Acronis Software-Defined Infrastructure 2.5

Administrator's Command Line Guide

March 26, 2019

Copyright Statement

Acronis patented technologies

Technologies, used in this product, are covered and protected by one or more U.S. Patent Numbers: 7,047,380; 7,246,211; 7,275,139; 7,281,104; 7,318,135; 7,353,355; 7,366,859; 7,383,327; 7,475,282; 7,603,533; 7,636,824; 7,650,473; 7,721,138; 7,779,221; 7,831,789; 7,836,053; 7,886,120; 7,895,403; 7,934,064; 7,937,612; 7,941,510; 7,949,635; 7,953,948; 7,979,690; 8,005,797; 8,051,044; 8,069,320; 8,073,815; 8,074,035; 8,074,276; 8,145,607; 8,180,984; 8,225,133; 8,261,035; 8,296,264; 8,312,259; 8,347,137; 8,484,427; 8,645,748; 8,732,121; 8,850,060; 8,856,927; 8,996,830; 9,213,697; 9,400,886; 9,424,678; 9,436,558; 9,471,441; 9,501,234; and patent pending applications.

Contents

1. Introduction **1**

1.1 Providing Credentials 2

1.2 Managing Tasks 2

2. Managing Storage Cluster **4**

2.1 Managing Tokens 4

2.1.1 vinfra node token show 4

2.1.2 vinfra node token create 4

2.1.3 vinfra node token validate 5

2.2 Managing Traffic Types and Networks 5

2.2.1 vinfra cluster traffic-type create 5

2.2.2 vinfra cluster traffic-type list 6

2.2.3 vinfra cluster traffic-type show 7

2.2.4 vinfra cluster traffic-type set 7

2.2.5 vinfra cluster traffic-type delete 8

2.2.6 vinfra cluster network create 8

2.2.7 vinfra cluster network list 9

2.2.8 vinfra cluster network show 9

2.2.9 vinfra cluster network set 10

2.2.10 vinfra cluster network set-bulk 11

2.2.11 vinfra cluster network delete 12

2.3 Managing Storage Nodes 12

2.3.1 vinfra node join 12

2.3.2 vinfra node list 14

2.3.3 vinfra node show 15

2.3.4 vinfra node release 15

2.3.5 vinfra node forget 16

2.4 Managing Node Network Interfaces . . . 17
2.4.1 vinfra node network iface list . . . 17
2.4.2 vinfra node network iface show . . . 17
2.4.3 vinfra node network iface . . . 19
2.4.4 vinfra node network iface set . . . 21
2.4.5 vinfra node network bond create . . . 24
2.4.6 vinfra node network bond delete . . . 27
2.4.7 vinfra node network vlan create . . . 27
2.4.8 vinfra node network vlan delete . . . 30
2.5 Managing Node Disks . . . 31
2.5.1 vinfra node disk list . . . 31
2.5.2 vinfra node disk show . . . 32
2.5.3 vinfra node disk assign . . . 33
2.5.4 vinfra node disk release . . . 34
2.5.5 vinfra node disk blink . . . 35
2.5.6 vinfra node iscsi target add . . . 36
2.5.7 vinfra node iscsi target delete . . . 37
2.6 vinfra cluster create . . . 38
2.7 Showing Storage Cluster Overview and Details . . . 39
2.7.1 vinfra cluster overview . . . 39
2.7.2 vinfra cluster show . . . 41
2.8 vinfra cluster delete . . . 42

3. Managing Compute Cluster . . . 43
3.1 vinfra service compute cluster create . . . 43
3.2 Showing Compute Cluster Details and Overview . . . 45
3.2.1 vinfra service compute cluster show . . . 45
3.2.2 vinfra service compute cluster stat . . . 46
3.3 vinfra service compute cluster set . . . 47
3.4 Managing Compute Nodes . . . 47
3.4.1 vinfra service compute node add . . . 47
3.4.2 vinfra service compute node list . . . 48
3.4.3 Showing Compute Node Details . . . 49
3.4.4 vinfra service compute node fence . . . 49
3.4.5 vinfra service compute node unfence . . . 50
3.4.6 vinfra service compute node release . . . 50

