10.20.5
```

Notes:

1. You can obtain the identifier and repository path for the current MDS with the `pstorage list-services -M` command.

2. If you omit the port, the default port `2510` will be used.

3 Start the MDS service:

```
# service pstorage-mdsd start
```

Managing Virtuozzo Storage Licenses

This section describes the process of managing Virtuozzo Storage licenses. You will learn to do the following:

- Install a new license for a Virtuozzo Storage cluster (p. 43).
- Update the installed license (p. 44).
- View the installed license contents (p. 44).
- Check the current license status (p. 45).

Obtaining the Hardware Node ID

The Hardware Node ID (HWID) is required to purchase a Virtuozzo Storage license. You can obtain the HWID with the `pstorage stat --license-hwid` command. For example:

```
# pstorage -c stor1 stat --license-hwid
...
3F96.DFF2.EAF6.CE86.DD49.786C.DC01.3D53
```

Note: A Virtuozzo Storage Hardware Node ID is not the same as a Virtuozzo Hardware Node ID. For details on how to obtain a Virtuozzo HWID, see the *Virtuozzo User's Guide*.

Installing the License

Along with installing Virtuozzo licenses on all clients in a cluster, you need to install a separate license to start using the Virtuozzo Storage functionality. One license is required per cluster. You can install the license from any server participating in the cluster: an MDS server, a chunk server, or a client.

To install the license, use the `pstorage load-license` command:

```
# pstorage -c stor1 load-license -p XXXXXX-XXXXXX-XXXXXX-XXXXXX-XXXXXX
```

If you have obtained the license in the form of a file, you can install it by using the `-f` option instead of `-p` and specifying the full path to the license file:

```
# pstorage -c stor1 load-license -f /etc/storlicense
```

Updating the License

In Virtuozzo, you can use the `pstorage update-license` command to update the license currently installed on your server. When executed, the utility connects to the Key Authentication (KA) server, retrieves a new license, downloads it to the server, and installs it there.

To update a license, do the following:

1. Make sure that the server where you want to update the license is connected to the Internet.
2. Run the `pstorage update-license` command to update the license.

For example, to update a license installed in the `pcs1` cluster, execute this command:

```
# pstorage -c pcs1 update-license
```

By default, `pstorage` obtains a new license from the default KA server. However, you can explicitly specify what KA server to use by passing the `--server` option to the command:

```
# pstorage -c pcs1 update-license --server ka.server.com
```

Once you run the command, `pstorage` will connect to the KA server with the hostname of `ka.server.com`, download a new license, and install it on your server.

Viewing the License Contents

You can use the `pstorage view-license` command to view the information on the license currently installed in your cluster. When executed, the utility processes the license and shows its contents on the screen. A sample output of `pstorage view-license` is given below:

```
# pstorage -c stor1 view-license
HWID: XXXX.XXXX.XXXX.XXXX.XXXX.XXXX.XXXX.XXXX
PCSSTOR
        status="ACTIVE"
        version=1.0
        expiration="08/24/2012 19:59:59"
        graceperiod=3600
        key_number="PCSS.XXXXXXXX.XXXX"
        platform="Linux"
        product="PCSS"
        gracecapacity=5
        autorecovery=0
        autorebalance=0
        snapshots=1
        capacity=500
        replicas=5
```

The main license parameters are explained in the table below.

Name	Description
HWID	Cluster ID.
status	License status. For details, see **Checking the License Status** (p. 45).
version	Version of Virtuozzo Storage for which the license was issued.
expiration	License expiration date and time.
graceperiod	Period, in seconds, during which Virtuozzo Storage continues functioning after the license has expired.
key_number	Key number under which the license is registered on the Key Authentication server.
platform	Operating system with which the license is compatible.
product	Product for which the license has been issued.
gracecapacity	Amount of disk space that data chunks may occupy in the cluster, in per cent to the **capacity** limit value. For example, if the capacity limit is set to 1 TB, and the grace capacity is 5%, data chunks may use 50 GB above the capacity limit.
capacity	Total amount of disk space, in GB, data chunks may occupy in the cluster. To view the disk space currently used by chunks, run the `pstorage top` command, press the V key on your keyboard, and check the **FS** field. For details, see **Understanding Disk Space Usage** (p. 71).
replicas	Maximum number of replicas a data chunk may have.
autorecovery	Denotes whether the auto-recovery feature is enabled (1) or disabled (0).
autorebalance	Denotes whether the auto-rebalance feature is enabled (1) or disabled (0).
snapshots	Denotes whether the snapshots feature is enabled (1) or disabled (0).

Checking the License Status

You can check the status of your license in one of the following ways:

- Using the `pstorage view-license`, for example:

```
# pstorage -c stor1 view-license | grep status
status="ACTIVE"
```

- Using the `pstorage stat` or `pstorage top` command, for example:

```
# pstorage -c stor1 stat | grep License
connected to MDS#1
License: PCSS.XXXXXXXX.XXXX is ACTIVE
```

The table below lists all statuses a license can have.

Status	Description
ACTIVE	License is valid and active.
VALID	License is valid and can be installed in the cluster.
EXPIRED	License has expired.

GRACED	License is currently on the grace period or data chunks in the cluster use disk space from the grace capacity.
INVALID	License is invalid (for example, because its start date is in the future).

Shutting Down Virtuozzo Storage Clusters

To shut down a Virtuozzo Storage cluster completely, do the following:

1 Stop all clients in the cluster. To do this, on each client:

a Shut down all running Containers and virtual machines.

b Unmount the cluster file system using the `umount` command. For example, if the cluster is mounted to the `/pstorage/stor1` directory on a client, you can unmount it as follows:

```
# umount /pstorage/stor1
```

c Disable the automatic mounting of the cluster by removing the cluster entry from the `/etc/fstab` file.

2 Stop the `shaman` service and disable its automatic start:

```
# service shamand stop
# chkconfig shamand off
```

3 Stop all MDS servers and disable their automatic start:

```
# service pstorage-mdsd stop
# chkconfig pstorage-mdsd off
```

4 Stop all chunk servers and disable their automatic start:

```
# service pstorage-csd stop
# chkconfig pstorage-csd off
```

CHAPTER 4

Exporting Virtuozzo Storage Cluster Data

Usually, you access virtual machines and Containers natively from clients running Virtuozzo. To do this, you use standard command-line utilities like `prlctl`. Another way to remotely access data located in Virtuozzo Storage clusters is to export it as

- NFS (running on ploops), or
- images over iSCSI.

These methods are described further in this section.

In This Chapter

Accessing Virtuozzo Storage Clusters via NFS

To access a Virtuozzo Storage Cluster via NFS, you need to:

1. Create and mount a ploop with the ext4 file system.
2. Set up an NFS share using either the standard `exportfs` command or the `/etc/exports` file.
3. Access the NFS share on the remote computer.

The following sections describe these steps in detail.

Preparing the Ploop

Loopback block devices (ploops) allow you to attach any Virtuozzo Storage file as a block device and format it to a conventional file system like ext4. Since Virtuozzo Storage is not optimized for small files and does not use a POSIX-compliant file system, you can use ploops with ext4 when you need the aforementioned functionality.

To prepare the ploop, do the following:

1. Load the required modules:

```
# modprobe ploop pfmt_ploop1 pio_kaio
```

2 Create the ploop:

```
# mkdir /pstorage/ploop0
# ploop init -s 1g -t ext4 /pstorage/ploop0/img0
```

This command creates a 1 GB ploop with the ext4 file system.

3 Mount the ploop:

```
# mkdir /mnt/ploop0
# ploop mount -m /mnt/ploop0 /pstorage/ploop0/DiskDescriptor.xml
```

4 (Optional) Set up a permanent ploop mount using /etc/fstab:

```
# cat >> /etc/fstab <<EOF
/pstorage/ploop0/DiskDescriptor.xml        /mnt/ploop0     ploop           defaults 0 0
EOF
```

Setting Up the NFS Share

Note: For the purpose of this example, let us assume that:

1. The source computer IP address is 192.168.0.1 and the destination computer IP address is 192.168.0.2.

2. The `exportfs` command is used to set up the NFS share

3. On the destination computer, the NFS share is mounted to the `/nfsshare` directory.

To share the created file system via NFS, do the following:

1 On the source computer, make sure the `nfsd` service is running.

2 On the source computer, run the `exportfs` command as follows:

```
# exportfs 192.168.0.2:/mnt/ploop0
```

3 On the remote computer, mount the shared path as follows:

```
# mount 192.168.0.1:/mnt/ploop0 /nfsshare
```

Now you can access the contents of the shared ext4 file system on the remote computer.

Note: More details on setting up NFS shares are provided in the *Red Hat Enterprise Linux Storage Administration Guide*.

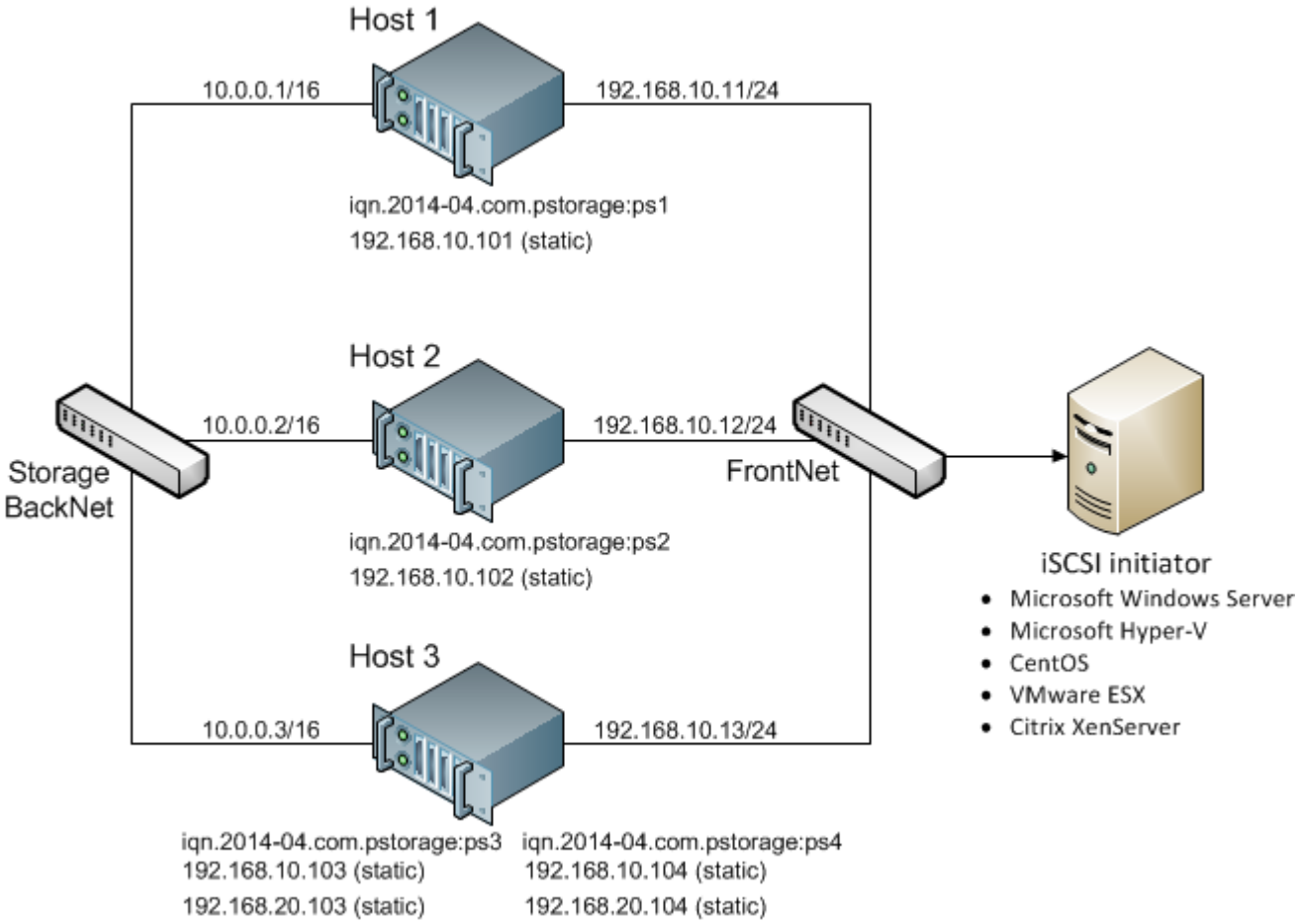
Accessing Virtuozzo Storage Clusters via iSCSI

Virtuozzo Storage allows you to export cluster disk space outside Virtuozzo Storage bounds to operating systems and third-party virtualization solutions. Using dedicated `pstorage-iscsi` tools, you can export Virtuozzo Storage disk space as LUN block devices over iSCSI in a SAN-like manner.

In Virtuozzo Storage, you can create and run multiple iSCSI targets per Virtuozzo Storage cluster Node. In turn, each iSCSI target can have multiple LUNs (virtual disks). At any given moment, each iSCSI target runs on a single Hardware Node. Thanks to High Availability, if a Node fails, iSCSI

targets hosted on it are moved to and relaunched on a healthy Node (for more details on High Availability, see the HA chapter in the Virtuozzo 6 User's Guide).

The figure below shows a typical network configured for exporting Virtuozzo Storage disk space over iSCSI.



In this example are three Virtuozzo Hardware Nodes working in a Virtuozzo Storage cluster. Two Nodes host one iSCSI target each while the third hosts two iSCSI targets. Each Node has a static or dynamic IP address assigned from the Storage BackNet (created along with the Virtuozzo Storage cluster) and the FrontNet. Each iSCSI target has a static IP address assigned from the FrontNet.

Preparing to Work with Virtuozzo Storage iSCSI Targets

On each Virtuozzo Hardware Node, where you need to create and run iSCSI targets, do the following:

1. Make sure the `pstorage-iscsi` and `pstorage-scsi-target-utils` packages are installed on the Hardware Node.

2. Make sure that the Hardware Node has access to the Virtuozzo Storage cluster as client and has an entry in `/etc/fstab`. For more information, see **Setting Up Clients** (p. 26).
3. Create a directory in the Virtuozzo Storage cluster where you will store iSCSI targets and their configuration. For example, `/pstorage/stor1/iscsi`.
4. Set the `ISCSI_ROOT` variable in `/etc/pstorage/iscsi/config` to the directory from the previous step. For example:

```
ISCSI_ROOT=/pstorage/stor1/iscsi
```

5. Enable High Availability support for the Hardware Node. For information on how to do it, see the *Virtuozzo 6 User's Guide*.

You are now ready to create and run iSCSI targets in your Virtuozzo Storage cluster.

Creating and Running Virtuozzo Storage iSCSI Targets

Notes:

1. Each iSCSI target must be assigned at least one unique IP address from frontnet's static pool.

2. The name of each iSCSI target must be unique in the Virtuozzo Storage cluster.

3. Virtuozzo Storage iSCSI targets support persistent reservations to allow iSCSI initiators obtain exclusive access to the specified target's LUNs.

To create and start a target `test1` with the size of 100 GB, the LUN of 1, and the IP address of 192.168.10.100, execute the following commands:

```
# pstorage-iscsi create -n test1 -a 192.168.10.100
IQN: iqn.2014-04.com.pstorage:test1
# pstorage-iscsi lun-add -t iqn.2014-04.com.pstorage:test1 -l 1 -s 100G
# pstorage-iscsi start -t iqn.2014-04.com.pstorage:test1
```

Notes:

1. If you need to change target's IP address, stop the target as described in **Stopping Virtuozzo Storage iSCSI Targets** (p. 52), then run the command `pstorage-iscsi set -t <target_name> -a <new_IP_address>`.

2. If you need to increase the size of a LUN, stop the target as described in **Stopping Virtuozzo Storage iSCSI Targets** (p. 52), then run the command `pstorage-iscsi lun-grow -t <target_name> -l <lun_ID> -s <new_size>`.

To check that the target is up, run the `pstorage-iscsi list` command with the target's name as the option. For example:

```
[root@dhcp-10-30-24-73 ~]# pstorage-iscsi list -t iqn.2014-04.com.pstorage:test1
Target iqn.2014-04.com.pstorage:test1:
 Portals:     192.168.10.100
 Status:      running
 Registered:  yes
 Host:        fefacc38a2f140ca
```

```
 LUN:   1, Size:  102400M, Used:     1M, Online: Yes
```

For information about the command output, see **Listing Virtuozzo Storage iSCSI Targets** (p. 51).

iSCSI initiators can now access the target `iqn.2014-04.com.pstorage:test1` via the portal 192.168.10.100.

Performance Tips

- Spread iSCSI targets evenly across Hardware Nodes in the cluster. For example, 10 Hardware Nodes with 1 iSCSI target per each will perform better than a single Hardware Node with 10 iSCSI targets on it.
- More LUNs per fewer iSCSI targets will perform better than fewer LUNs per more iSCSI targets.

Listing Virtuozzo Storage iSCSI Targets

Using the `pstorage-iscsi list` command, you can list all iSCSI targets registered on a Virtuozzo Storage Node or display detailed information about a specific iSCSI target on a Virtuozzo Storage Node.

To list all iSCSI targets registered on a Virtuozzo Storage Node, run the command as follows:

```
# pstorage-iscsi list
IQN                              STATUS  LUNs HOST             PORTAL(s)
iqn.2014-04.com.pstorage:test1  running 1    fefacc38a2f140ca 192.168.10.100
iqn.2014-04.com.pstorage:test2  running 1    fefacc38a2f140ca 192.168.10.101
iqn.2014-04.com.pstorage:test3  stopped 1    fefacc38a2f140ca 192.168.10.102
iqn.2014-04.com.pstorage:test4  stopped 0    fefacc38a2f140ca 192.168.10.103
```

To display detailed information about an iSCSI target registered on a Virtuozzo Storage Node, run the `pstorage-iscsi list` command with the target's name as the option. For example:

```
# pstorage-iscsi list -t iqn.2014-04.com.pstorage:test1
Target iqn.2014-04.com.pstorage:test1:
 Portals:     192.168.10.100
 Status:      running
 Registered:  yes
 Host:        fefacc38a2f140ca
 LUN:   1, Size:  102400M, Used:     1M, Online: Yes
```

The command outputs above show the following data:

Item	Description
Target	Unique alphanumeric name of the iSCSI target.
Portals	Target's IP address(es).
Status	Target's current state. • `running`: target is running and ready for use (for local targets). • `stopped`: target is stopped (for local targets). • `service failed`: the iSCSI service is down (for local targets). • `remote`: target is registered on a different Node.

	• `unregistered`: target is not registered on any Node in the Virtuozzo Storage cluster.
Registered	Whether or not the target is registered on the host which ID is shown in the **Host** entry.
Host	Virtuozzo Storage Hardware Node ID.
LUN	Virtual disk's integer number within the target.
Size	Virtual disk's logical size (16 TB maximum).
Used	Virtual disk's physical size. The physical size can be smaller than logical due to the expanding format of the virtual disk (for more information, see the *Virtuozzo 6 User's Guide*).
Online	• `Yes`: the LUN is visible to and can be mounted by iSCSI initiators. • `No`: the LUN is invisible to and cannot be mounted by iSCSI initiators.

Transferring Virtuozzo Storage iSCSI Targets Between Virtuozzo Storage Nodes

You can transfer stopped iSCSI targets between Virtuozzo Storage Nodes. After the transfer, you will be able to start and manage the iSCSI target on the destination Node. On the source Node, you will only be able to delete the transferred target with the `--force` option (for more details, see **Deleting Virtuozzo Storage iSCSI Targets** (p. 53)).

To transfer an iSCSI target, do the following:

1. Make sure the target is stopped. For more details, see **Stopping Virtuozzo Storage iSCSI Targets** (p. 52).
2. Unregister the target on its current Node with the `pstorage-iscsi unregister` command. For example:

```
# pstorage-iscsi unregister -t iqn.2014-04.com.pstorage:test1
```

3. Register the target on the new Node with the `pstorage-iscsi register` command. For example:

```
# pstorage-iscsi register -t iqn.2014-04.com.pstorage:test1
```

Stopping Virtuozzo Storage iSCSI Targets

To stop a Virtuozzo Storage iSCSI target to which no initiators are connected, use the `pstorage-iscsi stop` command. For example, for the target `iqn.2014-04.com.pstorage:test1`:

```
# pstorage-iscsi stop -t iqn.2014-04.com.pstorage:test1
```

If one or more iSCSI initiators are still connected to the target, you will be informed as follows:

```
# pstorage-iscsi stop -t iqn.2014-04.com.pstorage:test1
initiators still connected
 Initiator:   iqn.1994-05.com.redhat:c678b9f6f0 (192.168.30.100)
Unable stop target iqn.2014-04.com.pstorage:test1
```

In this case, disconnect the iSCSI initiator according to the product manual and run the `pstorage-iscsi stop` command again.

To forcibly stop a target to which one or more initiators are still connected, add the `-f` option to the command above. For example:

```
# pstorage-iscsi stop -t iqn.2014-04.com.pstorage:test1 -f
```

Breaking the iSCSI connection in such a way may result in I/O errors on the iSCSI initiator's side.

Deleting Virtuozzo Storage iSCSI Targets

You can delete Virtuozzo Storage iSCSI targets with the pstorage-iscsi delete command. Deleting a Virtuozzo Storage iSCSI target, you will also delete all the LUNs within it.

To delete a Virtuozzo Storage iSCSI target, do the following:

1. Make sure the target is stopped (for more details, see **Stopping Virtuozzo Storage iSCSI Targets** (p. 52)).
2. Run the `pstorage-iscsi delete` command with the target name as the option. For example:

```
# pstorage-iscsi delete -t iqn.2014-04.com.pstorage:test1
```

To delete a stopped iSCSI target registered on a different host, add the `--force` option to the `pstorage-iscsi delete` command. For example:

```
# pstorage-iscsi delete -t iqn.2014-04.com.pstorage:test1 --force
```

Accessing Virtuozzo Storage iSCSI Targets from Operating Systems and Third-Party Virtualization Solutions

This section describes ways to attach Virtuozzo Storage iSCSI targets to a number of operating systems and third-party virtualization solutions.

Accessing Virtuozzo Storage iSCSI Targets from CentOS 6.5

1. Make sure that the `iscsi-initiator-utils` package is installed.
2. Discover the required target by its IP address. For example:

```
# iscsiadm --mode discovery --type sendtargets --portal 192.168.10.100
```

3. Restart the `iscsid` service to rescan for newly added drives:

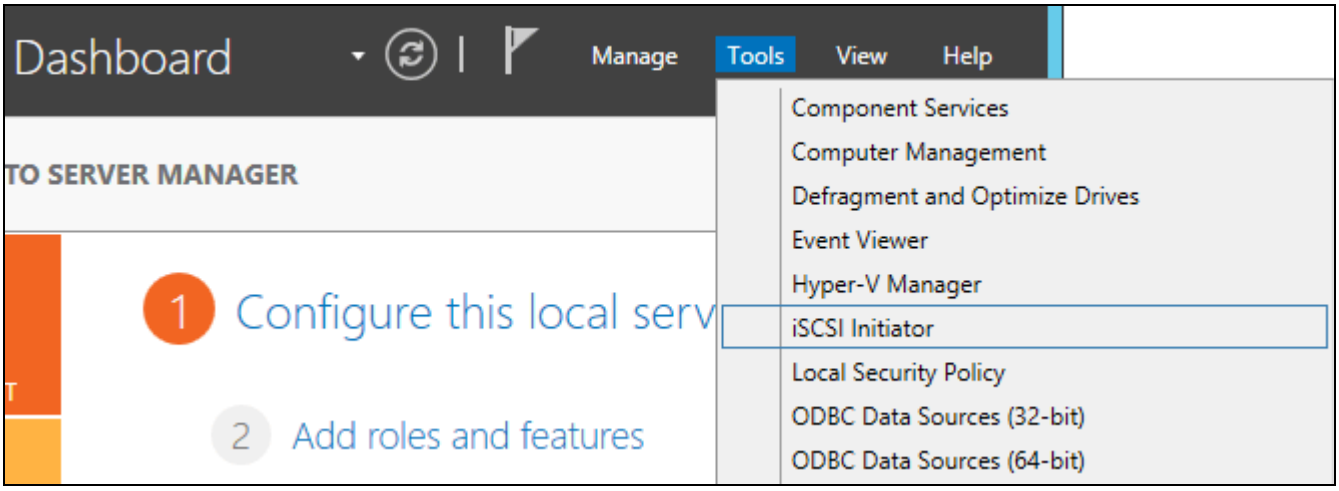
```
# service iscsi restart
```

To check that the new drive has appeared in the system, use `fdisk`, `parted` or similar tools.

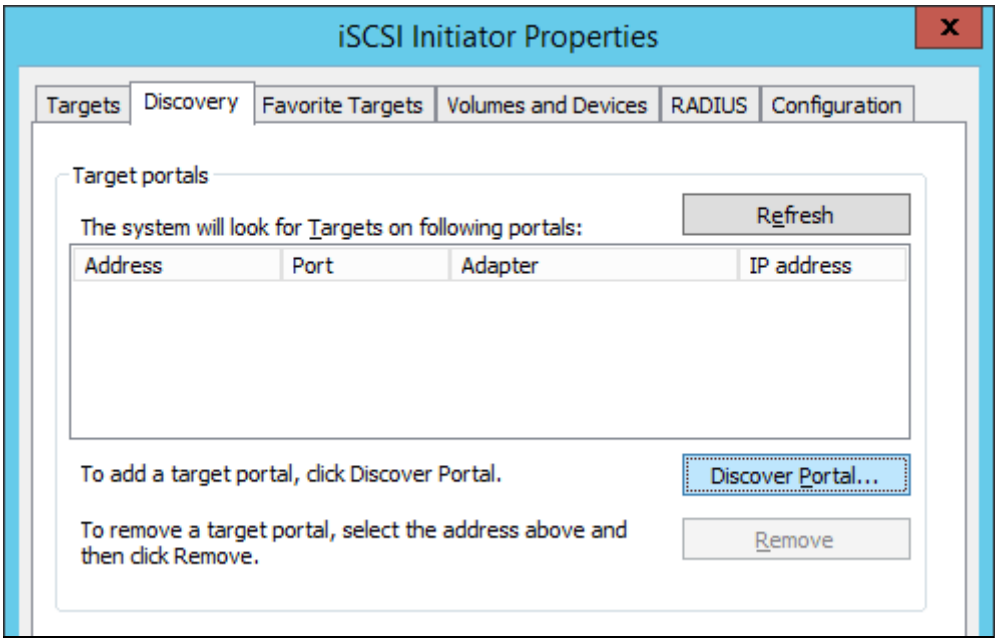
For more information, see the *Red Hat Enterprise Linux Storage Administration Guide* (*https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Storage_Administration_Guide/ch-iscsi.html*).

Accessing Virtuozzo Storage iSCSI Targets from Microsoft Windows Server 2012 R2

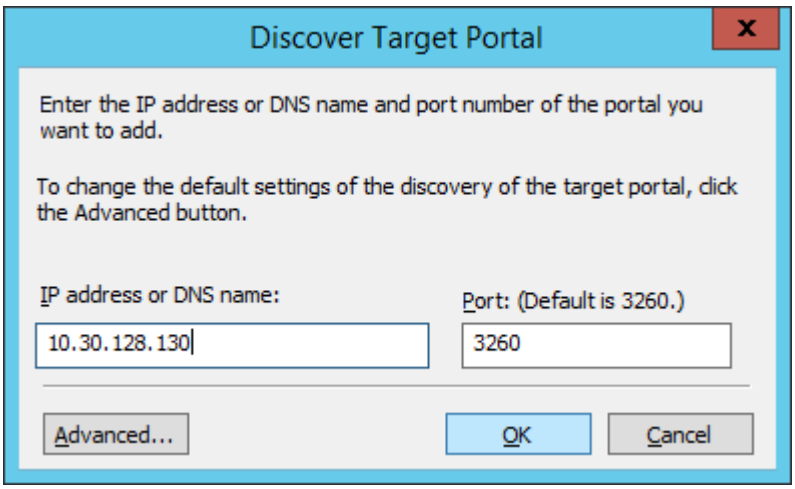
1 In the **Server Manager Dashboard**, click the **Tools** menu in the toolbar and select **iSCSI Initiator**.



2 In the **iSCSI Initiator Properties**, switch to the **Discovery** tab and click **Discover Portal...**.

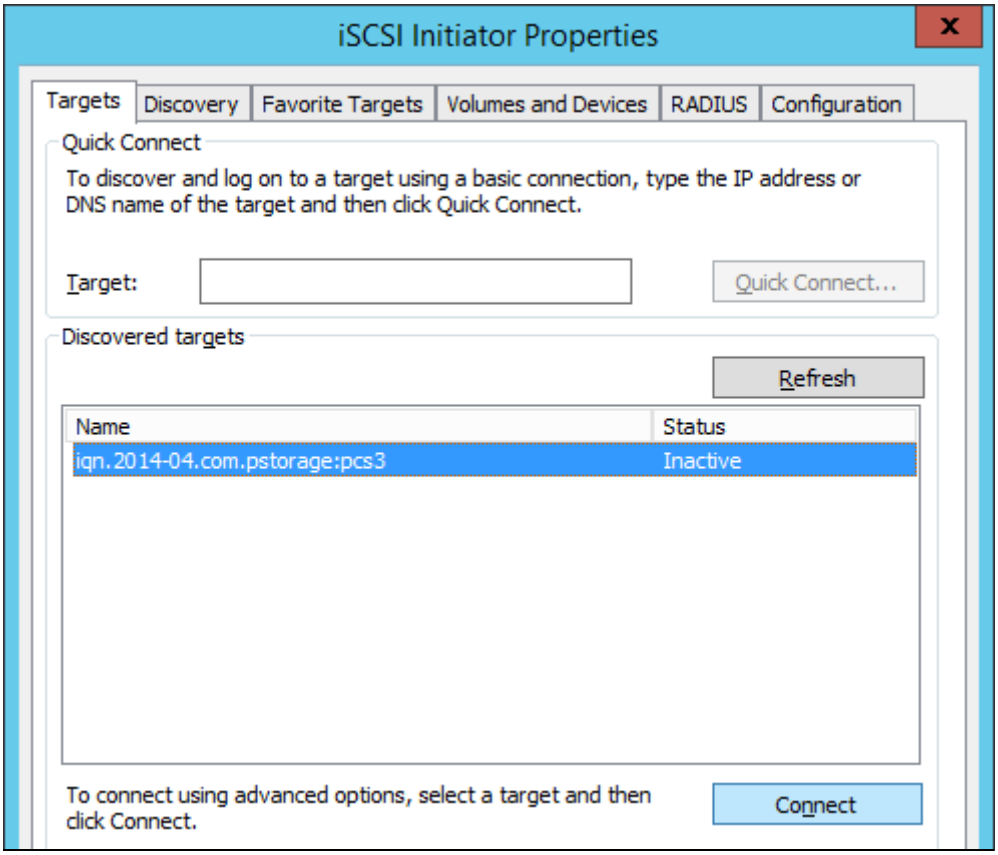


3 In the **Discover Target Portal** window, enter the portal IP address and click **OK**.

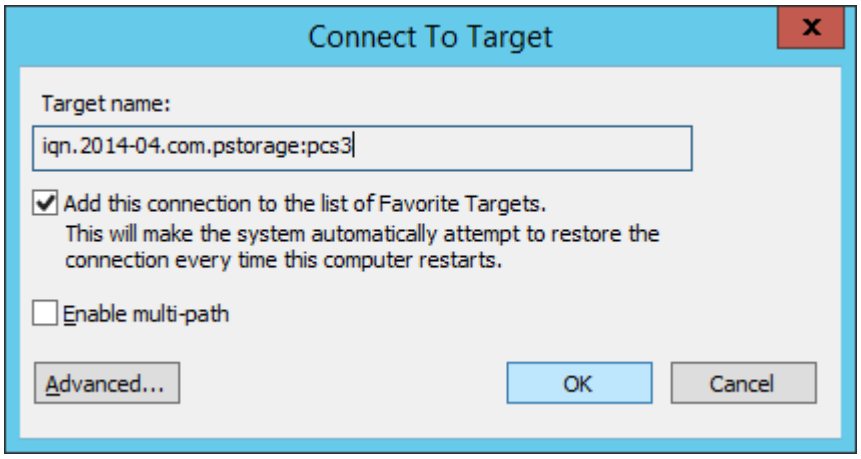


The newly added portal will appear in the **Target portals** section.

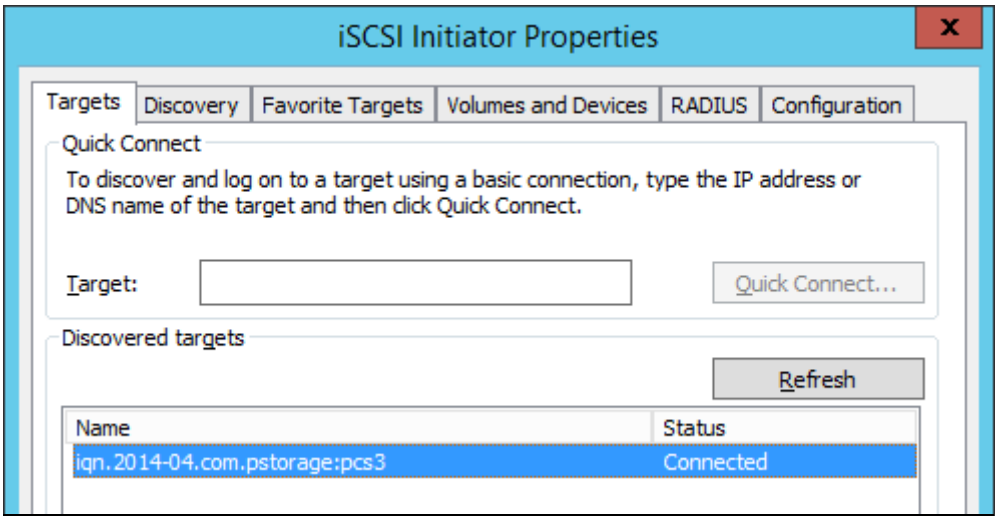
4 On the **iSCSI Initiator Properties** > **Targets** tab, select the new target in the **Discovered targets** section and click **Connect**.



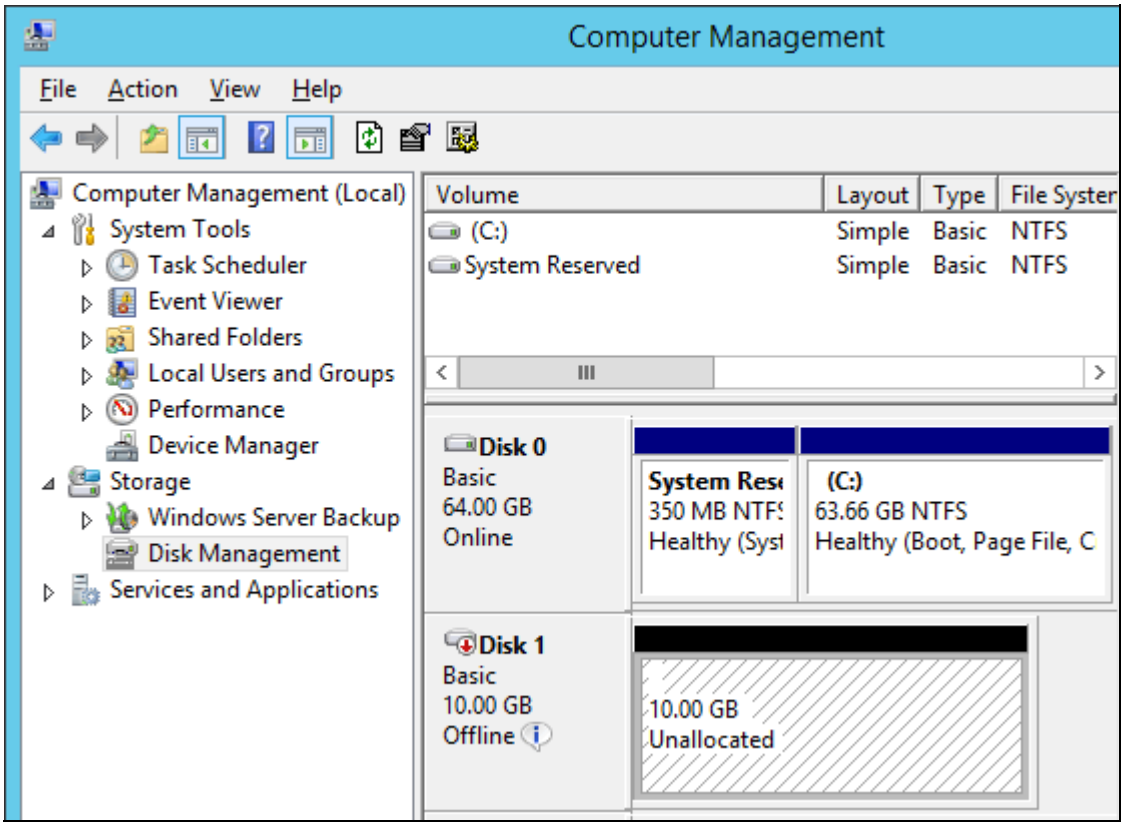
5 In the **Connect to Target** window, click **OK**.



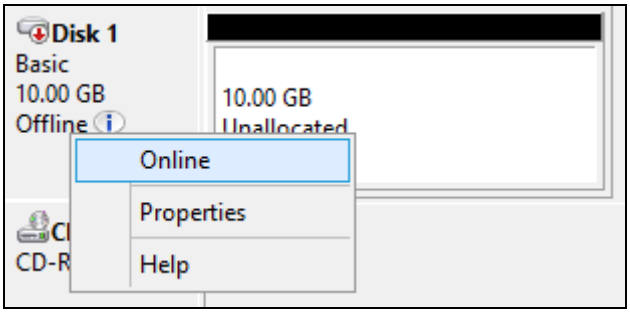
6 Target's **Inactive** status will change to **Connected**.



7 The newly attached disk will appear in **Server Manager Dashboard** > **Computer Management** > **Storage** > **Disk Management**.

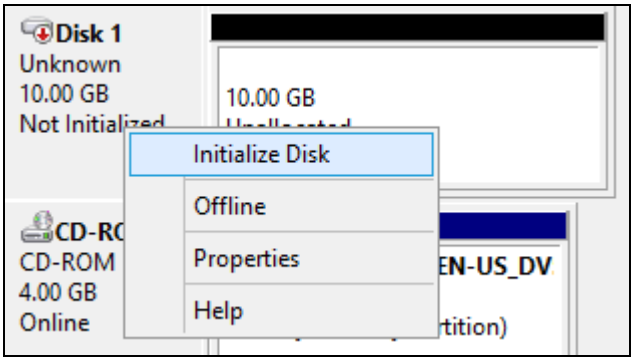


8 Right-click the disk information section and select **Online**.

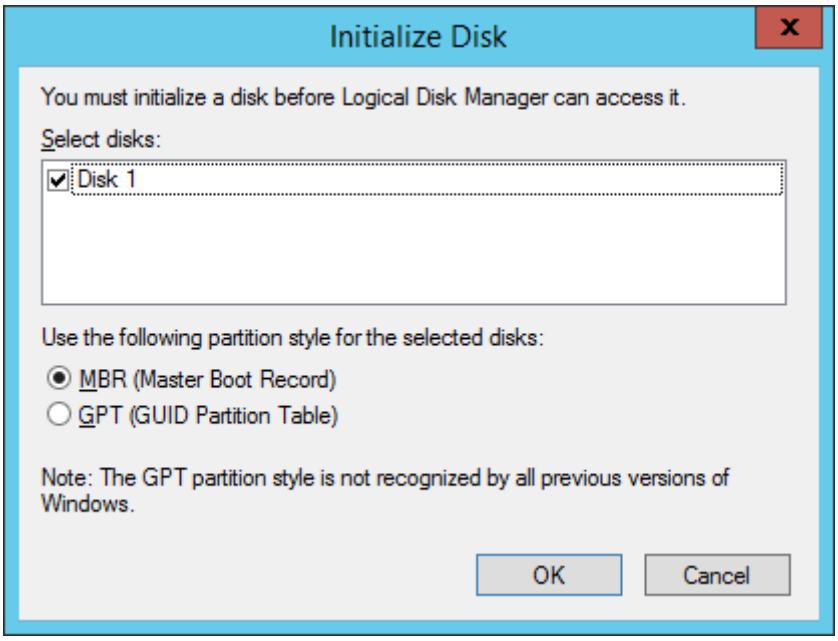


The disk status will change to **Online**.

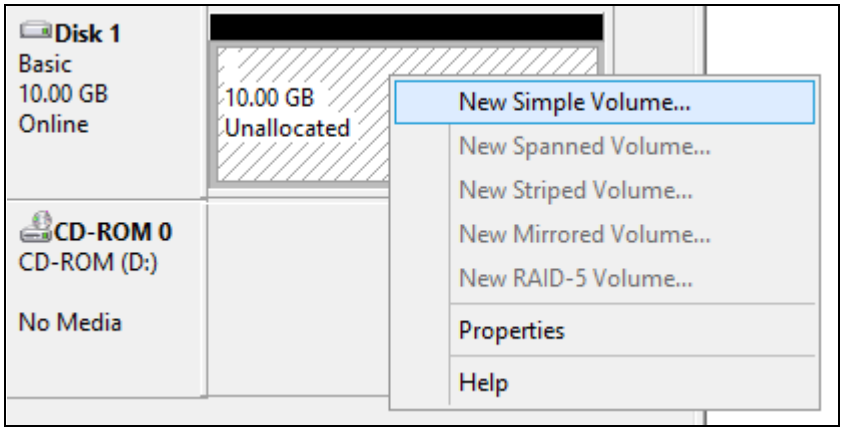
9 Right-click the disk information section and select **Initialize Disk**.



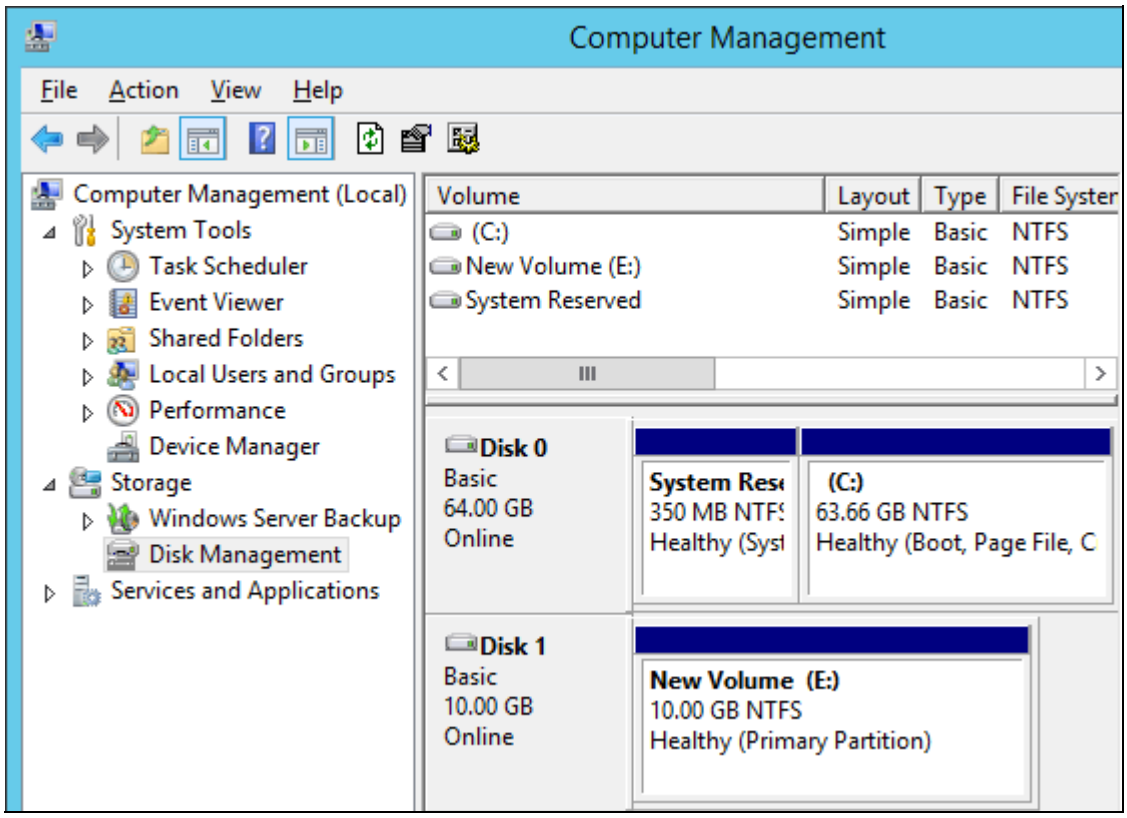
10 In the Initialize Disk window, click **OK**.



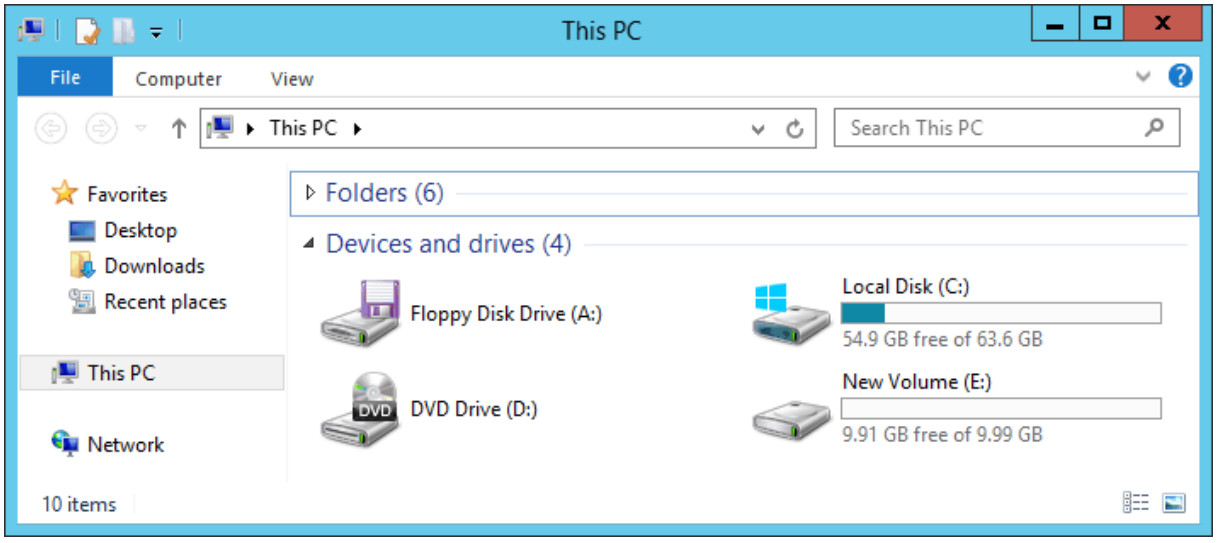
11 Right-click the disk space section, select **New Simple Volume...**, and follow the wizard's instruction to format the new disk to NTFS.



12 The disk state will change to **Healthy**.



13 The new disk will appear in Windows Explorer.



Accessing Virtuozzo Storage iSCSI Targets from VMware ESXi

1 In the **vSphere Client**, switch to the **Configuration** tab, and click **Storage Adapters** in the **Hardware** section.

2. If no software iSCSI adapters have been added, do so by right-clicking in the **Storage Adapters** section and selecting **Add Software iSCSI Adapter...**.
3. Open the software iSCSI adapter's properties, switch to the **Static Discovery** tab and click **Add...**.
4. In the **Add Static Target Server** window, enter the target's IP address and name.
5. Close the software iSCSI adapter's properties window and rescan the adapter as prompted.
6. The newly added iSCSI target will appear in the **Details** section of the software iSCSI adapter you have configured.
7. For more information, see the *VMware vSphere Storage Guide* (*http://pubs.vmware.com/vsphere-50/topic/com.vmware.ICbase/PDF/vsphere-esxi-vcenter-server-50-storage-guide.pdf*).

Accessing Virtuozzo Storage iSCSI Targets from Citrix XenServer 6.2

1. In XenCenter, switch to the **Storage** tab and click **New SR...**.
2. In the **New Storage Repository** window:
   1. In the **Type** section, select the **Software iSCSI** option,
   2. In the **Name** section, provide a name or leave the default,
   3. In the **Location** section, enter target's IP address in the **Target Host** field, click **Discover IQNs** and select the desired target, then click **Discover LUNs** and select the desired LUN.
3. Click **Finish** to format the disk.

The new storage repository will appear in XenCenter.

For more information, see XenCenter documentation at http://support.citrix.com/proddocs/topic/xencenter-62/xs-xc-storage.html.

Accessing Virtuozzo Storage iSCSI Targets from Microsoft Hyper-V

Note: Names of the targets to be mounted must not contain underscore characters.

1. Make sure that Microsoft iSCSI Initiator Service, `MSiSCSI`, is running.
2. Discover a new target portal. For example, for the portal 192.168.10.100, run:

```
PS C:\Users\Administrator>new-iscsitargetportal -targetportaladdress 192.168.10.100
Initiator Instance Name  :
Initiator Portal Address :
IsDataDigest             : False
IsHeaderDigest           : False
TargetPortalAddress      : 192.168.10.100
TargetPortalPortNumber   : 3260
PSComputerName           :
```

3. Connect to the desired target. For example, for the target iqn.2014-03.com.pstorage:test1

```
PS C:\Users\Administrator> connect-iscsitarget
cmdlet Connect-IscsiTarget at command pipeline position 1
Supply values for the following parameters:
```

```
NodeAddress: iqn.2014-04.com.pstorage:test1
AuthenticationType      : NONE
InitiatorInstanceName   : ROOT\ISCSIPRT\0000_0
InitiatorNodeAddress    : iqn.1991-05.com.microsoft:win-l2dj7g36n7e.sw.swsoft.com
InitiatorPortalAddress  : 0.0.0.0
InitiatorSideIdentifier : 400001370000
IsConnected             : True
IsDataDigest            : False
IsDiscovered            : True
IsHeaderDigest          : False
IsPersistent            : False
NumberOfConnections     : 1
SessionIdentifier       : ffffe00000b5e020-4000013700000005
TargetNodeAddress       : iqn.2014-04.com.pstorage:test1
TargetSideIdentifier    : 0001
PSComputerName          :
```

4 To check that the disk has been connected, run

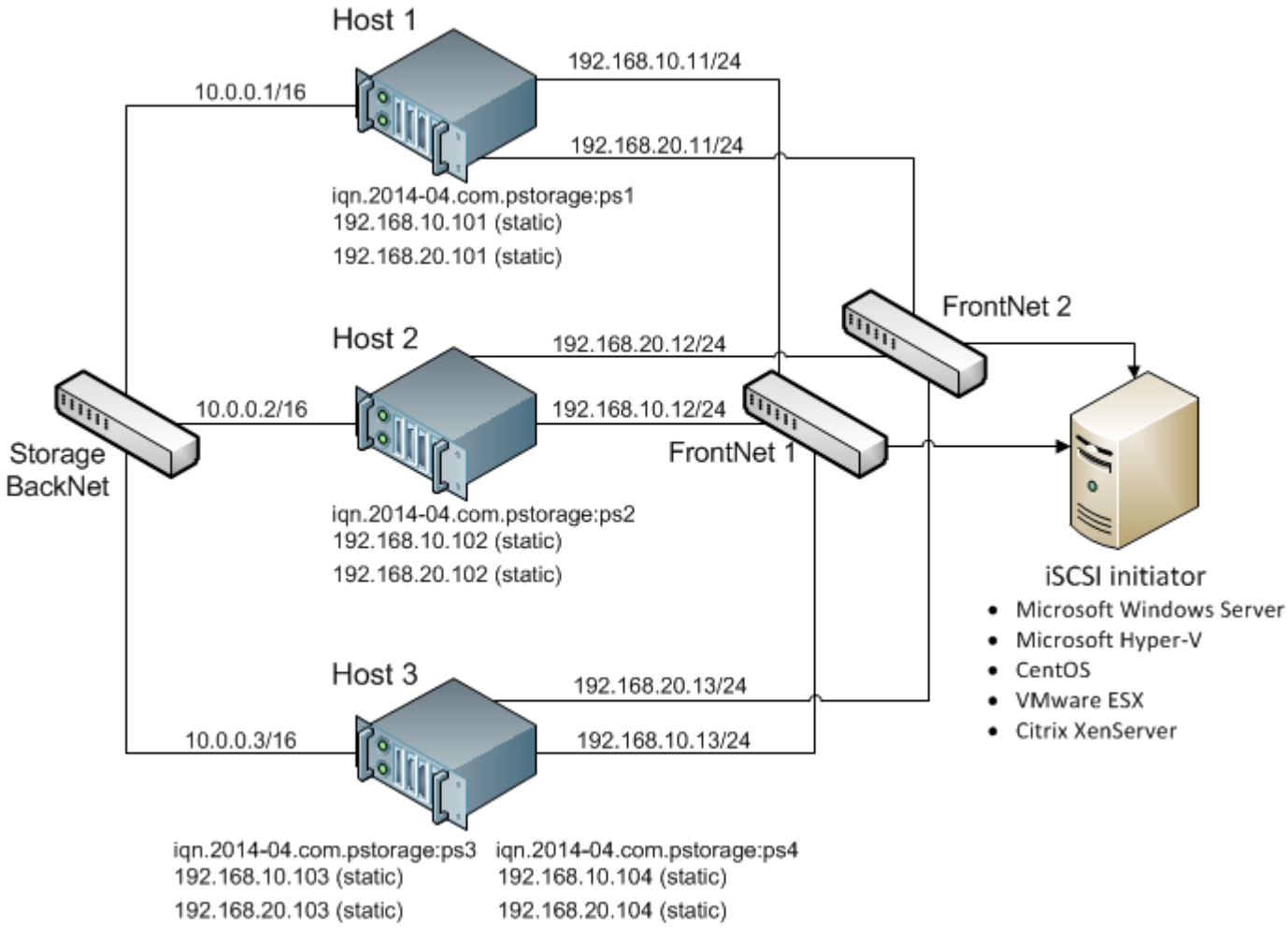
```
PS C:\Users\Administrator> get-disk
Number Friendly Name                              OperationalStatus
Total Size Partition Style
------ -------------                              -----------------                  --
-------- ---------------
1      IET VIRTUAL-DISK SCSI Disk Device          Offline
100 GB RAW
<...>
```

You can now initialise the newly mounted disk for use in Microsoft Hyper-V.

For more information, see *iSCSI Cmdlets in Windows PowerShell* (*http://technet.microsoft.com/en-us/library/hh826099.aspx*).

Configuring Multipath I/O for Virtuozzo Storage iSCSI Targets

Multipath I/O is a technique called to increase fault tolerance and performance by establishing multiple paths to the same iSCSI target. The figure below shows a typical multipath-enabled network configured for exporting Virtuozzo Storage disk space over iSCSI.



In this example are three Virtuozzo Hardware Nodes working in a Virtuozzo Storage cluster. Two Nodes host one iSCSI target each while the third hosts two iSCSI targets. Each Hardware Node is assigned a static or dynamic IP address from the FrontNet 1 and the same from the FrontNet 2. In turn, each iSCSI target is assigned a static IP address from the FrontNet 1 and a static IP address from the FrontNet 2. In case one of the frontnets fails, the iSCSI targets will still be accessible via the other one.

To enable multipath I/O for a Virtuozzo Storage iSCSI target, assign to it multiple IP addresses from different networks using the `-a` option. For example, for a Node connected to two networks, 192.168.10.0/24 and 192.168.20.0/24, run the following command:

```
# pstorage-iscsi create -n ps1 -a 192.168.10.101 -a 192.168.20.101
```

Managing CHAP Accounts for Virtuozzo Storage iSCSI Targets

Virtuozzo Storage allows you to restrict access to iSCSI targets by means of CHAP authentication.

To make use of CHAP authentication, you need to:

1. Create a CHAP account.
2. Create an iSCSI target bound to this CHAP account.

These actions are described in detail in the following subsections.

Creating CHAP Accounts for Virtuozzo Storage iSCSI Targets

To create a CHAP account, use the `pstorage-iscsi account-create` command. For example, to create the CHAP account `user1`:

```
# pstorage-iscsi account-create -u user1
Enter password:
Verify password:
```

Note: The password should be 12 to 16 characters long for Windows clients to be able to establish connection.

Creating Virtuozzo Storage iSCSI Targets Bound to CHAP Accounts

To create a Virtuozzo Storage iSCSI target bound to a CHAP account, use the `pstorage-iscsi create` command with the additional `-u` option. For example, create a target bound to the CHAP account `user1`:

```
# pstorage-iscsi create -n test1 -a 192.168.10.100 -u user1
IQN: iqn.2014-04.com.pstorage:test1
```

Changing the CHAP Account Password

To change the password of a CHAP account, use the `pstorage-iscsi account-set` command. For example, to change the password of the CHAP account `user1`:

```
# pstorage-iscsi account-set -u user1
Enter password:
Verify password:
```

Note: The password should be 12 to 16 characters long for Windows clients to be able to establish connection.

The new password will become active after target reboot.

Listing CHAP Accounts and Virtuozzo Storage iSCSI Targets Assigned to Them

To list existing CHAP accounts, use the `pstorage-iscsi account-list` command. For example:

```
# pstorage-iscsi account-list
user1
```

To list Virtuozzo Storage iSCSI targets assigned to a specific CHAP account, use the `pstorage-iscsi account-list` command with the `-u` option. For example, to list iSCSI targets assigned to the CHAP account `user1`:

```
# pstorage-iscsi account-list -u user1
iqn.2014-04.com.pstorage:test1
```

Managing LUN Snapshots

As with virtual machines, you can create and manage snapshots of LUNs. At that, to create a snapshot of the entire target, you will need to create snapshots of each LUN within it.

Creating LUN Snapshots

To create a snapshot of a LUN in an iSCSI target, use the `pstorage-iscsi snapshot-create` command. For example, for LUN 1 on target `iqn.2014-04.com.pstorage:test1`:

```
# pstorage-iscsi snapshot-create -t iqn.2014-04.com.pstorage:test1 -l 1
Snapshot a1f54314-bc06-40c6-a587-965feb9d85bb successfully created.
```

Note: To generate a UUID manually, use `uuidgen`.

Listing LUN Snapshots

To list snapshots for the specified LUN, use the `pstorage-iscsi snapshot-list` command. For example, for LUN 1 on `target iqn.2014-04.com.pstorage:test1`:

```
# pstorage-iscsi snapshot-list -t iqn.2014-04.com.pstorage:stor4 -l 1
CREATED             C UUID                                 PARENT_UUID
2014-04-11 13:16:51   a1f54314-bc06-40c6-a587-965feb9d85bb 00000000-0000-0000-0000-
000000000000
2014-04-11 13:16:57 * 9c98b442-7482-4fd0-9c45-9259374ca84e a1f54314-bc06-40c6-a587-
965feb9d85bb
```

In the output above, the asterisk in the column `C` indicates the current snapshot, while the column `PARENT_UUID` shows snapshot dependency or history.

Switching Between LUN Snapshots

To switch to the specified LUN snapshot, use the `pstorage-iscsi snapshot-switch` command. For example:

```
# pstorage-iscsi snapshot-switch -u a1f54314-bc06-40c6-a587-965feb9d85bb
```

After you switch to a snapshot, the current LUN image will be removed.

Note: You can only switch between snapshots, if the LUN is offline.

Viewing LUN Snapshot Information

To view information about the specified snapshot, use the `pstorage-iscsi snapshot-info` command. For example:

```
# pstorage-iscsi snapshot-info -u 9c98b442-7482-4fd0-9c45-9259374ca84e
Target: iqn.2014-04.com.pstorage:stor4
LUN: 1
```

```
Created: 2014-04-11 13:16:57
Parent: 00000000-0000-0000-0000-000000000000}
{a1f54314-bc06-40c6-a587-965feb9d85bb}
{9c98b442-7482-4fd0-9c45-9259374ca84e
Description: None
```

Deleting LUN Snapshots

To delete the specifed LUN snapshot, use the `pstorage-iscsi snapshot-delete` command. For example:

```
# pstorage-iscsi snapshot-delete -u a1f54314-bc06-40c6-a587-965feb9d85bb
```

If the snapshot has no any children, it will be deleted. If the snapshot has a single child, it will be merged to that child.

Notes:

1. You can only delete offline snapshots.

2. Deleting a snapshot that has multiple children is currently not supported.

Accessing Virtuozzo Storage Clusters via S3 Protocol

Virtuozzo Storage supports the Amazon S3 protocol, enabling service providers to:

- run S3-based services in their own Virtuozzo Storage infrastructures,
- sell S3-based services to customers along with Virtuozzo Storage.

The support for S3 expands the functionality of Virtuozzo Storage and requires a working Virtuozzo Storage cluster. For detailed instructions on how to configure S3 in Virtuozzo Storage, see the *Virtuozzo Object Storage User's Guide*.

CHAPTER 5

Monitoring Virtuozzo Storage Clusters

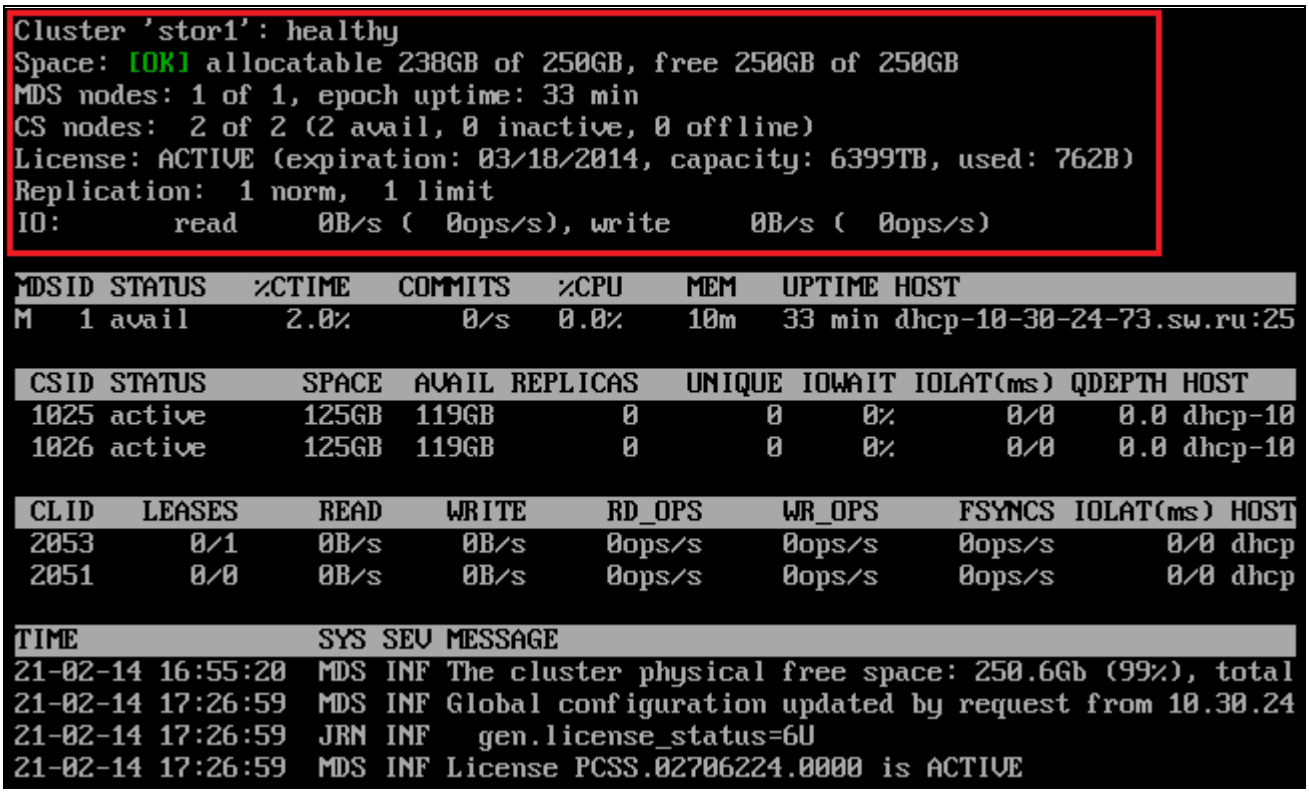
Monitoring a Virtuozzo Storage cluster is very important because it allows you to check the status and health of all computers in the cluster and react as necessary. This chapter explains how to complete the following tasks:

- monitor general cluster parameters (p. 66)
- monitor MDS servers (p. 68)
- monitor chunk servers (p. 70)
- monitor clients (p. 75)
- monitor physical disks (p. 76)
- monitor messages in the cluster event log (p. 78)
- monitor the status of replication parameters (p. 80)

In This Chapter

Monitoring General Cluster Parameters

By monitoring general parameters, you can get detailed information about all components of a Virtuozzo Storage cluster, its overall status and health. To display this information, use the `pstorage -c <cluster_name> top` command, for example:



The command above shows detailed information about the `stor1` cluster. The general parameters (highlighted in red) are explained in the table below.

Parameter	Description
Cluster	Overall status of the cluster: • **healthy**. All chunk servers in the cluster are active. • **unknown**. There is not enough information about the cluster state (e.g., because the master MDS server was elected a while ago). • **degraded**. Some of the chunk servers in the cluster are inactive. • **failure**. The cluster has too many inactive chunk servers; the automatic replication is disabled. • **SMART warning**. One or more physical disks attached to cluster Nodes are in pre-failure condition. For details, see **Monitoring Physical Disks** (p. 76).
Space	Amount of disk space in the cluster: • **free**. Free physical disk space in the cluster. • **allocatable**. Amount of logical disk space available to clients. Allocatable disk space is calculated on the basis of the current replication parameters and free disk space on chunk servers. It may also be limited by license. **Note:** For more information on monitoring and understanding disk space usage in clusters, see **Monitoring Disk Space Usage** (p. 71).

MDS nodes	Number of active MDS servers as compared to the total number of MDS servers configured for the cluster.
epoch time	Time elapsed since the MDS master server election.
CS nodes	Number of active chunk servers as compared to the total number of chunk servers configured for the cluster. The information in parentheses informs you of the number of • Active chunk servers (**avail.**) that are currently up and running in the cluster. • Inactive chunk servers (**inactive**) that are temporarily unavailable. A chunk server is marked as inactive during its first 5 minutes of inactivity. • Offline chunk servers (**offline**) that have been inactive for more than 5 minutes. A chunk server changes its state to offline after 5 minutes of inactivity. Once the state is changed to offline, the cluster starts replicating data to restore the chunks that were stored on the offline chunk server.
License	Key number under which the license is registered on the Key Authentication server and license state. For more information on licenses, see **Managing Virtuozzo Storage Licenses** (p. 43).
Replication	Replication settings. The normal number of chunk replicas and the limit after which a chunk gets blocked until recovered.
IO	Disks IO activity in the cluster: • Speed of read and write I/O operations, in bytes per second. • Number of read and write I/O operations per second.

Monitoring Metadata Servers

MDS servers are a critical component of any Virtuozzo Storage cluster, and monitoring the health and state of MDS servers is a very critical task. To monitor MDS servers, use the `pstorage -c <cluster_name> top` command, for example:

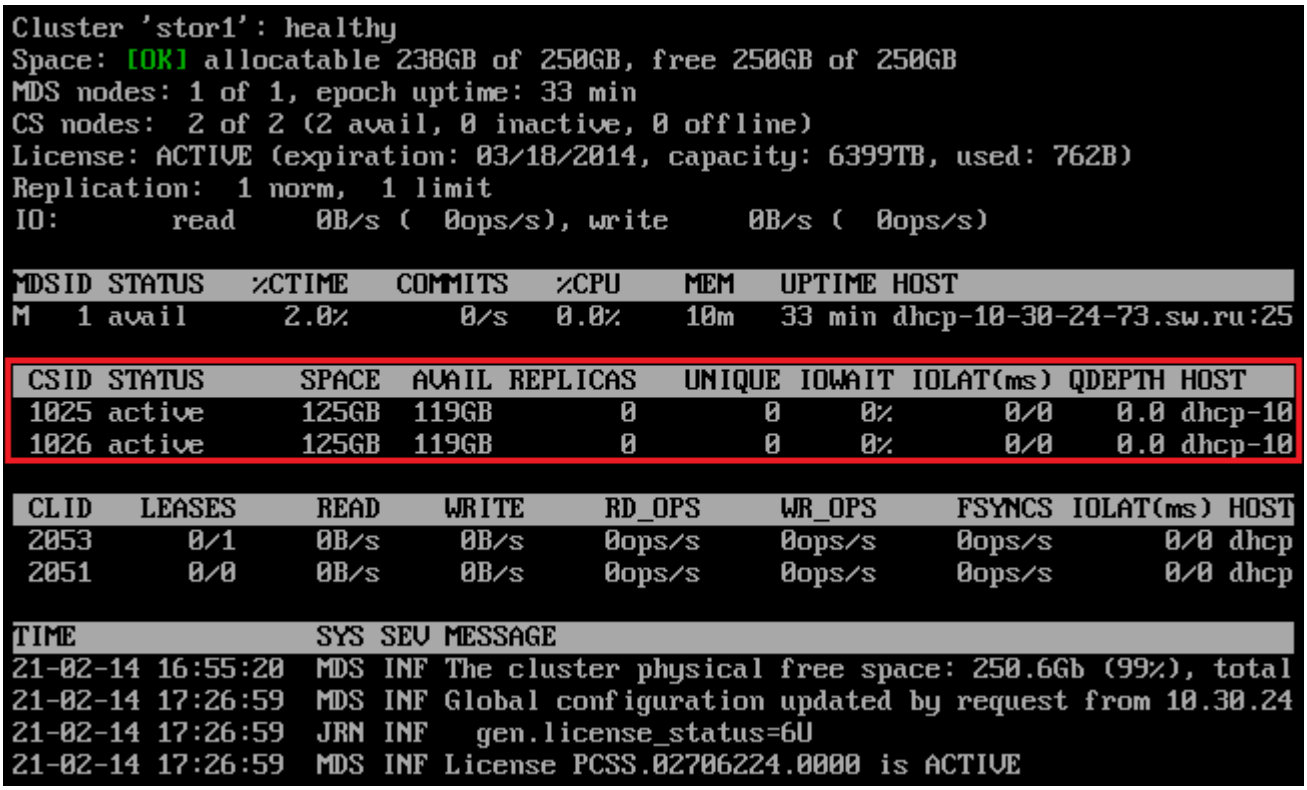
```
Cluster 'stor1': healthy
Space: [OK] allocatable 238GB of 250GB, free 250GB of 250GB
MDS nodes: 1 of 1, epoch uptime: 33 min
CS nodes:  2 of 2 (2 avail, 0 inactive, 0 offline)
License: ACTIVE (expiration: 03/18/2014, capacity: 6399TB, used: 762B)
Replication:  1 norm,  1 limit
IO:       read     0B/s (  0ops/s), write     0B/s (  0ops/s)

MDSID STATUS   %CTIME   COMMITS   %CPU    MEM   UPTIME HOST
M   1 avail      2.0%       0/s   0.0%    10m   33 min dhcp-10-30-24-73.sw.ru:25

 CSID STATUS      SPACE  AVAIL REPLICAS  UNIQUE IOWAIT IOLAT(ms) QDEPTH HOST
 1025 active      125GB  119GB        0       0     0%       0/0    0.0 dhcp-10
 1026 active      125GB  119GB        0       0     0%       0/0    0.0 dhcp-10

 CLID   LEASES     READ    WRITE     RD_OPS     WR_OPS     FSYNCS IOLAT(ms) HOST
 2053      0/1     0B/s     0B/s     0ops/s     0ops/s     0ops/s       0/0 dhcp
 2051      0/0     0B/s     0B/s     0ops/s     0ops/s     0ops/s       0/0 dhcp

TIME                 SYS SEV MESSAGE
21-02-14 16:55:20  MDS INF The cluster physical free space: 250.6Gb (99%), total
21-02-14 17:26:59  MDS INF Global configuration updated by request from 10.30.24
21-02-14 17:26:59  JRN INF   gen.license_status=6U
21-02-14 17:26:59  MDS INF License PCSS.02706224.0000 is ACTIVE
```

The command above shows detailed information about the `stor1` cluster. The monitoring parameters for MDS servers (highlighted in red) are explained in the table below:

Parameter	Description
MDSID	MDS server identifier (ID). The letter **"M"** before ID, if present, means that the given server is the master MDS server.
STATUS	MDS server status.
%CTIME	Total time the MDS server spent writing to the local journal.
COMMITS	Local journal commit rate.
%CPU	MDS server activity time.
MEM	Amount of physical memory the MDS server uses.
UPTIME	Time elapsed since the last MDS server start.
HOST	MDS server hostname or IP address.

Monitoring Chunk Servers

By monitoring chunk servers, you can keep track of the disk space available in a Virtuozzo Storage cluster. To monitor chunk servers, use the `pstorage -c <cluster_name> top` command, for example:



The command above shows detailed information about the `stor1` cluster. The monitoring parameters for chunk servers (highlighted in red) are explained in the table below:

Parameter	Description
CSID	Chunk server identifier (ID).
STATUS	Chunk server status: • **active**. The chunk server is up and running. • **Inactive**. The chunk server is temporarily unavailable. A chunk server is marked as inactive during its first 5 minutes of inactivity. • **offline**. The chunk server is inactive for more than 5 minutes. After the chunk server goes offline, the cluster starts replicating data to restore the chunks that were stored on the affected chunk server. • **dropped**. The chunk serve was removed by the administrator.
SPACE	Total amount of disk space on the chunk server.

FREE	Free disk space on the chunk server.
REPLICAS	Number of replicas stored on the chunk server.
IOWAIT	Percentage of time spent waiting for I/O operations being served.
IOLAT	Average/maximum time, in milliseconds, the client needed to complete a single IO operation during the last 20 seconds.
QDEPTH	Average chunk server I/O queue depth.
HOST	Chunk server hostname or IP address.
FLAGS	The following flags may be shown for active chunk servers: • `J`: The CS uses a write journal. • `C`: Checksumming is enabled for the CS. Checksumming lets you know when a third party changes the data on the disk. • `D`: Direct I/O, the normal state for a CS without a write journal. • `c`: The chunk server's write journal is clean, there is nothing to commit from the write journaling SSD to the HDD where the CS is located. **Note**: Flags which may be shown for failed chunk servers are described in **Failed Chunk Servers** (p. 105).

Understanding Disk Space Usage

Usually, you get the information on how disk space is used in your cluster with the `pstorage top` command. This command displays the following disk-related information: total space, free space, and allocatable space. For example:

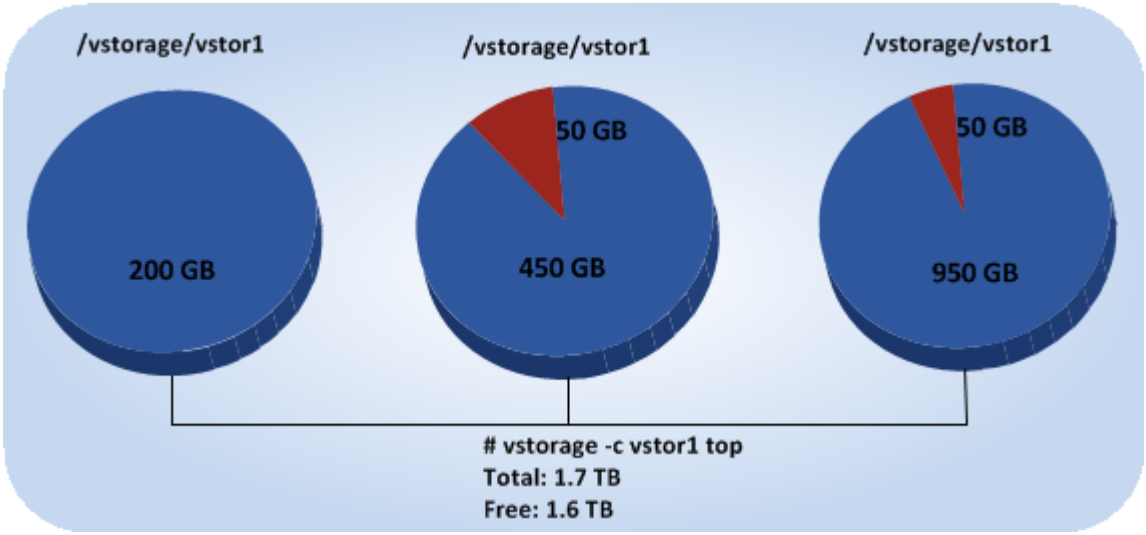
```
# pstorage -c stor1 top
connected to MDS#1
Cluster 'stor1': healthy
Space: [OK] allocatable 180GB of 200GB, free 1.6TB of 1.7TB
...
```

In this command output:

- 1.7TB is the total disk space in the `stor1` cluster. The total disk space is calculated on the basis of used and free disk space on all partitions in the cluster. Used disk space includes the space occupied by all data chunks and their replicas plus the space occupied by any other files stored on the cluster partitions.

  Let us assume that you have a 100 GB partition and 20 GB on this partition are occupied by some files. Now if you set up a chunk server on this partition, this will add 100 GB to the total disk space of the cluster, though only 80 GB of this disk space will be free and available for storing data chunks.
- 1.6TB is the free disk space in the `stor1` cluster. Free disk space is calculated by subtracting the disk space occupied by data chunks and any other files on the cluster partitions from the total disk space.

For example, if the amount of free disk space is 1.6 TB and the total disk space is 1.7 TB, this means that about 100 GB on the cluster partitions are already occupied by some files.



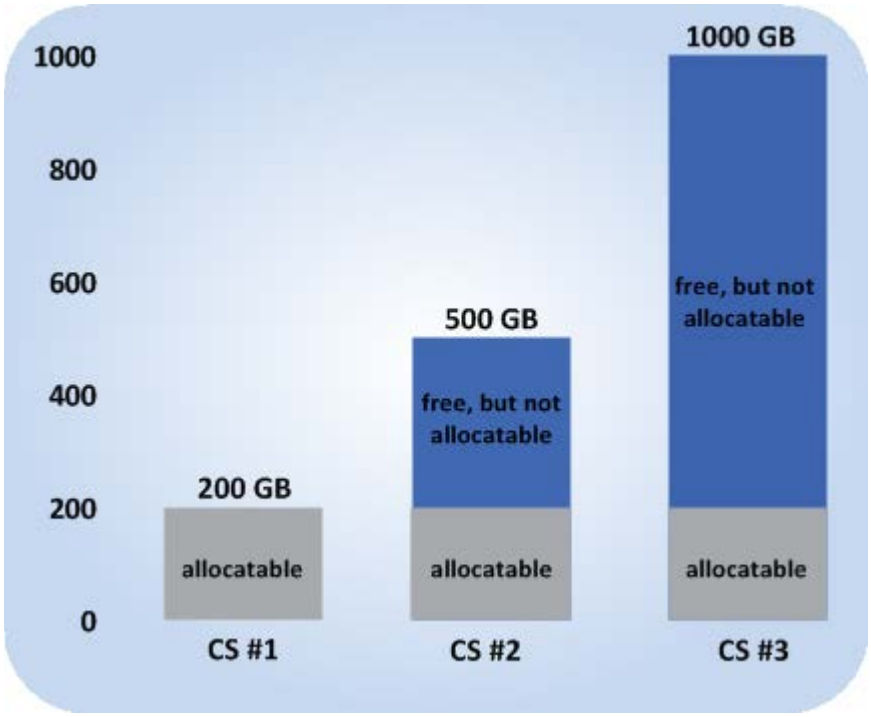
- `allocatable 180GB of 200GB` is the amount of free disk space that can used for storing data chunks. See **Understanding allocatable disk space** below for details.

Understanding allocatable disk space

When monitoring disk space information in the cluster, you also need to pay attention to the space reported by the `pstorage top` utility as *allocatable*. Allocatable space is the amount of disk space that is free and can be used for storing user data. Once this space runs out, no data can be written to the cluster.

To better understand how allocatable disk space is calculated, let us consider the following example:

- The cluster has 3 chunk servers. The first chunk server has 200 GB of disk space, the second one — 500 GB, and the third one — 1 TB.
- The default replication factor of 3 is used in the cluster, meaning that each data chunk must have 3 replicas stored on three different chunk servers.



In this example, the available disk space will equal 200 GB, that is, set to the amount of disk space on the smallest chunk server:

```
# pstorage -c stor1 top
connected to MDS#1
Cluster 'stor1': healthy
Space: [OK] allocatable 180GB of 200GB, free 1.6TB of 1.7TB
...
```

This is explained by the fact that in this cluster configuration each server is set to store one replica for each data chunk. So once the disk space on the smallest chunk server (200 GB) runs out, no more chunks in the cluster can be created until a new chunk server is added or the replication factor is decreased.

If you now change the replication factor to 2, the `pstorage top` command will report the available disk space as 700 GB:

```
# pstorage set-attr -R /pstorage/stor1 replicas=2:1
# pstorage -c stor1 top
connected to MDS#1
Cluster 'stor1': healthy
Space: [OK] allocatable 680GB of 700GB, free 1.6TB of 1.7TB
...
```

The available disk space has increased because now only 2 replicas are created for each data chunk and new chunks can be made even if the smallest chunk server runs out of space (in this case, replicas will be stored on a bigger chunk server).

Note: Allocatable disk space may also be limited by license.

Viewing space occupied by data chunks

To view the total amount of disk space occupied by all user data in the cluster, run the `pstorage top` command and press the V key on your keyboard. Once you do this, your command output should look like the following:

```
# pstorage -c stor1 top
Cluster 'stor1': healthy
Space: [OK] allocatable 180GB of 200GB, free 1.6TB of 1.7TB
MDS nodes: 1 of 1, epoch uptime: 2d  4h
CS nodes:  3 of 3 (3 avail, 0 inactive, 0 offline)
Replication:  2 norm,  1 limit,  4 max
Chunks: [OK] 38 (100%) healthy,  0 (0%) degraded,  0 (0%) urgent,
             0 (0%) blocked,  0 (0%) offline,  0 (0%) replicating,
             0 (0%) overcommitted,  0 (0%) deleting,  0 (0%) void
FS:  1GB in 51 files, 51 inodes,  23 file maps,  38 chunks,  76 chunk replicas
...
```

Note: The **FS** field shows the size of all user data in the cluster without consideration for replicas.

Exploring Chunk States

The table below lists all possible states a chunk can have.

Status	Description
healthy	Percentage of chunks that have enough active replicas. The normal state of chunks.
replicating	Percentage of chunks which are being replicated. Write operations on such chunks are frozen until replication ends.
offline	Percentage of chunks all replicas of which are offline. Such chunks are completely inaccessible for the cluster and cannot be replicated, read from or written to. All requests to an offline chunk are frozen until a CS that stores that chunk's replica goes online. Get offline chunk servers back online as fast as possible to avoid losing data.
void	Percentage of chunks that have been allocated but never used yet. Such chunks contain no data. It is normal to have some void chunks in the cluster.
pending	Percentage of chunks that must be replicated immediately. For a write request from client to a chunk to complete, the chunk must have at least the set minimum amount of replicas. If it does not, the chunk is blocked and the write request cannot be completed. As blocked chunks must be replicated as soon as possible, the cluster places them in a special high-priority replication queue and reports them as pending.
blocked	Percentage of chunks which have fewer active replicas than the set minimum amount. Write requests to a blocked chunk are frozen until it has at least the set minimum amount of replicas. Read requests to blocked chunks are allowed, however, as they still have some active replicas left. Blocked chunks have higher replication priority than degraded chunks. Having blocked chunks in the cluster increases the risk of losing data, so postpone any maintenance on working cluster nodes and get offline chunk servers back online as fast as possible.
degraded	Percentage of chunks with the number of active replicas lower than normal but equal to or higher than the set minimum. Such chunks can be read from and written to. However, in the latter case a degraded chunk becomes urgent.

urgent	Percentage of chunks which are degraded and have non-identical replicas. Replicas of a degraded chunk may become non-identical if some of them are not accessible during a write operation. As a result, some replicas happen to have the new data while some still have the old data. The latter are dropped by the cluster as fast as possible. Urgent chunks do not affect information integrity as the actual data is stored in at least the set minimum amount of replicas.
standby	Percentage of chunks that have one or more replicas in the standby state. A replica is marked standby if it has been inactive for no more than 5 minutes.
overcommitted	Percentage of chunks that have more replicas than normal. Usually these chunks appear after the normal number of replicas has been lowered or a lot of data has been deleted. Extra replicas are eventually dropped, however, this process may slow down during replication.

Monitoring Clients

By monitoring clients, you can check the status and health of servers that you use to access virtual machines and Containers. To monitor clients, use the `pstorage -c <cluster_name> top` command, for example:

```
Cluster 'stor1': healthy
Space: [OK] allocatable 238GB of 250GB, free 250GB of 250GB
MDS nodes: 1 of 1, epoch uptime: 33 min
CS nodes:  2 of 2 (2 avail, 0 inactive, 0 offline)
License: ACTIVE (expiration: 03/18/2014, capacity: 6399TB, used: 762B)
Replication:  1 norm,  1 limit
IO:       read     0B/s (  0ops/s), write     0B/s (  0ops/s)

MDSID STATUS   %CTIME   COMMITS   %CPU    MEM   UPTIME HOST
M   1 avail      2.0%       0/s   0.0%   10m   33 min dhcp-10-30-24-73.sw.ru:25

 CSID STATUS      SPACE  AVAIL REPLICAS   UNIQUE IOWAIT IOLAT(ms) QDEPTH HOST
 1025 active      125GB  119GB        0        0     0%       0/0    0.0 dhcp-10
 1026 active      125GB  119GB        0        0     0%       0/0    0.0 dhcp-10

 CLID   LEASES     READ    WRITE    RD_OPS    WR_OPS    FSYNCS IOLAT(ms) HOST
 2053      0/1     0B/s     0B/s    0ops/s    0ops/s    0ops/s       0/0 dhcp
 2051      0/0     0B/s     0B/s    0ops/s    0ops/s    0ops/s       0/0 dhcp

TIME                SYS SEV MESSAGE
21-02-14 16:55:20   MDS INF The cluster physical free space: 250.6Gb (99%), total
21-02-14 17:26:59   MDS INF Global configuration updated by request from 10.30.24
21-02-14 17:26:59   JRN INF   gen.license_status=6U
21-02-14 17:26:59   MDS INF License PCSS.02706224.0000 is ACTIVE
```

The command above shows detailed information about the `stor1` cluster. The monitoring parameters for clients (highlighted in red) are explained in the table below:

Parameter	Description
CLID	Client identifier (ID).

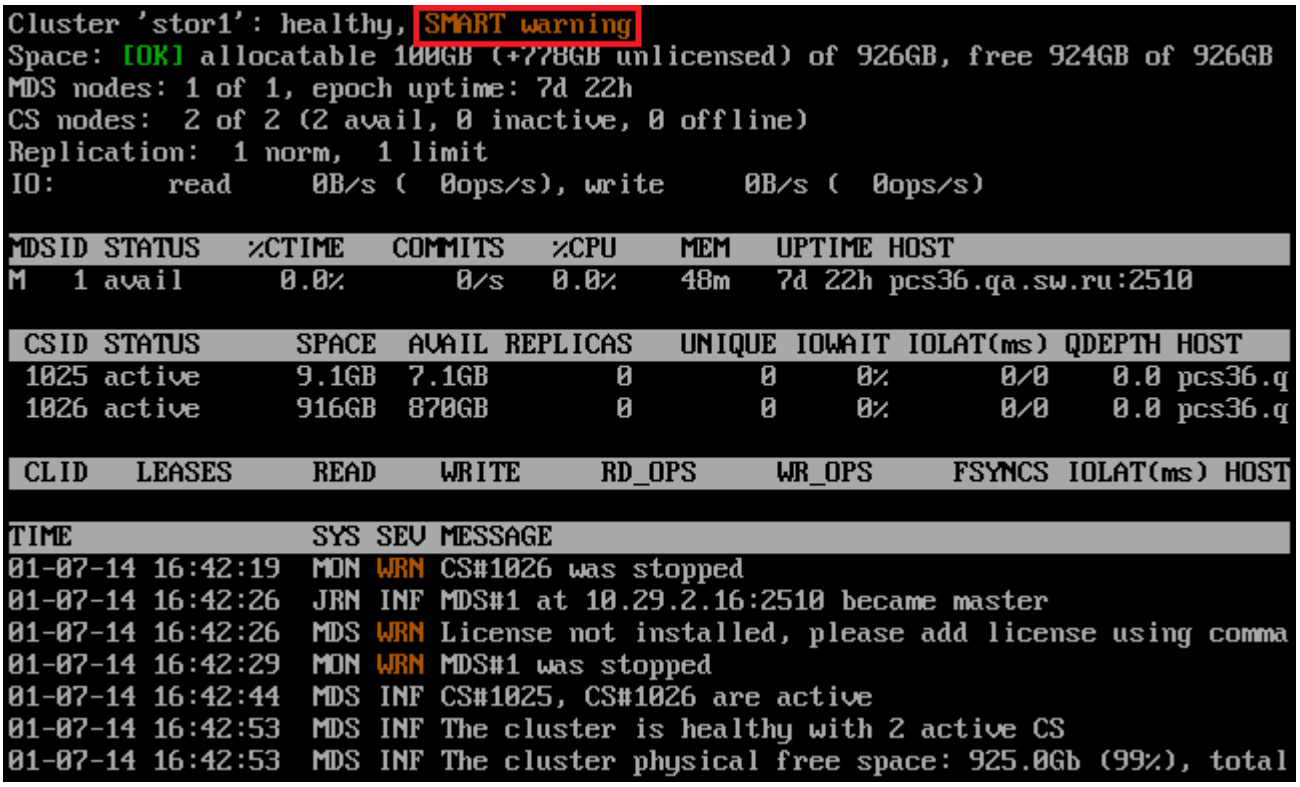
LEASES	Average number of files opened for reading/writing by the client and not yet closed, for the last 20 seconds.
READ	Average rate, in bytes per second, at which the client reads data, for the last 20 seconds.
WRITE	Average rate, in bytes per second, at which the client writes data, for the last 20 seconds.
RD_OPS	Average number of read operations per second the client made, for the last 20 seconds.
WR_OPS	Average number of write operations per second the client made, for the last 20 seconds.
FSYNCS	Average number of sync operations per second the client made, for the last 20 seconds.
IOLAT	Average/maximum time, in milliseconds, the client needed to complete a single IO operation, for the last 20 seconds.
HOST	Client hostname or IP address.

Monitoring Physical Disks

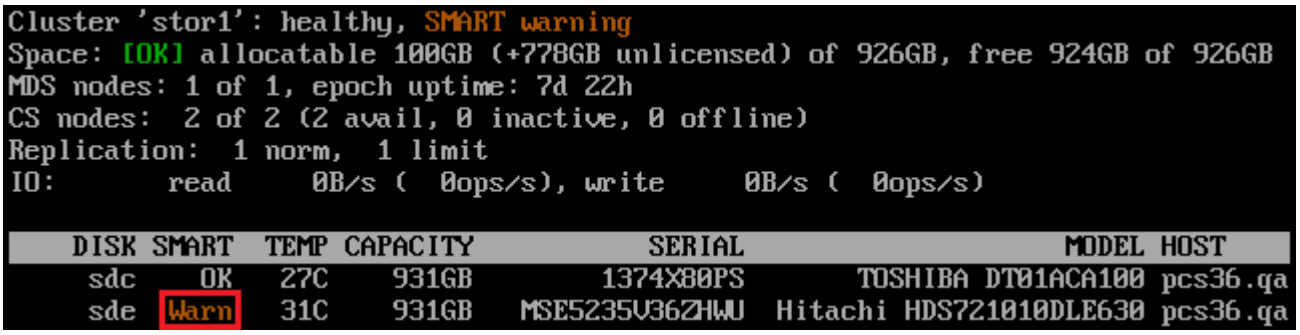
The S.M.A.R.T. status of physical disks is monitored by the `smartctl` tool installed along with Virtuozzo. The tool is run every 10 minutes as a `cron` job also added during Virtuozzo installation. The `smartctl` tool polls all physical disks attached to Hardware Nodes in the cluster, including caching and journaling SSDs, and reports the results to the MDS server.

Note: For the tool to work, enable the S.M.A.R.T. functionality in Node's BIOS.

You can view disk poll results for the last 10 minutes in the output of the `pstorage top` command. For example:



If the **SMART warning** message is shown in the main table, one of the physical disks is in pre-failure condition according to S.M.A.R.T. Press **d** to switch to the disks table to see more details. For example:



The disks table shows the following parameters:

Parameter	Description
DISK	Disk name assigned by operating system.
SMART	Disk's S.M.A.R.T. status: • **OK**: The disk is healthy. • **Warn**: The disk is in pre-failure condition. Pre-failure condition means that at least one of these S.M.A.R.T. counters is nonzero:

	• **Reallocated Sector Count** • **Reallocated Event Count** • **Current Pending Sector Count** • **Offline Uncorrectable**
TEMP	Disk temperature in Celsius.
CAPACITY	Disk capacity.
SERIAL	Disk serial number.
MODEL	Disk model.
HOST	Disk's host address.

Note: To disable S.M.A.R.T. disk monitoring, delete the corresponding `cron` job.

Monitoring Event Logs

You can use the `pstorage -c <cluster_name> top` utility to monitor significant events happening in a Virtuozzo Storage cluster, for example:

```
Cluster 'stor1': healthy
Space: [OK] allocatable 238GB of 250GB, free 250GB of 250GB
MDS nodes: 1 of 1, epoch uptime: 33 min
CS nodes:  2 of 2 (2 avail, 0 inactive, 0 offline)
License: ACTIVE (expiration: 03/18/2014, capacity: 6399TB, used: 762B)
Replication:  1 norm,  1 limit
IO:       read     0B/s (  0ops/s), write     0B/s (  0ops/s)

MDSID STATUS   %CTIME   COMMITS   %CPU    MEM   UPTIME HOST
M   1 avail      2.0%       0/s   0.0%    10m   33 min dhcp-10-30-24-73.sw.ru:25

 CSID STATUS      SPACE  AVAIL REPLICAS   UNIQUE IOWAIT IOLAT(ms) QDEPTH HOST
 1025 active      125GB  119GB        0        0     0%       0/0    0.0 dhcp-10
 1026 active      125GB  119GB        0        0     0%       0/0    0.0 dhcp-10

 CLID   LEASES     READ    WRITE    RD_OPS    WR_OPS    FSYNCS IOLAT(ms) HOST
 2053      0/1     0B/s     0B/s    0ops/s    0ops/s    0ops/s       0/0 dhcp
 2051      0/0     0B/s     0B/s    0ops/s    0ops/s    0ops/s       0/0 dhcp

TIME                 SYS SEV MESSAGE
21-02-14 16:55:20  MDS INF The cluster physical free space: 250.6Gb (99%), total
21-02-14 17:26:59  MDS INF Global configuration updated by request from 10.30.24
21-02-14 17:26:59  JRN INF   gen.license_status=6U
21-02-14 17:26:59  MDS INF License PCSS.02706224.0000 is ACTIVE
```

The command above shows the latest events in the `stor1` cluster. The information on events (highlighted in red) is given in the table with the following columns:

Column	Description
TIME	Time when the event happened.
SYS	Component of the cluster where the event happened (e.g., **MDS** for an MDS server or **JRN** for local journal).
SEV	Event severity.
MESSAGE	Event description.

Exploring Basic Events

The table below describes the basic events displayed when you run the `pstorage top` utility.

Event	Severity	Description
`MDS#N (addr:port) lags behind for more than 1000 rounds`	JRN err	Generated by the MDS master server when it detects that MDS#*N* is stale. This message may indicate that some MDS server is very slow and lags behind.
`MDS#N (addr:port) didn't accept commits for M sec`	JRN err	Generated by the MDS master server if MDS#*N* did not accept commits for *M* seconds. MDS#*N* gets marked as *stale*. This message may indicate that the MDS service on MDS#*N* is experiencing a problem. The problem may be critical and should be resolved as soon as possible.
`MDS#N (addr:port) state is outdated and will do a full resync`	JRN err	Generated by the MDS master server when MDS#*N* will do a full resync. MDS#*N* gets marked as *stale*. This message may indicate that some MDS server was too slow or disconnected for such a long time that it is not really managing the state of metadata and has to be resynchronized. The problem may be critical and should be resolved as soon as possible.
`MDS#N at <addr:port> became master`	JRN info	Generated every time a new MDS master server is elected in the cluster. Frequent changes of MDS masters may indicate poor network connectivity and may affect the cluster operation.
`The cluster is healthy with N active CS`	MDS info	Generated when the cluster status changes to *healthy* or when a new MDS master server is elected. This message indicates that all chunk servers in the cluster are active and the number of replicas meets the set cluster requirements.
`The cluster is degraded with N active, M inactive, K offline CS`	MDS warn	Generated when the cluster status changes to *degraded* or when a new MDS master server is elected. This message indicates that some chunk servers in the cluster are • inactive (do not send any registration messages) or • offline (are inactive for a period longer than `mds.wd.offline_tout = 5min` (by default)).
`The cluster failed with N active, M inactive, K offline CS (mds.wd.max_offline_cs=n)`	MDS err	Generated when the cluster status changes to *failed* or when a new MDS master server is elected. This message indicates that the number of offline chunk servers exceeds `mds.wd.max_offline_cs` (2 by default). When the cluster fails, the

		automatic replication is not scheduled any more. So the cluster administrator must take some actions to either repair failed chunk servers or increase the `mds.wd.max_offline_cs` parameter. Setting the value of this parameter to 0 disables the failed mode completely.
`The cluster is filled up to N%`	MDS info/warn	Shows the current space usage in the cluster. A warning is generated if the disk space consumption equals or exceeds 80%. It is important to have spare disk space for data replicas if one of the chunk servers fails.
`Replication started, N chunks are queued`	MDS info	Generated when the cluster starts the automatic data replication to recover the missing replicas.
`Replication completed`	MDS info	Generated when the cluster finishes the automatic data replication.
`CS#N has reported hard error on 'path'`	MDS warn	Generated when the chunk server `CS#N` detects disk data corruption. You are recommended to replace chunk servers with corrupted disks as soon as possible with new ones and to check the hardware for errors.
`CS#N has not registered during the last T sec and is marked as inactive/offline`	MDS warn	Generated when the chunk server `CS#N` has been unavailable for a while. In this case, the chunk server first gets marked as inactive. After 5 minutes, the state is changed to offline, which starts the automatic replication of data to restore the replicas that were stored on the offline chunk server.
`Failed to allocate N replicas for 'path' by request from <addr:port> - K out of M chunks servers are available`	MDS warn	Generated when the cluster cannot allocate chunk replicas, for example, when it runs out of disk space.
`Failed to allocate N replicas for 'path' by request from <addr:port> since only K chunk servers are registered`	MDS warn	Generated when the cluster cannot allocate chunk replicas because not enough chunk servers are registered in the cluster.

Monitoring the Status of Replication Parameters

When you configure replication parameters, keep in mind that the new settings do not come into effect immediately. For example, increasing the default replication parameter for data chunks may take some time to complete, depending on the new value of this parameter and the number of data chunks in the cluster.

To check that the new replication parameters have been successfully applied to your cluster:

1. Run the `pstorage -c <cluster_name> top` command.
2. Press the V key on your keyboard to display additional information about the cluster. Your command output should look similar to the following:

```
# pstorage -c stor1 top
connected to MDS#1
Cluster 'stor1': healthy
Space: [OK] allocatable 200GB of 211GB, free 211GB of 211GB
```