3.5 Managing Networks . . . 51
3.5.1 vinfra service compute network create . . . 51
3.5.2 vinfra service compute network list . . . 53
3.5.3 vinfra service compute network show . . . 53
3.5.4 vinfra service compute network set . . . 54
3.5.5 vinfra service compute network delete . . . 55
3.6 Managing Images . . . 55
3.6.1 vinfra service compute image create . . . 55
3.6.2 vinfra service compute image list . . . 57
3.6.3 vinfra service compute image show . . . 57
3.6.4 vinfra service compute image set . . . 58
3.6.5 vinfra service compute image save . . . 59
3.6.6 vinfra service compute image delete . . . 60
3.7 Managing Flavors . . . 60
3.7.1 vinfra service compute flavor create . . . 60
3.7.2 vinfra service compute flavor list . . . 61
3.7.3 vinfra service compute flavor show . . . 61
3.7.4 vinfra service compute flavor delete . . . 62
3.8 Managing Storage Policies . . . 62
3.8.1 vinfra cluster storage-policy create . . . 62
3.8.2 vinfra cluster storage-policy list . . . 63
3.8.3 vinfra cluster storage-policy show . . . 64
3.8.4 vinfra cluster storage-policy set . . . 64
3.8.5 vinfra cluster storage-policy delete . . . 65
3.9 Managing Volumes . . . 66
3.9.1 vinfra service compute volume create . . . 66
3.9.2 vinfra service compute volume list . . . 67
3.9.3 vinfra service compute volume show . . . 68
3.9.4 vinfra service compute volume set . . . 69
3.9.5 vinfra service compute volume extend . . . 70
3.9.6 vinfra service compute server volume attach . . . 70
3.9.7 vinfra service compute server volume detach . . . 71
3.9.8 vinfra service compute volume delete . . . 71
3.10 Managing Virtual Machines . . . 72
3.10.1 vinfra service compute server create . . . 72

3.10.2 vinfra service compute server list . . . 74
3.10.3 vinfra service compute server show . . . 74
3.10.4 vinfra service compute server stat . . . 75
3.10.5 vinfra service compute server iface attach . . . 76
3.10.6 vinfra service compute server iface list . . . 77
3.10.7 vinfra service compute server iface detach . . . 77
3.10.8 vinfra service compute server log . . . 78
3.10.9 vinfra service compute server migrate . . . 78
3.10.10 vinfra service compute server resize . . . 79
3.10.11 vinfra service compute server start . . . 79
3.10.12 vinfra service compute server pause . . . 79
3.10.13 vinfra service compute server unpause . . . 80
3.10.14 vinfra service compute server suspend . . . 80
3.10.15 vinfra service compute server resume . . . 81
3.10.16 vinfra service compute server reboot . . . 81
3.10.17 vinfra service compute server reset-state . . . 81
3.10.18 vinfra service compute server stop . . . 82
3.10.19 vinfra service compute server evacuate . . . 82
3.10.20 vinfra service compute server delete . . . 83
3.11 vinfra service compute cluster delete . . . 83

4. Managing General Settings . . . 84
4.1 Managing Licenses . . . 84
4.1.1 vinfra cluster license load . . . 84
4.1.2 vinfra cluster license show . . . 85
4.2 Managing Users . . . 85
4.2.1 vinfra cluster user list-available-roles . . . 85
4.2.2 vinfra cluster user create . . . 86
4.2.3 vinfra cluster user list . . . 87
4.2.4 vinfra cluster user show . . . 87
4.2.5 vinfra cluster user set . . . 88
4.2.6 vinfra cluster user change-password . . . 89
4.2.7 vinfra cluster user delete . . . 89
4.3 Managing SSH Keys . . . 90
4.3.1 vinfra cluster sshkey add . . . 90
4.3.2 vinfra cluster sshkey list . . . 91

4.3.3 vinfra cluster sshkey delete . . . 92
4.4 Managing External DNS Servers . . . 92
4.4.1 vinfra cluster settings dns show . . . 92
4.4.2 vinfra cluster settings dns set . . . 93
4.5 Configuring Management Node High Availability . . . 93
4.5.1 vinfra cluster ha create . . . 93
4.5.2 vinfra cluster ha join . . . 95
4.5.3 vinfra cluster ha show . . . 96
4.5.4 vinfra cluster ha release . . . 96
4.6 Managing Storage Tier Encryption . . . 97
4.6.1 vinfra cluster settings encryption show . . . 97
4.6.2 vinfra cluster settings encryption set . . . 98
4.7 Managing Alerts . . . 98
4.7.1 vinfra cluster alert list . . . 98
4.7.2 vinfra cluster alert show . . . 99
4.7.3 vinfra cluster alert delete . . . 100
4.8 Managing Audit Log . . . 101
4.8.1 vinfra cluster auditlog list . . . 101
4.8.2 vinfra cluster auditlog show . . . 102
4.9 vinfra cluster problem-report . . . 102

5. Monitoring Storage Cluster . . . 104
5.1 Monitoring General Storage Cluster Parameters . . . 104
5.2 Monitoring Metadata Servers . . . 107
5.3 Monitoring Chunk Servers . . . 108
5.3.1 Understanding Disk Space Usage . . . 109
5.3.1.1 Understanding Allocatable Disk Space . . . 110
5.3.1.2 Viewing Space Occupied by Data Chunks . . . 112
5.3.2 Exploring Chunk States . . . 112
5.4 Monitoring Clients . . . 114
5.5 Monitoring Physical Disks . . . 115
5.6 Monitoring Event Logs . . . 116
5.7 Monitoring Replication Parameters . . . 120

6. Accessing Storage Clusters via iSCSI . . . 122
6.1 iSCSI Workflow Overview . . . 123

6.2 Configuring CLI Tool 124
6.3 Managing Target Groups 125
6.3.1 Creating Target Groups 125
6.3.2 Starting and Stopping Target Groups 127
6.3.3 Listing Target Groups 127
6.3.4 Printing Details of Target Groups 128
6.3.5 Deleting Target Groups 128
6.4 Managing iSCSI Volumes 128
6.4.1 Creating iSCSI Volumes 128
6.4.2 Listing and Printing Details of iSCSI Volumes 129
6.4.3 Attaching iSCSI Volumes to Target Groups 129
6.4.4 Viewing and Setting iSCSI Volume Parameters 129
6.4.5 Increasing iSCSI Volume Size 130
6.4.6 Setting iSCSI Volume Limits 130
6.4.7 Detaching iSCSI Volumes from Target Groups 130
6.4.8 Deleting iSCSI Volumes 130
6.5 Managing Nodes 131
6.5.1 Adding Nodes to Target Groups 131
6.5.2 Setting Node Status 132
6.5.3 Deleting Nodes from Target Groups 132
6.6 Managing Targets and Portals 133
6.6.1 Creating Targets 133
6.6.2 Adding and Removing Target Portals 133
6.6.3 Deleting Targets 134
6.7 Managing CHAP Accounts 134
6.7.1 Creating and Listing CHAP Accounts 134
6.7.2 Changing CHAP Account Details 135
6.7.3 Assigning CHAP Accounts to Target Groups 135
6.7.4 Deleting CHAP Accounts 135
6.8 Managing LUN Views 135
6.8.1 Creating LUN Views 136
6.8.2 Listing LUN Views 136
6.8.3 Changing LUN View Details 136
6.8.4 Deleting LUN Views 137

7. Advanced Tasks 138

7.1 Updating Kernel with ReadyKernel 138
7.1.1 Installing ReadyKernel Patches Automatically 139
7.1.2 Managing ReadyKernel Patches Manually 139
7.1.2.1 Dowloading, Installing, and Loading ReadyKernel Patches 139
7.1.2.2 Loading and Unloading ReadyKernel Patches 139
7.1.2.3 Installing and Removing ReadyKernel Patches for Specific Kernels 140
7.1.2.4 Downgrading ReadyKernel Patches 140
7.1.2.5 Disabling Loading of ReadyKernel Patches on Boot 141
7.1.2.6 Managing ReadyKernel Logs 141
7.2 Managing Guest Tools 141
7.2.1 Installing Guest Tools 141
7.2.1.1 Installing Guest Tools in New VMs 142
7.2.1.2 Installing Guest Tools in Existing VMs 143
7.2.2 Uninstalling Guest Tools 145
7.2.2.1 Uninstalling Guest Tools from Linux VMs 145
7.2.2.2 Uninstalling Guest Tools from Windows VMs 145
7.3 Running Commands in Virtual Machines without Network Connectivity 146
7.3.1 Running Commands in Linux Virtual Machines 146
7.3.2 Running Commands in Windows Virtual Machines 147
7.4 Setting Virtual Machines CPU Model 148
7.5 Creating Linux Templates 148
7.6 Securing OpenStack API Traffic with SSL 150
7.7 Enabling Backup Gateway Geo-Replication 152

CHAPTER 1

Introduction

This guide describes the syntax and parameters of the `vinfra` command-line tool that can be used to manage Acronis Software-Defined Infrastructure from console and automate such management tasks.

Note: While the following chapters provide a reference of specific operations that you can perform with `vinfra`, you can also run `vinfra help` to get a list of all supported commands and their descriptions. For help on a specific command, either run `vinfra help <command>` or `vinfra <command> --help`.

In addition, this guide describes how to use the command line to perform operations unsupported by `vinfra` as of now.

Note that the following operations should not be done from the command line:

- setting custom paths for Acronis Software-Defined Infrastructure services, in particular:
  - creating S3 clusters only in `/mnt/vstorage/vols/s3`,
  - creating iSCSI targets only in `/mnt/vstorage/vols/iscsi`,
- mounting clusters or change cluster mount options,
- configuring firewall with `firewall-cmd`,
- renaming network connections,
- managing MDS/CS,
- managing partitions, LVMs, or software RAID,
- modifying files in `/mnt/vstorage/vols` and `/mnt/vstorage/webcp/backup` directories,

- setting encoding or replication of cluster root.