```
MDS nodes: 1 of 1, epoch uptime: 2h 21m
CS nodes:  2 of 2 (2 avail, 0 inactive, 0 offline)
License: PCSS.02444715.0000 is ACTIVE, 6399TB capacity
Replication:  3 norm,  2 limit
Chunks: [OK] 431 (100%) healthy,  0 (0%) degraded,  0 (0%) urgent,
             0 (0%) blocked,  0 (0%) offline,  0 (0%) replicating,
             0 (0%) overcommitted,  0 (0%) deleting,  0 (0%) void
...
```

3 Check the **Chunks** field for the following:

- When decreasing the replication parameters, look for chunks that are in the **overcommitted** or **deleting** state. If the replication process is complete, no chunks with these states should be present in the output.
- When increasing the replication parameters, look for chunks that are in the blocked or urgent state. If the replication process is complete, no chunks with these states should be present in the output. Besides, when the process is still in progress, the value of the healthy parameter is less than 100%.

Note: For more information on available chunk statutes, see **Exploring Chunk States** (p. 74).

CHAPTER 6

Managing Cluster Security

This chapter describes some situations that may affect your cluster security.

In This Chapter

Security Considerations

This section describes the security limitations you should keep in mind when deploying a Virtuozzo Storage cluster.

Traffic sniffing

Virtuozzo Storage does not protect you from traffic sniffing. Anyone who has access to your network can capture and analyze the data being sent and received through your network.

To learn how to keep your data secure, see **Securing Server Communication in Clusters** (p. 82).

Absence of users and groups

Virtuozzo Storage does not use the concept of users and groups, providing specific users and groups with access to specific parts of a cluster. Anyone authorized to access a cluster can access all its data.

Non-encrypted data on disks

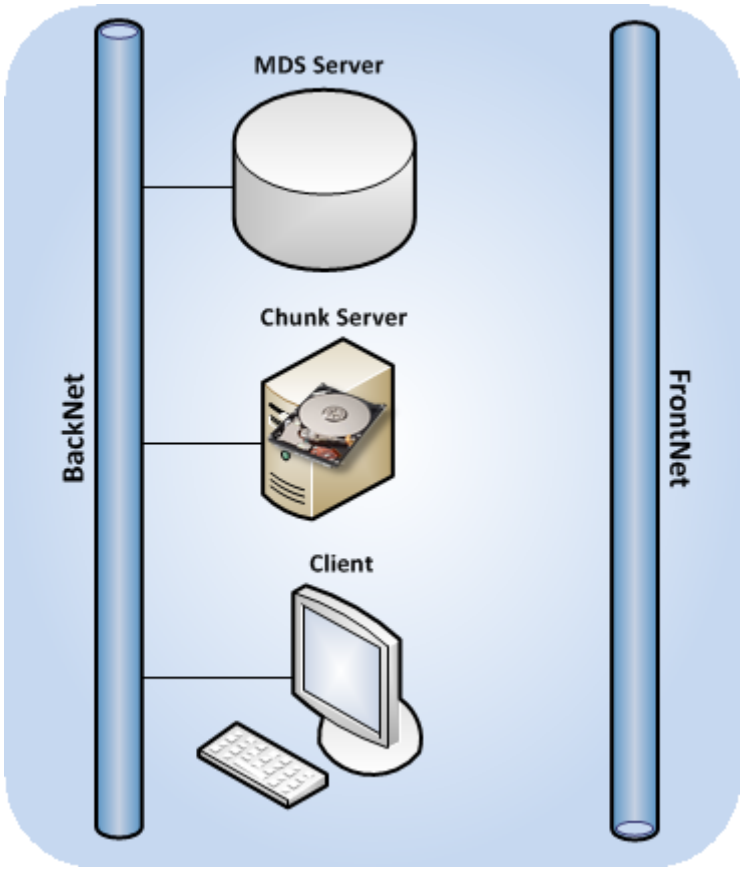
Virtuozzo Storage does not encrypt data stored in a cluster. Attackers can immediately see the data once they gain access to a physical disk drive.

Securing Server Communication in Clusters

A Virtuozzo Storage cluster can contain three types of servers:

- MDS servers
- chunk servers
- clients

During cluster operation, the servers communicate with each other. To secure their communication, you should keep all servers on an isolated private network—BackNet. The figure below shows an example cluster configuration where all servers are set up on the BackNet.



The process of deploying such a configuration can be described as follows:

1 You create the cluster by making the MDS server and specifying one of its IP addresses:

```
# pstorage -c Cluster-Name make-mds -I -a MDS-IP-Address -r Journal-Directory -p
```

The specified address will then be used for MDS interconnection and intercommunication with the other servers in the cluster.

2 You set up a chunk server:

```
# pstorage -c Cluster-Name make-cs -r CS-Directory
```

Once it is created, the chunk server connects to the MDS server and binds to the IP address it uses to establish the connection. If the chunk server has several networks cards, you can explicitly assign the chunk server to the IP address of a specific network card so that all communication between the chunk and MDS servers is carried out via this IP address.

To bind a chunk server to a custom IP address, you pass the `-a` option to the `pstorage make-cs` command when you create the chunk server:

```
# pstorage make-cs -r CS-Directory -a Custom-IP-Address
```

Note: A custom IP address must belong to the BackNet not to compromise your cluster security.

3 You mount the cluster on the client:

```
# pstorage-mount -c Cluster-Name Mount-Directory
```

Once the cluster is mounted, the client connects to the MDS and chunk server IP addresses.

This example configuration provides a high level of security for server communication because the MDS server, the chunk server, and the client are located on the isolated BackNet and cannot be compromised.

Cluster Discovery Methods

When choosing a cluster discovery method for your cluster, pay attention to the following:

- The recommended way of configuring auto-discovery for a Virtuozzo Storage cluster is to use DNS records. For more details on this discovery method, see **Using DNS Records** (p. 16).
- Zeroconf autodiscovery can be used for testing and evaluating the Virtuozzo Storage functionality. This discovery method is not recommended for use in production for security reasons. Malicious users can potentially perform DoS attacks on the Zeroconf service even if you use security certificates for authentication in your network.

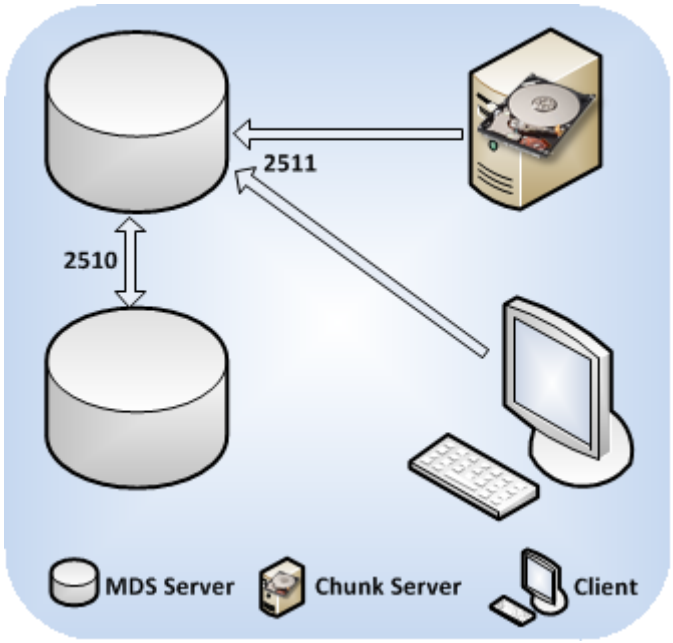
Virtuozzo Storage Ports

This section lists the ports that must be open on servers that participate in a Virtuozzo Storage cluster, in addition to the ports used by Virtuozzo and Virtuozzo Automator.

MDS Servers

The following ports must be open on an MDS server:

- **Listening ports**: 2510 for incoming connections from other MDS serves and 2511 for incoming connections from chunk servers and clients
- **Outbound ports**: 2510 for outgoing connections to other MDS servers



By default, Virtuozzo Storage uses port 2510 for communication between MDS servers and port 2511 for incoming connections from both chunk servers and clients. You can override the default ports when creating MDS servers:

1. Reserve two unoccupied consecutive ports.

   The ports must be the same on all MDS servers you plan to set up in the cluster.

2. Execute the `pstorage make-mds` command to create the MDS server and specify the lower port after the server IP address.

For example, to set up ports 4110 and 4111 for MDS communication in the `stor1` cluster, you can run this command:

```
# pstorage -c stor1 make-mds -I -a 10.30.100.101:4110 -r /pstorage/stor1-mds -p
```

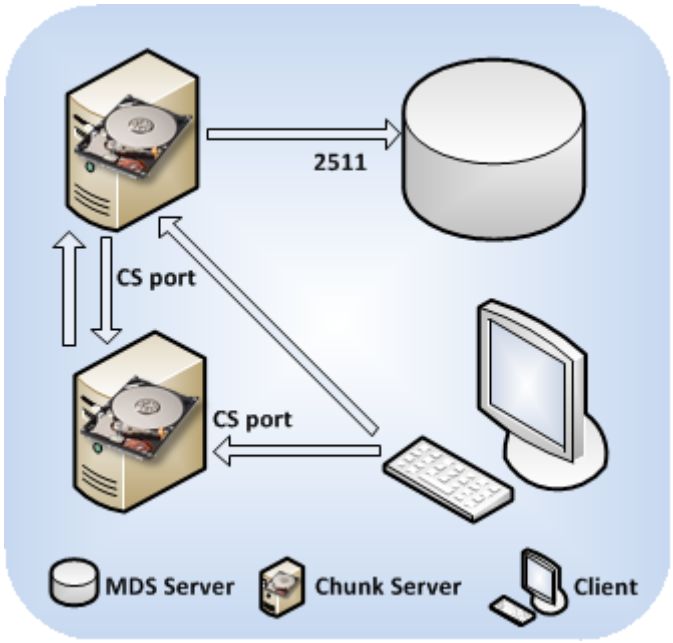
If you choose to use custom ports 4110 and 4111, do the following:

- On each MDS server with custom ports, open ports 4110 and 4111 for inbound traffic and port 4110 for outbound traffic.
- On all chunk servers and clients in the cluster, open port 4111 for outbound traffic.

Chunk Servers

The following ports must be open on a chunk server:

- **Listening ports**: a randomly chosen port for incoming connections from clients and other chunk servers
- **Outbound ports**: 2511 for outgoing connections to MDS servers and a randomly chosen port for outgoing connections to other chunk servers



The chunk server management service requires

- A port to communicate with MDS servers (either the default port 2511 or your custom port).
- A port to communicate with chunk servers and clients.

  By default, the service binds to any available port. You can manually redefine the port by passing the `-a` option to the `pstorage make-cs` command when creating a chunk server. For example, to configure the management service to use port 3000, run this command:

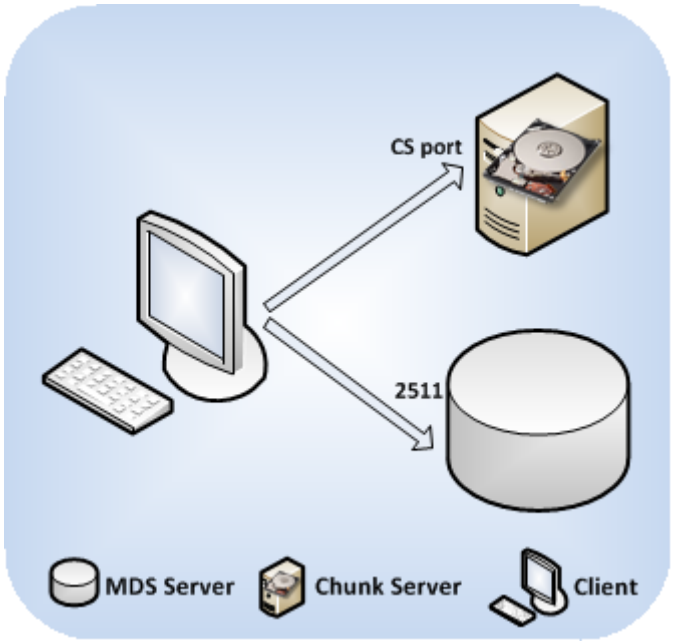
```
# pstorage make-cs -r /pstorage/stor1-cs -a 132.132.1.101:3000
```

  Once you set a custom port, open this port for outbound traffic on all clients and other chunk servers in the cluster.

Clients

The following ports must be open on a client:

- **Listening ports**: no
- **Outbound ports**: 2511 for outgoing connections to MDS servers and a port configured as the listening one on chunk servers



By default, Virtuozzo Storage automatically opens on a client the following outbound ports:

- For communication with MDS servers, the default port 2511.
- For communication with chunk servers, the port that was configured as the listening one on your chunk servers.

If you specify custom ports for MDS and chunk server communication, open these ports on the client for outgoing traffic. For example, if you configure port 4111 on MDS servers and port 3000 on chunk servers for communication with clients, open outbound ports 2511 and 3000 on the client.

Password-based Authentication

Virtuozzo Storage uses password-based authentication to enhance security in clusters. Password authentication is mandatory meaning that you have to pass the authentication phase before you can add a new server to the cluster.

Password-based authentication works as follows:

1. You set the authentication password when you create the first MDS server in the cluster. The password you specify is encrypted and saved into the `/etc/pstorage/clusters/stor1/auth_digest.key` file on the server.
2. You add new MDS servers, chunk servers, or clients to the cluster and use the `pstorage auth-node` command to authenticate them. During authentication, you use the password you set when creating the first MDS server.

3 Virtuozzo Storage compares the provided password with the one stored on the first MDS server, and if the passwords match, successfully authenticates the server.

For each physical server, authentication is a one-time process. Once a server is authenticated in the cluster (for example, when you configure it as an MDS server), the `/etc/pstorage/clusters/stor1/auth_digest.key` file is created on the authenticated server. When you set up this server as another cluster component (e.g., as a chunk server), the cluster checks that the `auth_digest.key` file is present and does not require you to authenticate the server again.

Installations via PXE Servers

Kickstart files you use to perform an unattended installation of Virtuozzo 6 and Virtuozzo Storage over a network contain a cluster authentication password in plain text. Attackers may intercept the password and gain access to your cluster. To secure your system, do one of the following:

- Physically isolate the network where the PXE server is located from all other (potentially insecure) networks.
- Install Virtuozzo 6 via the PXE server but do not set up a Virtuozzo Storage cluster. Once the installation is complete, manually deploy the cluster in your network using the Virtuozzo Storage password-based authentication mechanism.

CHAPTER 7

Maximizing Cluster Performance

This chapter describes recommended configurations for Virtuozzo Storage clusters and ways to tune them for maximum performance.

Notes:

1. Please also see **Troubleshooting** (p. 102) for common issues that might affect your cluster performance.

2. You may also consider updating Hardware Nodes in the cluster as described in the *Virtuozzo 6 User's Guide*.

In This Chapter

Exploring Possible Disk Drive Configurations

If the servers you plan to include in a Virtuozzo Storage cluster have several disk drives, you can choose one of the following configurations for the cluster:

1. **(Recommended with SSDs) No local RAIDs**

   A chunk server is set up per each hard drive and each data chunk has two or more replicas. This configuration provides independence of disk failures, and the cluster replication adds reliability close to RAID 1 (mirroring without parity or striping). Solid-state drives are highly recommended for CS journaling, improving write commit latency (e.g., for databases).

2. **(Recommended) Passthrough individual drives attached to a hardware RAID controller (without RAID levels 0/1/10/5/6), with or without SSDs**

   This highly recommended configuration is similar to the previous but is much faster. It features individual disks attached to a hardware RAID controller with a memory cache and a backup battery unit (BBU). RAID's write-back cache greatly improves random I/O write operations as well as database performance. The use of solid-state drives will further optimize random I/O, especially read operations.

3 **(Not recommended) Local striping RAID 0, data chunks with two or more replicas, with or without SSDs**

This configuration offers lower cluster reliability, because a single drive failure will cause the entire RAID 0 to fail, forcing the cluster to replicate more data each time such a failure occurs. Yet this can be considered a minor issue as Virtuozzo Storage clusters perform parallel recovery from multiple servers, which is much faster than rebuilding a traditional RAID 1.

The use of solid-state drives for caching and journaling will further improve overall cluster performance and provide data checksumming and scrubbing to improve cluster reliability.

4 **(Not recommended) Local mirror RAID 1, data chunks with one or more replicas**

When deployed with one replica per each data chunk, this configuration does not provide high availability for your cluster if some of your servers fail. Besides, such configurations do no bring any disk space savings as they are equivalent to cluster mirroring, and a local RAID just doubles the data replication factor and saves cluster network traffic.

5 **(Highly not recommended) Local RAID 1, 5, or 6, data chunks with two or more replicas**

Avoid running Virtuozzo Storage on redundant types of RAID like 1, 5, or 6 over local storage. In this case, a single write operation may affect a significant number of HDDs resulting in very poor performance. For example, for 3 Virtuozzo Storage replicas and RAID5 on servers with 5 HDDs each, a single write operation may result in 15 I/O operations.

Note: For information on recommended SSD drives, see **Using SSD drives** (p. 93).

Carrying Out Performance Benchmarking

When testing the performance of a Virtuozzo Storage cluster and comparing it with non-Virtuozzo Storage setups:

- Compare configurations with similar redundancy levels. For example, it is incorrect to compare the performance of a cluster with two or three replicas per data chunk with a standalone server that does not use any data redundancy, like RAID 1, 10, or 5.
- Take into account the usage of file system interfaces. Keep in mind that mounting a Virtuozzo Storage cluster using the FUSE interface provides a convenient view into the cluster but is not optimal for performance. Therefore, do benchmarks from inside Containers and virtual machines.
- Keep in mind that the data replication factor affects the cluster performance: clusters with two replicas are slightly faster than those with three replicas.

Using 1 GbE and 10 GbE Networks

1 Gbit/s Ethernet networks can deliver 110-120 MB/s, which is close to a single drive performance on sequential I/O. Since several drives on a single server can deliver higher throughput than a single 1 Gbit/s Ethernet link, networking may become a bottleneck.

However, in real-life applications and virtualized environments, sequential I/O is not common (backups mainly) and most of the I/O operations are random. Thus, typical HDD throughput is usually much lower, close to 10-20 MB/s, according to statistics accumulated from hundreds of servers by a number of major hosting companies.

Based on these two observations, we recommend to use one of the following network configurations (or better):

- A 1 Gbit/s link per each 2 HDDs on the Hardware Node. Although if you have 1 or 2 HDDs on a Hardware Node, two bonded network adapters are still recommended for better reliability (see **Setting Up Network Bonding** (p. 92)).
- A 10 Gbit/s link per Hardware Node for the maximum performance.

The table below illustrates how these recommendations may apply to a Hardware Node with 1 to 6 HDDs:

HDDs	1 GbE Links	10 GbE Links
1	1 (2 for HA)	1 (2 for HA)
2	1 (2 for HA)	1 (2 for HA)
3	2	1 (2 for HA)
4	2	1 (2 for HA)
5	3	1 (2 for HA)
6	3	1 (2 for HA)

Notes:

1. For the maximum sequential I/O performance, we recommend to use one 1Gbit/s link per each hard drive, or one 10Gbit/s link per Hardware Node.

2. It is not recommended to configure 1 Gbit/s network adapters to use non-default MTUs (e.g., 9000-byte jumbo frames). Such settings require switch configuration and often lead to human errors. 10 Gbit/s network adapters, on the other hand, need to be configured to use jumbo frames to achieve full performance.

3. For maximum efficiency, use the `balance-xor` bonding mode with the `layer3+4` hash policy. If you want to use the `802.3ad` bonding mode, also configure your switch to use the `layer3+4` hash policy.

Setting Up Network Bonding

Bonding multiple network interfaces together provides the following benefits:

1. High network availability. If one of the interfaces fails, the traffic will be automatically routed to the working interface(s).
2. Higher network performance. For example, two Gigabit interfaces bonded together will deliver about 1.7 Gbit/s or 200 MB/s throughput. The required number of bonded storage network interfaces may depend on how many storage drives are on the Hardware Node. For example, a rotational HDD can deliver up to 1 Gbit/s throughput.

To configure a bonding interface, do the following:

1. Create the `/etc/modprobe.d/bonding.conf` file containing the following line:

```
alias bond0 bonding
```

2. Create the `/etc/sysconfig/network-scripts/ifcfg-bond0` file containing the following lines:

```
DEVICE=bond0
ONBOOT=yes
BOOTPROTO=none
IPV6INIT=no
USERCTL=no
BONDING_OPTS="mode=balance-xor xmit_hash_policy=layer3+4 miimon=300 downdelay=300
updelay=300"
NAME="Storage net0"
NM_CONTROLLED=no
IPADDR=xxx.xxx.xxx.xxx
PREFIX=24
```

> **Notes:**
>
> 1. Make sure to enter the correct values in the `IPADDR` and `PREFIX` lines.
>
> 2. The `balance-xor` mode is recommended, because it offers both fault tolerance and better performance. For more details, see the documents listed below.

3. Make sure the configuration file of each Ethernet interface you want to bond (e.g., `/etc/sysconfig/network-scripts/ifcfg-eth0`) contains the lines shown in this example:

```
DEVICE="eth0"
BOOTPROTO=none
NM_CONTROLLED="no"
ONBOOT="yes"
TYPE="Ethernet"
HWADDR=xx:xx:xx:xx:xx:xx
MASTER=bond0
SLAVE=yes
USERCTL=no
```

4. Bring up the `bond0` interface:

```
# ifup bond0
```

5 Use `dmesg` output to verify that `bond0` and its slave Ethernet interfaces are up and links are ready.

Note: More information on network bonding is provided in the *Red Hat Enterprise Linux Deployment Guide* and *Linux Ethernet Bonding Driver HOWTO.*

Using SSD Drives

Virtuozzo Storage supports SSD drives formatted to the ext4 filesystem and optionally mounted with TRIM support enabled.

Note: Virtuozzo Storage SSD usage scenario does not generate TRIM commands. Also, modern drives like Intel SSD DC S3700 do not need TRIM at all.

Along with using SSD drives for storing data chunks, Virtuozzo Storage also supports the use of such drives for special purposes:

- **write journaling** (p. 94). You can attach an SSD drive to a chunk server and configure the drive to store a write journal. By doing so, you can boost the performance of write operations in the cluster by up to 2 and more times.
- **data caching** (p. 97). You can attach an SSD drive to a client and configure the drive to store a local cache of frequently accessed data. By having a local cache on a client's SSD drive, you can increase the overall cluster performance by up to 10 and more times.

Solid-state drive space should be split between write journals and read cache based on the nature of workload, i.e. the proportion of write operations to read operations. The read cache size also depends on how demanding your applications are. If you are unsure, split SSD space in equal parts. For example, if you have a 100GB SSD and four chunk servers on four 1TB HDDs, divide the SSD space as follows:

- 20 GB reserved for checksums and emergency needs and also to prevent the SSD from filling up completely (because its performance would then degrade),
- 40 GB for read cache,
- 40 GB for write journals, i.e. 10 GB per HDD/chunk server.

Checksums require 4 bytes of space for each 4KB page (1:1000 ratio). For example, 4 TB of storage will require 4 GB of space for checksums.

To understand how much space should be allocated for write journals and read cache in a specific cluster configuration, run the `pstorage advise-configuration` command. Using cluster parameters as input data, the command will output suggestions on how to optimize the cluster performance, set up the host, and mount the cluster via `/etc/fstab`. (see the following sections for examples).

In general, optimization for write journals means that about 70% of SSD space should be used for said journals and about 30% for read cache; Optimization for reads means the opposite proportion.

If optimization for either writes or reads is not the goal, the idea is to use equal parts of SSD space for write journals and read cache. In each case, some space should be reserved for checksums and such (the amount to reserve is also suggested by the `pstorage advise-configuration` command).

Finally, the table below will help you understand how many SSDs you will need for your cluster.

SSD Type	Number of SSDs
Intel SSD 320 Series, Intel SSD 710 Series, Kingston SSDNow E enterprise series, or other SATA 3Gbps SSD models providing 150-200MB/s of sequential write of random data.	1 SSD per up to 3 HDDs
Intel SSD DC S3700 Series, Samsung SM1625 enterprise series, or other SATA 6Gbps SSD models providing at least 300MB/sec of sequential write of random data.	1 SSD per up to 5-6 HDDs

The following sections provide detailed information on configuring SSD drives for write journaling and data caching.

Notes:

1. Not all solid-state drives obey flush semantics and commit data in accordance with the protocol. This may result in arbitrary data loss or corruption in case of a power failure. Always check your SSDs with the `pstorage-hwflush-check` tool (for more information, see **Checking Data Flushing** (p. 19)).

2. We recommend using Intel SSD DC S3700 drives. However, you can also use Samsung SM1625, Intel SSD 710, Kingston SSDNow E or any other SSD drive with support for data protection on power loss. Some of the names of this technology are: Enhanced Power Loss Data Protection (Intel), Cache Power Protection (Samsung), Power-Failure Support (Kingston), Complete Power Fail Protection (OCZ). For more information, see **SSD Drives Ignore Disk Flushes** (p. 105).

3. The minimal recommended SSD size is 30 GB.

Configuring SSD Drives for Write Journaling

Using SSD drives for write journaling can help you reduce write latencies, thus improving the overall cluster performance.

To determine how much SSD space you will need for CS journals, use the `pstorage advise-configuration` command with the `-w` option. For example:

```
# pstorage -c stor1 advise-configuration -w --cs /pstorage/stor1-cs1 --cs
/pstorage/stor1-cs2 --cs /pstorage/stor1-cs3 --cs /pstorage/stor1-cs4 --ssd
/pstorage/stor1-ssd -m /pstorage/stor1
You have the following setup:
CS  on /pstorage/stor1-cs4 -- Total disk space 1007.3GB
CS  on /pstorage/stor1-cs3 -- Total disk space 1007.3GB
CS  on /pstorage/stor1-cs2 -- Total disk space 1007.3GB
CS  on /pstorage/stor1-cs1 -- Total disk space 1007.3GB
SSD on /pstorage/stor1-ssd -- Total disk space 251.8GB
Proposed server configuration optimized for writes:
```

```
 -  155.9GB (61%) for CS journals, 66.8GB (26%) for mount read cache on
/pstorage/stor1-ssd,
 29.1GB (11%) reserved (including  3.9GB checksums for 3.9TB of data)
 - CS journal sizes:
         38.9GB for /pstorage/stor1-cs4 at /pstorage/stor1-ssd
         38.9GB for /pstorage/stor1-cs3 at /pstorage/stor1-ssd
         38.9GB for /pstorage/stor1-cs2 at /pstorage/stor1-ssd
         38.9GB for /pstorage/stor1-cs1 at /pstorage/stor1-ssd
How to setup the node:
pstorage -c stor1 make-cs -r /pstorage/stor1-cs4/cs -j /pstorage/stor1-ssd/cs4-stor1-
journal -s 39914
pstorage -c stor1 make-cs -r /pstorage/stor1-cs3/cs -j /pstorage/stor1-ssd/cs3-stor1-
journal -s 39914
pstorage -c stor1 make-cs -r /pstorage/stor1-cs2/cs -j /pstorage/stor1-ssd/cs2-stor1-
journal -s 39914
pstorage -c stor1 make-cs -r /pstorage/stor1-cs1/cs -j /pstorage/stor1-ssd/cs1-stor1-
journal -s 39914
pstorage-mount -c stor1 /pstorage/stor1 -C /pstorage/stor1-ssd/pstorage-stor1-cache -R
68424
Mount option for automatic cluster mount from /etc/fstab:
pstorage://stor1 /pstorage/stor1 fuse.pstorage cache=/pstorage/stor1-ssd/pstorage-
stor1-cache,cachesize=68424 0 0
```

In this example, the suggestion is to allocate 61% of SSD space for CS journals to achieve optimal cluster performance.

Note:

1. If you have multiple chunk servers on a single host, create a separate SSD journal for each CS, making sure that the SSD has enough space for all CS journals.To modify the size of existing CS journals use the `pstorage configure-cs` command.

2. When deciding on a journal size without using the `pstorage advise-configuration` command, make sure there is 1GB of SSD space per each 1TB of HDD space for checksums.

Setting Up a Chunk Server with a Journal on SSD

To set up a chunk server that stores a journal on an SSD drive, do the following:

1. Log in to the Node you want to act as a chunk server as root or as a user with root privileges. The Node must have at least one hard disk drive (HDD) and one solid state drive (SSD).
2. Download and install the following RPM packages: `pstorage-ctl`, `pstorage-libs-shared`, and `pstorage-chunk-server`.

   These packages are available in the Virtuozzo remote repository (this repository is automatically configured for your system when you install Virtuozzo) and can be installed with this command:

```
# yum install pstorage-chunk-server
```

3. Make sure that cluster discovery is configured for the server. For details, see **Configuring Cluster Discovery** (p. 16).
4. Authenticate the server in the cluster, if it is not yet authenticated:

```
# pstorage -c stor1 auth-node
```

5. If required, prepare the SSD as described in **Preparing Disks for Virtuozzo Storage** (p. 20).

6 Create the chunk server configuration, repository, and the journal, for example:

```
# pstorage -c stor1 make-cs -r /pstorage/stor1-cs -j /ssd/stor1/cs1 -s 30720
```

This command

- Makes the `/pstorage/stor1-cs` directory on your computer's hard disk drive and configures it for storing data chunks.
- Configures your computer as a chunk server and joins it to the `stor1` Virtuozzo Storage cluster.
- Creates the journal in the `/ssd/stor1/cs1` directory on the SSD drive and allocates 30 GB of disk space to this journal.

> **Note:** When choosing a directory for the journal and deciding on its size, allocate the required space for the journal and make sure there is 1GB of SSD space per each 1TB of HDD space for checksums.

7 Start the chunk server management service (`pstorage-csd`) and configure it to start automatically on the chunk server boot:

```
# service pstorage-csd start
# chkconfig pstorage-csd on
```

Adding, Destroying, and Configuring Chunk Server Journals in Live Virtuozzo Storage Clusters

> **Note:** To obtain CS repository paths, use the `pstorage list-services -C` command.

Adding Chunk Server Journals

To add a new journal to a chunk server, use the `pstorage configure-cs -a -s` command. For example, to add a 2048MB journal to the chunk server CS#1 and place it in a directory on an SSD drive mounted to `/ssd`, run:

```
# pstorage -c stor1 configure-cs -r /pstorage/stor1-cs1/data -a /ssd/stor1-cs1-journal
-s 2048
```

Destroying Chunk Server Journals

To destroy a chunk server journal, use the `pstorage configure-cs -d` command. For example:

```
# pstorage -c stor1 configure-cs -r /pstorage/stor1-cs1/data -d
```

Moving Chunk Server Journals

To change the chunk server journal directory, do the following using the commands above:

1 Destroy the existing journal

2 Add a new journal with the required size at the required location.

Resizing Chunk Server Journals

To resize a chunk server journal, use the `pstorage configure-cs -s` command. For example, to resize a CS journal to 4096MB:

```
# pstorage -c stor_1 configure-cs -r /pstorage/stor_1-cs1/data -s 4096
```

Disabling Checksumming

Using checksumming, you can provide better reliability and integrity of all data in the cluster. When checksumming is enabled, Virtuozzo Storage generates checksums each time some data in the cluster is modified. When this data is then read, the checksum is computed once more and compared with the already existing value.

By default, data checksumming is automatically enabled for newly created chunk servers that use journaling. If necessary, you can disable this functionality using the `-S` option when you set up a chunk server, for example:

```
# pstorage -c stor1 make-cs -r /pstorage/stor1-cs -j /ssd/stor1/cs1 -s 30720 -S
```

Configuring Data Scrubbing

Data scrubbing is the process of checking data chunks for durability and verifying their contents for readability and correctness. By default, Virtuozzo Storage is set to examine two data chunks per minute on each chunk server in the cluster. If necessary, you can configure this number using the `pstorage` utility, for example:

```
# pstorage -c stor1 set-config mds.wd.verify_chunks=3
```

This command sets the number of chunks to be examined on each chunk server in the `stor1` cluster to 3.

Configuring SSD Drives for Data Caching

Another way of improving the overall cluster performance is to create a local cache on a client's SSD drive. Once you create the cache, all cluster data accessed two or more times will be put to that cache.

The table below lists the main features specific to a local cache:

Feature	Description
Quick access time	Data in the local cache can be accessed much faster (up to 10 times and more) as compared to accessing the same data stored on chunk servers in the cluster.
No network bandwidth consumption	Cluster network bandwidth is not consumed because the data is accessed locally.
Special boot cache	Local cache uses a special boot cache to store small amounts of data on file openings. This significantly speeds up the process of starting virtual machines and Containers.

Cache survivability	Local cache is persistent and can survive a graceful system shutdown; however, it is dropped when the system crashes.
Sequential access filtering	Only randomly accessed data is cached. Data backup applications may generate a huge amount of sequential IO. Preventing such IO from being cached helps to avoid stressing the cache.

To determine how much SSD space you will need for the cache, use the `pstorage advise-configuration` command with the `-r` option. For example:

```
# pstorage -c stor1 advise-configuration -r --cs /pstorage/stor1-cs1 --cs
/pstorage/stor1-cs2 --cs /pstorage/stor1-cs3 --cs /pstorage/stor1-cs4 --ssd
/pstorage/stor1-ssd -m /pstorage/stor1
You have the following setup:
CS  on /pstorage/stor1-cs1 -- Total disk space 1007.3GB
CS  on /pstorage/stor1-cs2 -- Total disk space 1007.3GB
CS  on /pstorage/stor1-cs3 -- Total disk space 1007.3GB
CS  on /pstorage/stor1-cs4 -- Total disk space 1007.3GB
SSD on /pstorage/stor1-ssd -- Total disk space 251.8GB
Proposed server configuration optimized for reads:
 -  66.8GB (26%) for CS journals, 155.9GB (61%) for mount read cache on
/pstorage/stor1-ssd,
 29.1GB (11%) reserved (including  3.9GB checksums for 3.9TB of data)
 - CS journal sizes:
         16.7GB for /pstorage/stor1-cs4 at /pstorage/stor1-ssd
         16.7GB for /pstorage/stor1-cs3 at /pstorage/stor1-ssd
         16.7GB for /pstorage/stor1-cs2 at /pstorage/stor1-ssd
         16.7GB for /pstorage/stor1-cs1 at /pstorage/stor1-ssd
How to setup the node:
pstorage -c stor1 make-cs -r /pstorage/stor1-cs4/cs -j /pstorage/stor1-ssd/cs4-stor1-
journal -s 17106
pstorage -c stor1 make-cs -r /pstorage/stor1-cs3/cs -j /pstorage/stor1-ssd/cs3-stor1-
journal -s 17106
pstorage -c stor1 make-cs -r /pstorage/stor1-cs2/cs -j /pstorage/stor1-ssd/cs2-stor1-
journal -s 17106
pstorage -c stor1 make-cs -r /pstorage/stor1-cs1/cs -j /pstorage/stor1-ssd/cs1-stor1-
journal -s 17106
pstorage-mount -c stor1 /pstorage/stor1 -C /pstorage/stor1-ssd/pstorage-stor1-cache -R
159658
Mount option for automatic cluster mount from /etc/fstab:
pstorage://stor1 /pstorage/stor1 fuse.pstorage cache=/pstorage/stor1-ssd/pstorage-
stor1-cache,cachesize=159658 0 0
```

In this example, the suggestion is to allocate 61% of SSD space for the cache to achieve optimal cluster performance.

Creating a Local Cache

Note: Unlike directories used in most Virtuozzo Storage configuration steps, the local cache on SSD is a file. Make sure you supply correct paths to the `pstorage-mount -C` command and the `cache` parameter in the corresponding `/etc/fstab` entry.

You create a local cache when mounting a Virtuozzo Storage cluster to a client. This process includes two steps:

1 If required, preparing the SSD as described in **Preparing Disks for Virtuozzo Storage** (p. 20).

2 Using the `pstorage-mount` command to mount the cluster and create the cache.

For example, to make a 64 GB local cache for the `stor1` cluster and store it in the file `/mnt/ssd/pstorage-cache-for-cluster-stor1`, you can execute the following command:

```
# pstorage-mount -c stor1 /pstorage/stor1 -C /mnt/ssd/pstorage-cache-for-cluster-stor1
-R 64000
```

If you do not specify the cluster size, `pstorage-mount` will automatically calculate it using the following formula:

```
SSD_free_space - 10 GB - SSD_total_space/10
```

So if the total size of your SSD drive is 100 GB and it has 80 GB of free space, the command will create the local cache with the size of 60 GB.

Notes:

1. The local cache is not created if the resulting cache size is less than 1 GB.

2. If you also plan to configure the SSD drive for write journaling, first create the journal to reserve disk space for it and then create a local cache. For more information, see **Configuring SSD Drives for Write Journaling** (p. 94).

Configuring Automatic Cache Creation

You can automate the procedure of creating a local cache so that it is automatically created each time you start the client. To do this, add the information about the cache to the `/etc/fstab` file on the client.

For example, to (1) have an automatically created cache with the name of `pstorage-cache-for-cluster-stor1` and size of 64 GB, (2) store it in the `/mnt/ssd` directory on the client, and (3) disable checksumming for data in the local cache, specify the following parameters in `/etc/fstab` and separate them by commas:

- **cache=*N***. Sets the full path to the local cache file.
- **cachesize=*N***. Specifies the size of the local cache, in megabytes.
- **cachechksum=n**. Disables checksumming for your data; by default, data checksumming is enabled.

Once you set these parameters, your `fstab` file should look like the following:

```
pstorage://stor1 /pstorage/stor1 fuse.pstorage cache=/mnt/ssd/pstorage-cache-for-
cluster-stor1,cachesize=64000,cachechksum=n 0 0
```

For more information on options you can use to set up and configure a local cache, see the `pstorage-mount` man pages.

Disabling Checksumming

To provide better reliability and integrity of your data, the `pstorage-mount` command automatically enables checksumming for the data in the local cache. If necessary, you can disable data checksumming by passing the `-S` option to `pstorage-mount`:

```
# pstorage-mount -c stor1 /pstorage/stor1 -C /mnt/ssd/pstorage-cache-for-cluster-stor1
-R 64000 -S
```

Querying Cache Information

To check whether the cache for a mounted cluster is active and view its current parameters, you can use this command:

```
# cat /pstorage/stor1/.pstorage.info/read_cache_info
path            : /mnt/ssd/pstorage-cache-for-cluster-stor1
main  size (Mb) : 56000
boot  size (Mb) : 8000
block size (Kb) : 64
checksum        : enabled
```

If the cache does not exist, the command output is empty. Otherwise, the command prints:

- path to the cache file
- size of the main and boot caches
- block size
- checksum status

Improving High-Capacity HDD Performance

Unlike older hard disks with 512-byte sectors, many modern HDDs (3TB and more in capacity) use 4KB physical sectors. In certain cases, this can greatly reduce system performance (by 3-4 times) due to extra Read-Modify-Write (RMW) cycles required to align the source write request. Why this happens? When an operating system issues an unaligned write request, the HDD has to align the beginning and end of that request to 4KB boundaries. To do this, the HDD reads the request's head and tail ranges to determine an even number of sectors to modify. For example, on a request to write a 4KB block at a 2KB offset, HDD will read the 0-2KB and 6-8KB ranges to modify the entire 0-8KB data range.

The typical reasons of poor performance with 4KB sector HDDs are:

1. Host OS file system unaligned on the 4KB boundary. The `make-cs` command of Virtuozzo Storage tries to detect and report such issues to the administrator in advance, but be aware that the `fdisk` utility is not recommended for partitioning HDDs. You should use `parted` instead.
2. Unaligned writes (e.g., 1KB) performed by guest OS. Many legacy operating systems, like Microsoft Windows XP and Windows Server 2003 or Red Hat Enterprise Linux 5.x, have unaligned partitions by default and generate unaligned I/O patterns which are quite slow on

both Virtuozzo Storage and actual HDDs with 4KB sectors. If you plan running such legacy operating systems, consider the following:

- Using smaller HDDs with 512-byte sectors, or use SSD journaling for CS services which mitigates the issue to some extent.
- Aligning OS partitions properly as described in **Aligning Disks and Partitions in Virtual Machines** in the *Virtuozzo 6 User's Guide*.

You can check for unaligned write operations in the cluster by as follows:

1. Run the `pstorage top` or `stat` command. For example:
```
# pstorage -c stor1 top
```
2. Press **i** to display the **RMW** and **JRMW** columns in the CS part of the `top` output.
3. Check the **RMW** or **JRMW** counters, which are explained below.
   - When SSD journaling is used, the **RMW** counter shows the number of requests which lead to Read-Modify-Write cycles, while the **JRMW** counter shows the number of Read-Modify-Write cycles mitigated by the use of SSD journals.
   - When SSD journaling is not used, the **JRMW** counter shows the number of unaligned requests which potentially generate Read-Modify-Write cycles on the HDD in question.

Improving Virtual Disk Performance

This section lists ways of improving the performance of virtual disks.

Preventing Partition Misalignment in Legacy Operating Systems

Virtual disks in virtual machines running legacy guest operating systems such as Windows Server 2003, Windows XP, Windows 2000, CentOS 5, or RHEL 5 may work slower because of partition misalignment. For solutions to this issue, see **Aligning Disks and Partitions in Virtual Machines** in the *Virtuozzo 6 User's Guide*.

Maximizing Virtuozzo Containers Performance

Containers created with Virtuozzo Containers installed in a virtual machine may not work at maximum performance by default. To maximize their performance, make sure you install the update VZU460053 for Virtuozzo Containers before you create or migrate any Containers. For more information on working with Virtuozzo Containers, see the *Virtuozzo Containers 6.0 User's Guide*.

Note: A write journal on a solid-state drive is highly recommended for virtual machines running Virtuozzo Containers.

CHAPTER 8

Appendices

This chapter contains two appendices:

- **Appendix A** (p. 102) provides troubleshooting for most common problems in Virtuozzo Storage clusters.
- **Appendix B** (p. 107) gives answers to most frequently asked questions.

In This Chapter

Appendix A - Troubleshooting

This chapter describes common issues you may encounter when working with Virtuozzo Storage clusters and ways to resolve them. The main tool you use to solve cluster problems and detect hardware failures is `pstorage top`.

Submitting Problem Reports to Technical Support

If your cluster is experiencing a problem that you cannot solve on your own, you can use the `pstorage make-report` command to compile a detailed report about the cluster. You can then send the report to the support team who will closely examine your problem and make their best to solve it as quickly as possible.

To generate a report:

1. Configure passwordless SSH access for the `root` user from the server where you plan to run the `pstorage make-report` command to all servers that participate in the cluster.

   The easiest way to do this is to create an SSH key with `ssh-keygen` and use `ssh-copy-id` to configure all servers to trust this key. For details, see the man pages for `ssh-keygen` and `ssh-copy-id`.

2. Run the `pstorage make-report` command to compile the report:

```
# pstorage -c stor1 make-report
The report is created and saved to pstorage-report-20121023-90630.tgz
```

The command collects cluster-related information on all servers participating in the cluster and saves it to a file in your current working directory. You can find the exact file name by checking the `pstorage` output (`pstorage-report-20121023-90630.tgz` in the example above).

If necessary, you can save the report to a file with a custom name and put it to a custom location. To do this, pass the `-f` option to the command and specify the desired file name (with the `.tgz` extension) and location, for example:

```
# pstorage -c stor1 make-report -f /home/reportPCS1.tgz
```

Once it is ready, submit the report to the support team. For example, you can do this using the Support Request Tracker tool. For more information on using this tool, consult the *Virtuozzo 6 User's Guide*.

Note: The report contains only information related to your cluster settings and configuration. It does not contain any private information.

Out of Disk Space

When very little free disk space remains in a Virtuozzo Storage cluster, it is critically important to increase it as soon as possible by adding more chunk servers or removing some data. As soon as 95% of cluster disk space becomes occupied, the allocation of new data chunks is no longer possible and such requests are blocked until the cluster can satisfy the demand. As a result, user I/O becomes blocked as well, effectively freezing Containers and virtual machines.

Note: It is highly recommended to keep at least 10% of disk space free for recovery in case of host machine failures. You should also monitor usage history, for example, using the `pstorage top` or `pstorage get-event` commands (for more information, see **Monitoring Virtuozzo Storage Clusters** (p. 66)).

Symptoms

1. Stuck I/O or unresponsive mount point, `dmesg` messages about stuck I/O, frozen Containers and virtual machines.
2. `pstorage top` and `pstorage get-event` show error messages like "Failed to allocate X replicas at tier Y since only Z chunk servers are available for allocation".

Solutions

1. Remove any unnecessary data to free disk space.

   Note: Additional effect which may surprise at first is that as soon as I/O queues in the kernel are full with the blocked I/O, a mount point on the client machine may stuck responding altogether and no longer be able to service the requests even such as file listing. In this case an additional mount point can be created to list, access and remove the unneeded data.

2. Add more Chunk Servers on unused disks (see **Setting Up Chunk Servers** (p. 24)).

If the solutions above are not possible, you can use one of the following *temporary* workarounds:

1. Lower the replication factor for some of the least critical user data (see **Configuring Replication Parameters** (p. 36)). Remember to revert the changes afterwards.
2. Reduce the allocation reserve. For example, for cluster `stor1`:

```
# pstorage -c stor1 set-config mds.alloc.fill_margin=2
```

   where `mds.alloc.fill_margin` is the percentage of reserved disk space for CS operation needs (the default value is 5). Remember to revert the changes afterwards.

Poor Write Performance

Some network adapters, like RTL8111/8168B, are known to fail to deliver full-bandwidth, full-duplex network traffic. This can result in poor write performance.

So before deploying a Virtuozzo Storage cluster, you are highly recommended to test networks for full-duplex support. You can use the `netperf` utility to simultaneously generate in and out traffic. For example, in 1 GbE networks, it should constantly deliver about 2 Gbit/s of total traffic (1 Gbit/s for incoming and 1 Gbit/s for outgoing traffic).

Poor Disk I/O Performance

In most BIOS setups, AHCI mode is configured to work by default with the **Legacy** option enabled. With this option, your servers work with SATA devices via the legacy IDE mode, which may affect the cluster performance, making it up to 2 times slower than expected. You can check that the option is currently enabled by running the `hdparm` command, for example:

```
# hdparm -i /dev/sda
...
 PIO modes:  pio0 pio1 pio2 pio3 *pio4
 DMA modes:  mdma0 mdma1 mdma2
 UDMA modes: udma0 udma1 udma2 udma3 udma4 udma5 udma6
```

The asterisk before `pio4` in the **PIO modes** field indicates that your hard drive `/dev/sda` is currently operating in the legacy mode.

To solve this problem and maximize the cluster performance, always enable the **AHCI** option in your BIOS settings.

Hardware RAID Controller and Disk Write Caches

It is important that all hard disk drives obey the "flush" command and write their caches before the command completes. Unfortunately, not all hardware RAID controllers and drives do this, which may lead to data inconsistencies and file system corruptions on a power failure.

Some 3ware RAID controllers do not disable disk write caches and do not send "flush" commands to disk drives. As a result, the file system may sometimes become corrupted even though the RAID controller itself has a battery. To solve this problem, disable writes caches on all disk drives in the RAID.

Also, make sure that your configuration is thoroughly tested for consistency before deploying a Virtuozzo Storage cluster. For information on how you can do this, see **SSD Drives Ignore Disk Flushes** (p. 105).

SSD Drives Ignore Disk Flushes

A lot of desktop-grade SSD drives can ignore disk flushes and fool operating systems by reporting that data was written while it was actually not. Examples of such drives are OCZ Vertex 3 and Intel X25-E, X-25-M G2 that are known to be unsafe on data commits. Such disk drives should not be used with databases and may easily corrupt the file system on a power failure.

The 3rd generation Intel SSD drives (S3700 and 710 series) do not have these problems, featuring a capacitor to provide a battery backup for flushing the drive cache when the power goes out.

Use SSD drives with care and only when you are sure that drives are server-grade drives and obey "flush" rules. For more information on this issue, read the following article about PostreSQL: http://www.postgresql.org/docs/current/static/wal-reliability.html.

Cluster Cannot Create Enough Replicas

Sometimes, the cluster might not create the required number of data chunks even if enough chunk servers are present in the cluster.

This may be the case when you create new chunk servers by making copies of an existing chunk server (e.g., you set up a chunk server in a virtual machine and then clone this machine). In this case, all copied chunk servers have the same UUID — that is, the UUID of the original server. The cluster has information that all chunk servers are located on the original host and cannot allocate new data chunks.

To solve the problem, generate a new UUID for a cloned chunk server by running the following command on the destination host:

```
# /usr/bin/uuidgen -r | tr '-' ' ' | awk '{print $1$2$3}' > /etc/pstorage/host_id
```

For more information on the `uuidgen` utility, see its man page.

Failed Chunk Servers

If a chunk server in your Virtuozzo Storage cluster fails, you need to identify the cause of failure and choose a correct way to solve the problem.

Do the following:

1. Run the `pstorage top` command. For example:

```
# pstorage -c stor1 top
```

2. Press **i** to cycle to the FLAGS column in the chunk server section and find the flags corresponding to the failed CS.

3 Find the shown flags in the table below to identify the cause of failure and the way to solve the problem.

Flag	Issue	What to do
H	An I/O error. The disk on which the chunk server runs is broken.	Check the disk for errors. If the disk is broken, replace it and recreate the CS as described in **Replacing Disks used as Chunk Servers** (p. 106). Otherwise, contact technical support.
h	A chunk checksum mismatch. Either the chunk is corrupt or the disk where the chunk is stored is broken.	Check the disk for errors. If the disk is broken, replace it and recreate the CS as described in **Replacing Disks used as Chunk Servers** (p. 106). Otherwise, contact technical support.
S	The CS journal stored on a journaling SSD is not accessible. Either the journal is corrupt or the journaling SSD is broken.	Check the journaling SSD for errors. If the disk is broken, replace it as described in **Failed Write Journaling SSDs** (p. 107).
s	The chunks' checksums stored on a caching SSD are not accessible. Either the checksums are corrupt or the caching SSD is broken.	Check the caching SSD for errors. If the disk is broken, replace it as described in **Failed Data Caching SSDs** (p. 107).
R	The path to the chunk repository is invalid on CS start. The disk on which the chunk server runs is not attached or mounted.	Make sure the disk is attached and correctly mounted. Make sure the disk's entry in `/etc/fstab` is correct.
T	An I/O request timeout. The disk may only be inaccessible for some reason and not necessarily broken.	Make sure the disk is attached and check `dmesg` output for I/O request timeout messages to find out why the disk might be inaccessible.

Replacing Disks Used as Chunk Servers

To replace a broken HDD or SSD disk used as a chunk server, do the following:

1 Remove the failed CS from the cluster as described in **Removing Chunk Servers** (p. 32).

> **Note:** Do not detach the broken disk until you remove the CS.

2 Replace the broken disk with a new one.

If hotplugging is enabled, the new CS will be created automatically within a minute. Otherwise, do the following:

1 Prepare the SSD as described in **Preparing Disks for Virtuozzo Storage** (p. 20).

2 Create a new CS as described in **Creating Chunk Servers** (p. 25).

Failed Write Journaling SSDs

If the SSD used to store write journals breaks, all chunk servers which had journals on this SSD will fail. The cluster will continue to work and will create new replicas to make up for those which have been lost. If you need to set up same write journals on a replacement SSD, do the following:

1. Remove the failed chunk servers as described in **Removing Chunk Servers** (p. 32).
2. Prepare the SSD as described in **Preparing Disks for Virtuozzo Storage** (p. 20).
3. Create new chunk servers which will keep write journals on the new SSD as described in **Configuring SSD Drives for Write Journaling** (p. 94).

Failed Data Caching SSDs

If an SSD disk used to store read cache breaks, the cache will be lost and the client will continue to work without it. If you need to set up the same read cache on a replacement SSD, do the following:

1. Stop all virtual environments running on the client or migrate them to a different Node.
2. Unmount the cluster. For example:

```
# umount /pstorage/stor1
```

3. Prepare the SSD as described in **Preparing Disks for Virtuozzo Storage** (p. 20).
4. Create the cache and mount the cluster as you have had as described in **Configuring SSD Drives for Data Caching** (p. 97).

Failed MDS Servers

If the disk hosting an MDS server fails, replace it as follows:

1. Delete the failed MDS server as described in **Removing MDS Servers** (p. 31).
2. Create a new MDS server as described in **Adding MDS Servers** (p. 30).

Appendix B - Frequently Asked Questions

This Appendix lists most frequently asked questions about Virtuozzo Storage clusters.

General

Can /pstorage directory still be used on newer installations?

Yes. In newer installations, `/pstorage` remains as a symlink to the new `/pstorage` directory for compatibility purposes.

Do I need to buy additional storage hardware for Virtuozzo Storage?

No. Virtuozzo Storage eliminates the need for external storage devices typically used in SANs by converting locally attached storage from multiple nodes into a shared storage.

What are the hardware requirements for Virtuozzo Storage?

Virtuozzo Storage does not require any special hardware and can run on commodity computers with traditional SATA drives and 1 GbE networks. Some hard drives and RAID controllers, however, ignore the FLUSH command to imitate better performance and must not be used in clusters as this may lead to file system or journal corruptions. This is especially true for RAID controllers and SSD drives. Please consult with your hard drive's manual to make sure you use reliable hardware.

For more information, see **System Requirements** (p. 12), **Network Requirements** (p. 13), **Hardware RAID Controller and Disk Write Caches** (p. 104).

How many servers do I need to run a Virtuozzo Storage cluster?

You need only one physical server to start using Virtuozzo Storage. However, to provide high availability for your data, you are recommended to configure your cluster to have at least 3 replicas per each data chunk. This requires at least 3 online servers—and at least 5 servers in total—in to be set up in the cluster. For details, see **Virtuozzo Storage Configurations** (p. 9) and **Configuring Replication Parameters** (p. 36).

Can I join Hardware Nodes running different supported operating systems into a single Virtuozzo Storage cluster?

Yes. You can create Virtuozzo Storage clusters of Hardware Nodes running any combination of supported operating systems. For example, you can have metadata servers on Hardware Nodes with Ubuntu 14.04, chunk servers on Hardware Nodes with Red Hat Enterprise Linux 7, and clients on computers with CentOS 7.

Note: The current standalone version of Virtuozzo Storage does not support Virtuozzo.

Scalability and Performance

How many servers can I join to a Virtuozzo Storage cluster?

There is no strict limit on the number of servers you can include in a Virtuozzo Storage cluster. However, you are recommended to limit the servers in the cluster to a single rack to avoid any possible performance degradation due to inter-rack communications.

How much disk space can a Virtuozzo Storage cluster have?

A Virtuozzo Storage cluster can support up to 8 PB of effective available disk space, which totals to 24 PB of physical disk space when 3 replicas are kept for each data chunk.

Can I add nodes to an existing Virtuozzo Storage cluster?

Yes, you can dynamically add and remove nodes from a Virtuozzo Storage cluster to increase its capacity or to take nodes offline for maintenance. For more information, see **Adding and Replacing Chunk Servers** (p. 31).

What is the expected performance of a Virtuozzo Storage cluster?

The performance depends on the network speed and the hard disks used in the cluster. In general, the performance should be similar to locally attached storage or better. You can also use SSD caching to increase the performance of commodity hardware by adding SSD drives to the cluster for caching purposes. For more information, see **Using SSD Drives** (p. 93).

What performance should I expect on 1-gigabit Ethernet?

The maximum speed of a 1 GbE network is close to that of a single **rotational** drive. In most workloads, however, random I/O access is prevalent and the network is usually not a bottleneck. Research with large service providers has proved that average I/O performance rarely exceeds 20 MB/sec due to randomization. Virtualization itself introduces additional randomization as multiple independent environments perform I/O access simultaneously. Nevertheless, 10-gigabit Ethernet will often result in better performance and is recommended for use in production.

Will the overall cluster performance improve if I add new chunk servers to the cluster?

Yes. Since data is distributed among all hard drives in the cluster, applications performing random I/O experience an increase in IOPS when more drives are added to the cluster. Even a single **client machine** may get noticeable benefits by increasing the number of chunk servers and achieve performance far beyond traditional, locally attached storage.

Does performance depend on the number of chunk replicas?

Each additional replica degrades write performance by about 10%, but at the same time it may also improve read performance because the Virtuozzo Storage cluster has more options to select a faster server.

Availability

How does Virtuozzo Storage protect my data?

Virtuozzo Storage protects against data loss and temporary unavailability by creating data copies (replicas) and storing them on different servers. To provide additional reliability, you can configure Virtuozzo Storage to maintain user data checksums and verify them when necessary.

What happens when a disk is lost or a server becomes unavailable?

Virtuozzo Storage automatically recovers from a degraded state to the specified redundancy level by replicating data on live servers. Users can still access their data during the recovery process.

How fast does Virtuozzo Storage recover from a degraded state?

Since Virtuozzo Storage recovers from a degraded state using all the available hard disks in the cluster, the recovery process is much faster than for traditional, locally attached RAIDs. This makes the reliability of the storage system significantly better as the probability of losing the only remaining copy of data during the recovery period is very small.

Can I change redundancy settings on the fly?

Yes, at any point you can change the number of data copies, and Virtuozzo Storage will apply the new settings by creating new copies or removing unneeded ones. For more details on configuring replication parameters, see **Configuring Replication Parameters** (p. 36).

Do I still need to use local RAIDs?

No, Virtuozzo Storage provides the same built-in data redundancy as a mirror RAID1 array with multiple copies. However, for better sequential performance, you can use local stripping RAID0 exported to your Virtuozzo Storage cluster. For more information on using RAIDs, see **Possible Disk Drive Configurations** (p. 89).

Does Virtuozzo Storage have redundancy levels similar to RAID5?

No. To build a reliable software-based RAID5 system, you also need to use special hardware capabilities like backup power batteries. In the future, Virtuozzo Storage may be enhanced to provide RAID5-level redundancy for read-only data such as backups.

What is the recommended number of data copies?

It is recommended to configure Virtuozzo Storage to maintain 3 copies, which allows your cluster to survive the simultaneous loss of 2 hardware nodes (given that `failure-domain=host`).

Cluster Operation

How do I know that the new replication parameters have been successfully applied to the cluster?

To check whether the replication process is complete, run the `pstorage top` command, press the V key on your keyboard, and check information in the **Chunks** field:

- When decreasing the replication parameters, no chunks in the **overcommitted** or **deleting** state should be present in the output.
- When increasing the replication parameters, no chunks in the **blocked** or **urgent** state should be present in the output. Besides, the overall cluster health should equal 100%.

For details, see **Monitoring the Status of Replication Parameters** (p. 80).

How do I shut down a cluster?

To shut down a Virtuozzo Storage cluster:

1. Stop all clients.
2. Stop all MDS servers.
3. Stop all chunk servers.

For details, see **Shutting Down Virtuozzo Storage Clusters** (p. 46).

What tool do I use to monitor the status and health of a cluster?

You can monitor the status and health of your cluster using the `pstorage top` command. For details, see **Monitoring Virtuozzo Storage Clusters** (p. 66).

To view the total amount of disk space occupied by all user data in the cluster, run the `pstorage top` command, press the V key on your keyboard, and look for the **FS** field in the output. The **FS** field shows how much disk space is used by all user data in the cluster and in how many files these data are stored. For details, see **Understanding Disk Space Usage** (p. 71).

How do I configure a Virtuozzo server for a cluster?

To prepare a server with Virtuozzo for work in a cluster, you simply tell the server to store its Containers and virtual machines in the cluster rather than on its local disk. For details, see **Configuring virtual machines and Containers** (p. 27).

Why vmstat/top and pstorage stat show different IO times?

The `pstorage` and `vmstat`/`top` utilities use different methods to compute the percentage of CPU time spent waiting for disk IO (**wa%** in `top`, **wa** in `vmstat`, and **IOWAIT** in `pstorage`). The `vmstat` and `top` utilities mark an idle CPU as waiting only if an outstanding IO request is started on that CPU, while the `pstorage` utility marks all idle CPUs as waiting, regardless of the number of IO requests waiting for IO. As a result, `pstorage` can report much higher IO values. For example, on a system with 4 CPUs and one thread doing IO, `pstorage` will report over 90% IOWAIT time, while `vmstat` and `top` will show no more than 25% IO time.

What effect tier numbering has on Virtuozzo Storage operation?

When assigning storage to tiers, have in mind that faster storage drives should be assigned to higher tiers. For example, you can use tier 0 for backups and other cold data (CS without SSD journals), tier 1 for virtual environments—a lot of cold data but fast random writes (CS with SSD journals), tier 2 for hot data (CS on SSD), journals, caches, specific virtual machine disks, and such.

This recommendation is related to how Virtuozzo Storage works with storage space. If a storage tier runs out of free space, Virtuozzo Storage will attempt to temporarily use a lower tier. If you add more storage to the original tier later, the data, temporarily stored elsewhere, will be moved to the tier where it should have been stored originally.

For example, if you try to write data to the tier 2 and it is full, Virtuozzo Storage will attempt to write that data to tier 1, then to tier 0. If you add more storage to tier 2 later, the aforementioned data, now stored on the tier 1 or 0, will be moved back to the tier 2 where it was meant to be stored originally.

CHAPTER 9

Glossary

Client computer (or **client**). A computer from where you access a Virtuozzo Storage cluster and run virtual machines and Containers.

Chunk replicas (or **replicas**). Copies of data chunks stored on different chunk servers in a Virtuozzo Storage cluster.

Chunk server. A computer storing the contents of real data, such as virtual machine and Container disk images, in the form of fixed-size chunks and providing access to these data chunks.

Data chunk (or **chunk**). A fixed-size (e.g., 64 MB) piece of data residing in a Virtuozzo Storage cluster. For example, each virtual machine or Container disk image is split into such chunks, and the chunks are distributed across servers in the cluster.

Master metadata server (or **master MDS server**). The central MDS server responsible for monitoring all cluster activity and keeping all metadata current, along with acting as a regular MDS server. Any MDS server can become the master one if the current master MDS server fails.

Virtuozzo Storage cluster. A set of computers in a network where Virtuozzo Storage is deployed.

Replication. The process of storing data chunks on multiple storage devices to improve reliability, fault-tolerance, or accessibility of virtual machines and Containers.

Regular metadata server (or **regular metadata server** or **MDS server**). A computer storing metadata about Virtuozzo Storage chunks, their replicas and locations. Regular MDS servers periodically communicate with the master MDS server to get metadata updates.

Index

T

U

V

W