1.1 Providing Credentials

The `vinfra` CLI tool requires the following information to be used:

- IP address or hostname of the management node (set to `backend-api.svc.vstoragedomain` by default).
- Username (`admin` by default).
- Password (created during installation of Acronis Software-Defined Infrastructure).

This information can be supplied via the `--vinfra-portal`, `--vinfra-username`, and `--vinfra-password` command-line parameters with each command. Alternatively, you can supply it by setting the environment variables `VINFRA_PORTAL`, `VINFRA_USERNAME`, and `VINFRA_PASSWORD` (e.g., in your `~/.bash_profile`). In this case, you will be able to run the CLI tool without the aforementioned command-line parameters.

As you typically run `vinfra` from the management node as admin, the only variable you usually need to set is the password. For example:

```
# export VINFRA_PASSWORD=12345
```

If you installed `vinfra` on a remote machine and/or run it as a different user, you will need to set `VINFRA_PORTAL` and/or `VINFRA_USERNAME` on that machine in addition to `VINFRA_PASSWORD`.

1.2 Managing Tasks

The `vinfra` CLI tool executes some commands immediately, while for other commands (that may take some time to complete) it creates system tasks that are queued. Examples of actions performed via tasks are creating the storage or compute cluster and adding nodes to it.

To keep track of tasks being performed by `vinfra`, use the `vinfra task list` and `vinfra task show` commands. For example:

```
# vinfra task list
+--------------------------------------+---------+----------------------------------------+
| task_id                              | state   | name                                   |
+--------------------------------------+---------+----------------------------------------+
| 8fc27e7a-ba73-471d-9134-e351e1137cf4 | success | backend.tasks.cluster.CreateNewCluster |
| e61377db-9df4-4282-99aa-6a4ae73a7f96 | success | backend.tasks.disks.ApplyDiskRoleTask  |
| a005b748-cb85-40f8-a09d-291a8599bb9c | success | backend.tasks.node.AddNodeInClusterTask |
```

```
+--------------------------------------+---------+----------------------------------------+
# vinfra task show 8fc27e7a-ba73-471d-9134-e351e1137cf4
+---------+----------------------------------------+
| Field   | Value                                  |
+---------+----------------------------------------+
| args    | - stor1                                |
|         | - 7ffa9540-5a20-41d1-b203-e3f349d62565 |
|         | - null                                 |
|         | - null                                 |
| kwargs  | {}                                     |
| name    | backend.tasks.cluster.CreateNewCluster |
| result  | cluster_id: 1                          |
| state   | success                                |
| task_id | 8fc27e7a-ba73-471d-9134-e351e1137cf4   |
+---------+----------------------------------------+
```

CHAPTER 2

Managing Storage Cluster

2.1 Managing Tokens

2.1.1 vinfra node token show

Display the backend token:

```
usage: vinfra node token show
```

Example:

This command shows the details of the current token.

```
# vinfra node token show
+-------+---------------+
| Field | Value         |
+-------+---------------+
| host  | 10.37.130.101 |
| token | dc56d4d2      |
| ttl   | 86398         |
+-------+---------------+
```

2.1.2 vinfra node token create

Create the backend token:

```
usage: vinfra node token create [--ttl <ttl>]
```

`--ttl <ttl>`

Token TTL, in seconds

Example:

```
# vinfra node token create --ttl 86400
+-------+---------------+
| Field | Value         |
+-------+---------------+
| host  | 10.37.130.101 |
| token | dc56d4d2      |
| ttl   | 86398         |
+-------+---------------+
```

This command creates a new token with the time to live (TTL) of 86400 seconds.

2.1.3 vinfra node token validate

Validate the backend token:

```
usage: vinfra node token validate <token>
```

`<token>`

Token value

Example:

```
# vinfra node token validate dc56d4d2
+--------+-------+
| Field  | Value |
+--------+-------+
| status | valid |
+--------+-------+
```

This command validates the token `dc56d4d2`.

2.2 Managing Traffic Types and Networks

2.2.1 vinfra cluster traffic-type create

Create a new traffic type:

```
usage: vinfra cluster traffic-type create --port <port> <traffic-type-name>
```

`--port <port>`

Traffic type port

`<traffic-type-name>`

Traffic type name

Example:

```
# vinfra cluster traffic-type create "MyTrafficType" --port 6900
+-----------+-----------------+
| Field     | Value           |
+-----------+-----------------+
| exclusive | False           |
| name      | MyTrafficType   |
| port      | 6900            |
| type      | custom          |
+-----------+-----------------+
```

This command creates a custom traffic type `MyTrafficType` on port 6900.

2.2.2 vinfra cluster traffic-type list

List available traffic types:

```
usage: vinfra cluster traffic-type list
```

Example:

```
# vinfra cluster traffic-type list
+---------------------+------------+-----------+------+
| name                | type       | exclusive | port |
+---------------------+------------+-----------+------+
| Storage             | predefined | True      |      |
| Internal management | predefined | True      |      |
| OSTOR private       | predefined | True      |      |
| S3 public           | predefined | False     |      |
| iSCSI               | predefined | False     |      |
| NFS                 | predefined | False     |      |
| ABGW private        | predefined | True      |      |
| ABGW public         | predefined | False     |      |
| Admin panel         | predefined | False     |      |
| SSH                 | predefined | False     |      |
| VM public           | predefined | False     |      |
| VM private          | predefined | True      |      |
| Compute API         | predefined | True      |      |
| MyTrafficType       | custom     | False     | 6900 |
+---------------------+------------+-----------+------+
```

This command lists all traffic types in Acronis Software-Defined Infrastructure.

2.2.3 vinfra cluster traffic-type show

Show details of a traffic type:

```
usage: vinfra cluster traffic-type show <traffic-type>
```

`<traffic-type>`

Traffic type name

Example:

```
# vinfra cluster traffic-type show Storage
+-----------+------------+
| Field     | Value      |
+-----------+------------+
| exclusive | True       |
| name      | Storage    |
| port      |            |
| type      | predefined |
+-----------+------------+
```

This command shows the details of the traffic type `Storage`.

2.2.4 vinfra cluster traffic-type set

Modify traffic type parameters:

```
usage: vinfra cluster traffic-type set [--name <name>] [--port <port>] <traffic-type>
```

`--name <name>`

A new name for the traffic type

`--port <port>`

A new port for the traffic type

`<traffic-type>`

Traffic type name

Example:

```
# vinfra cluster traffic-type set "MyTrafficType" \
--name "MyOtherTrafficType" --port 6901
+-----------+--------------------+
| Field     | Value              |
+-----------+--------------------+
| exclusive | False              |
```

```
| name      | MyOtherTrafficType |
| port      | 6901               |
| type      | custom             |
+-----------+--------------------+
```

This command renames the traffic type `MyTrafficType` to `MyOtherTrafficType` and changes its port to 6901.

2.2.5 vinfra cluster traffic-type delete

Delete a traffic type:

```
usage: vinfra cluster traffic-type delete <traffic-type>
```

`<traffic-type>`

Traffic type name

Example:

```
# vinfra cluster traffic-type delete "MyOtherTrafficType"
Operation successful
```

This command deletes the custom traffic type `MyOtherTrafficType`.

2.2.6 vinfra cluster network create

Create a new network:

```
usage: vinfra cluster network create [--traffic-types <traffic-types>] <network-name>
```

`--traffic-types <traffic-types>`

A comma-separated list of traffic type IDs or names

`<network-name>`

Network name

Example:

```
# vinfra cluster network create MyNet --traffic-types ssh
+-------+--------------------------------------+
| Field | Value                                |
+-------+--------------------------------------+
| id    | 03d5eeb3-1833-4626-885d-dd066635f5de |
| name  | MyNet                                |
| roles | - SSH                                |
| type  | Custom                               |
```

```
+-------+--------------------------------------+
```

This command creates a custom network `MyNet` and assigns the traffic type `SSH` to it.

2.2.7 vinfra cluster network list

List available networks:

```
usage: vinfra cluster network list
```

Example:

```
# vinfra cluster network list
+--------------------------------------+---------+-----------------------+
| id                                   | name    | roles                 |
+--------------------------------------+---------+-----------------------+
| 358bdc39-cd8b-4565-8ebf-e7c12dcd1cf7 | Public  | - ABGW public         |
|                                      |         | - iSCSI               |
|                                      |         | - NFS                 |
|                                      |         | - S3 public           |
|                                      |         | - SSH                 |
|                                      |         | - Admin Panel         |
| 6095a997-e5f1-493d-a750-41ddf277153b | Private | - ABGW private        |
|                                      |         | - Internal Management |
|                                      |         | - OSTOR private       |
|                                      |         | - SSH                 |
|                                      |         | - Storage             |
+--------------------------------------+---------+-----------------------+
```

This command lists all networks in Acronis Software-Defined Infrastructure.

2.2.8 vinfra cluster network show

Show details of a network:

```
usage: vinfra cluster network show <network>
```

`<network>`

Network ID or name

Example:

```
# vinfra cluster network show MyNet
+-------+--------------------------------------+
| Field | Value                                |
+-------+--------------------------------------+
```

```
| id    | 03d5eeb3-1833-4626-885d-dd066635f5de |
| name  | MyNet                                |
| roles | - SSH                                |
| type  | Custom                               |
+-------+--------------------------------------+
```

This command shows the details of the custom network `MyNet`.

2.2.9 vinfra cluster network set

Modify network parameters:

```
usage: vinfra cluster network set [--name <network-name>]
                                  [--traffic-types <traffic-types> |
                                  --add-traffic-types <traffic-types> |
                                  --del-traffic-types <traffic-types>]
                                  <network>
```

`--name <network-name>`

Network name

`--traffic-types <traffic-types>`

A comma-separated list of traffic type names (overwrites network's current traffic types)

`--add-traffic-types <traffic-types>`

A comma-separated list of traffic type names (adds the specified traffic types to the network)

`--del-traffic-types <traffic-types>`

A comma-separated list of traffic type names (removes the specified traffic types from the network)

`<network>`

Network ID or name

Example:

```
# vinfra cluster network set MyNet --name MyOtherNet --add-traffic-types iscsi,nfs
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | b29f6f66-37d7-47de-b02e-9f4087ad932b |
+---------+--------------------------------------+
```

This command creates a task to rename the network `MyNet` to `MyOtherNet` and assign to it the traffic types `iSCSI` and `NFS`.

Task outcome:

```
# vinfra task show b29f6f66-37d7-47de-b02e-9f4087ad932b
+---------+-------------------------------------------------------------+
| Field   | Value                                                       |
+---------+-------------------------------------------------------------+
| args    | - 03d5eeb3-1833-4626-885d-dd066635f5de                      |
| kwargs  | name: MyOtherNet                                            |
|         | roles:                                                      |
|         | - ssh                                                       |
|         | - iscsi                                                     |
|         | - nfs                                                       |
| name    | backend.presentation.network.roles.tasks.RolesSetChangeTask |
| result  | id: 03d5eeb3-1833-4626-885d-dd066635f5de                    |
|         | name: MyOtherNet                                            |
|         | roles:                                                      |
|         | - iSCSI                                                     |
|         | - NFS                                                       |
|         | - SSH                                                       |
|         | type: Custom                                                |
| state   | success                                                     |
| task_id | b29f6f66-37d7-47de-b02e-9f4087ad932b                        |
+---------+-------------------------------------------------------------+
```

2.2.10 vinfra cluster network set-bulk

Modify traffic types of multiple networks:

```
usage: vinfra cluster network set-bulk --network <network>:<traffic-types>
```

`--network <network>:<traffic-types>`

Network configuration in the format:

- `<network>`: network ID or name;
- `<traffic-types>`: a comma-separated list of traffic type names;

(this option can be used multiple times).

Example:

```
# vinfra cluster network set-bulk --network MyNet1:snmp --network MyNet2:ssh,snmp
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | c774f55d-c45b-42cd-ac9e-16fc196e9283 |
+---------+--------------------------------------+
```

This command creates a task to change the traffic type set of the network `MyNet1` to `SNMP` and that of `MyNet2` to `SSH` and `SNMP`.

Task outcome:

```
# vinfra task show c774f55d-c45b-42cd-ac9e-16fc196e9283
+---------+-----------------------------------------------------------------+
| Field   | Value                                                           |
+---------+-----------------------------------------------------------------+
| details |                                                                 |
| name    | backend.presentation.network.roles.tasks.RolesSetBulkChangeTask |
| result  | - id: adf49487-9deb-4180-bb0c-08a906257981                      |
|         |   name: MyNet1                                                  |
|         |   roles:                                                        |
|         |   - SNMP                                                        |
|         |   type: Custom                                                  |
|         | - id: 3f6ff4a3-31bc-440b-a36f-d755c80d5932                      |
|         |   name: MyNet2                                                  |
|         |   roles:                                                        |
|         |   - SNMP                                                        |
|         |   - SSH                                                         |
|         |   type: Custom                                                  |
| state   | success                                                         |
| task_id | c774f55d-c45b-42cd-ac9e-16fc196e9283                            |
+---------+-----------------------------------------------------------------+
```

2.2.11 vinfra cluster network delete

Delete a network:

```
usage: vinfra cluster network delete <network>
```

`<network>`

Network ID or name

Example:

```
# vinfra cluster network delete MyOtherNet
Operation successful
```

This command deletes the network `MyOtherNet`.

2.3 Managing Storage Nodes

2.3.1 vinfra node join

Join a node to the storage cluster:

```
usage: vinfra node join [--disk <disk>:<role>[:<key=value,...>]] <node>
```

`--disk <disk>:<role> [:<key=value,...>]`

Disk configuration in the format:

- `<disk>`: disk device ID or name;
- `<role>`: disk role (`cs`, `mds`, `journal`, `mds-journal`, `mds-system`, `cs-system`, `system`);
- comma-separated `key=value` pairs with keys (optional):
  - `tier`: disk tier (0, 1, 2 or 3);
  - `journal-tier`: journal (cache) disk tier (0, 1, 2 or 3);
  - `journal-type`: journal (cache) disk type (`no_cache`, `inner_cache` or `external_cache`);
  - `journal-disk`: journal (cache) disk ID or device name;
  - `journal-size`: journal (cache) disk size, in bytes;
  - `bind-address`: bind IP address for the metadata service.

E.g., `sda:cs:tier=0,journal-type=inner_cache`. This option can be used multiple times.

<node>

Node ID or hostname

Example:

```
# vinfra node join f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4 \
--disk sda:mds-system \
--disk sdb:cs \
--disk sdc:cs
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | a2713068-9544-4ea1-8ec8-69a068cf86f2 |
+---------+--------------------------------------+
```

This command creates a task to add the node with the ID `f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4` to the storage cluster and assigns roles to disks: `mds-system` to `sda`, `cs` to `sdb`` and `sdc`.

Task outcome:

```
# vinfra task show a2713068-9544-4ea1-8ec8-69a068cf86f2
+---------+------------------------------------------+
| Field   | Value                                    |
+---------+------------------------------------------+
```

```
| args    | - f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4     |
|         | - 1                                        |
| kwargs  | disks:                                     |
|         | - id: 85F32403-94A9-465A-9E6C-C1A2B41294FC |
|         |   role: mds-system                         |
|         |   service_params: {}                       |
|         | - id: FE0B5876-E054-489B-B0FD-72429BEFD46A |
|         |   role: cs                                 |
|         |   service_params: {}                       |
|         | - id: D3BEF4BB-AA3B-4DB6-9376-BC7CDA636700 |
|         |   role: cs                                 |
|         |   service_params: {}                       |
| name    | backend.tasks.node.AddNodeInClusterTask    |
| result  | {}                                         |
| state   | success                                    |
| task_id | a2713068-9544-4ea1-8ec8-69a068cf86f2       |
+---------+--------------------------------------------+
```

2.3.2 vinfra node list

List storage nodes:

```
usage: vinfra node list
```

Example:

```
# vinfra node list
+--------------------------+---------------+------------+-----------+-------------+----------+
| id                       | host          | is_primary | is_online | is_assigned | is_in_ha |
+--------------------------+---------------+------------+-----------+-------------+----------+
| 09bb6b84-70a5-41ae-b342- | node001.<...> | True       | True      | True        | False    |
| 23e5fc7cc126             |               |            |           |             |          |
| 187edb11-38c5-487b-bd7f- | node002.<...> | False      | True      | True        | False    |
| 57b0fa4b733c             |               |            |           |             |          |
| e6255aed-d6e7-41b2-ba90- | node003.<...> | False      | True      | True        | False    |
| 86164c1cd9a6             |               |            |           |             |          |
+--------------------------+---------------+------------+-----------+-------------+----------+
```

This command lists all nodes registered in Acronis Software-Defined Infrastructure (both unassigned and used in the storage cluster).

2.3.3 vinfra node show

Show storage node details:

```
usage: vinfra node show <node>
```

<node>

Node ID or hostname

Example:

```
# vinfra node show 4f96acf5-3bc8-4094-bcb6-4d1953be7b55
+---------------+--------------------------------------+
| Field         | Value                                |
+---------------+--------------------------------------+
| cpu_cores     | 2                                    |
| host          | stor-1.example.com.vstoragedomain.   |
| id            | 4f96acf5-3bc8-4094-bcb6-4d1953be7b55 |
| ipaddr        | stor-1.example.com.vstoragedomain.   |
| is_assigned   | False                                |
| is_in_ha      | False                                |
| is_installing | False                                |
| is_online     | True                                 |
| is_primary    | True                                 |
| is_virt       | True                                 |
| mem_total     | 8201310208                           |
| roles         | management:                          |
|               |   is_primary: true                   |
| tasks         |                                      |
+---------------+--------------------------------------+
```

This command show the details of the node with the ID `4f96acf5-3bc8-4094-bcb6-4d1953be7b55`.

2.3.4 vinfra node release

Release a node from the storage cluster. Start data migration from the node as well as cluster replication and rebalancing to meet the configured redundancy level:

```
usage: vinfra node release [--force] <node>
```

`--force`

Release node without data migration

<node>

Node ID or hostname

Example:

```
# vinfra node release f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | c2a653a2-8991-4b3a-8bdf-5c0872aa75b3 |
+---------+--------------------------------------+
```

This command creates a task to release the node with the ID `f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4` from the storage cluster with migration of data to maintain the set redundancy mode.

Task outcome:

```
# vinfra task show c2a653a2-8991-4b3a-8bdf-5c0872aa75b3
+---------+----------------------------------------+
| Field   | Value                                  |
+---------+----------------------------------------+
| args    | - f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4 |
|         | - false                                |
| kwargs  | {}                                     |
| name    | backend.tasks.node.ReleaseNodeTask     |
| state   | success                                |
| task_id | c2a653a2-8991-4b3a-8bdf-5c0872aa75b3   |
+---------+----------------------------------------+
```

2.3.5 vinfra node forget

Remove a node from the storage cluster:

```
usage: vinfra node forget <node>
```

<node>

Node ID or hostname

Example:

```
# vinfra node forget fd1e46de-6e17-4571-bf6b-1ac34ec1c225
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | 0eac3b74-e8f5-4974-9efe-a9070187d83c |
+---------+--------------------------------------+
```

This commands creates a task to unregister the node with the ID `fd1e46de-6e17-4571-bf6b-1ac34ec1c225` from Acronis Software-Defined Infrastructure.

Task outcome:

```
# vinfra task show 0eac3b74-e8f5-4974-9efe-a9070187d83c
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| args    | - fd1e46de-6e17-4571-bf6b-1ac34ec1c225 |
| kwargs  | {}                                   |
| name    | backend.tasks.node.DeleteNodeTask    |
| state   | success                              |
| task_id | 0eac3b74-e8f5-4974-9efe-a9070187d83c |
+---------+--------------------------------------+
```

2.4 Managing Node Network Interfaces

2.4.1 vinfra node network iface list

List node network interfaces:

```
usage: vinfra node network iface list (-a | --node <node>)
```

`-a, --all`

List all network interfaces on all nodes

`--node <node>`

Node ID or hostname to list network interfaces on

Example:

This command shows network interfaces of the node with the ID `4f96acf5-3bc8-4094-bcb6-4d1953be7b55`.

```
# vinfra node network iface list --node 4f96acf5-3bc8-4094-bcb6-4d1953be7b55
+------+--------------------------------------+--------------------+-------+---------+
| name | node_id                              | ipv4               | state | network |
+------+--------------------------------------+--------------------+-------+---------+
| eth0 | 4f96acf5-3bc8-4094-bcb6-4d1953be7b55 | - 10.94.29.218/16  | up    | Public  |
| eth1 | 4f96acf5-3bc8-4094-bcb6-4d1953be7b55 | - 10.37.130.101/24 | up    | Private |
+------+--------------------------------------+--------------------+-------+---------+
```

2.4.2 vinfra node network iface show

Show details of a network interface:

```
usage: vinfra node network iface show --node <node> <iface>
```

`--node <node>`

Node ID or hostname

`<iface>`

Network interface name

Example:

```
# vinfra node network iface show eth0 --node 4f96acf5-3bc8-4094-bcb6-4d1953be7b55
+----------------------+--------------------------------------+
| Field                | Value                                |
+----------------------+--------------------------------------+
| contained_in         |                                      |
| dhcp4                | 10.94.29.218                         |
| dhcp4_enabled        | True                                 |
| dhcp6                | fe80::21c:42ff:fe2a:4fdf             |
| dhcp6_enabled        | True                                 |
| dns4                 | - 127.0.0.1                          |
| dns6                 | []                                   |
| duplex               |                                      |
| gw4                  | 10.94.0.1                            |
| gw6                  |                                      |
| ignore_auto_dns_v4   | False                                |
| ignore_auto_dns_v6   | False                                |
| ignore_auto_routes_v4 | False                                |
| ignore_auto_routes_v6 | False                                |
| ipv4                 | - 10.94.29.218/16                    |
| ipv6                 | - fe80::21c:42ff:fe2a:4fdf/64        |
| mac_addr             | 00:1c:42:2a:4f:df                    |
| mtu                  | 1500                                 |
| multicast            | True                                 |
| name                 | eth0                                 |
| node_id              | 4f96acf5-3bc8-4094-bcb6-4d1953be7b55 |
| plugged              | True                                 |
| roles_set            | 237e58dd-6c10-49c1-be7f-7ddf7de2efd1 |
| rx_bytes             | 1844502614                           |
| rx_dropped           | 0                                    |
| rx_errors            | 0                                    |
| rx_overruns          | 0                                    |
| rx_packets           | 11543284                             |
| speeds               | current: null                        |
|                      | max: null                            |
| state                | up                                   |
| tx_bytes             | 28477979                             |
| tx_dropped           | 0                                    |
| tx_errors            | 0                                    |
| tx_overruns          | 0                                    |
| tx_packets           | 107649                               |
| type                 | iface                                |
+----------------------+--------------------------------------+
```

This command shows the details of the network interface `eth0` located on the node with the ID

4f96acf5-3bc8-4094-bcb6-4d1953be7b55.

2.4.3 vinfra node network iface

Bring a network interface up:

```
usage: vinfra node network iface up --node <node> <iface>
```

--node <node>

Node ID or hostname

<iface>

Network interface name

Bring a network interface down:

```
usage: vinfra node network iface down --node <node> <iface>
```

--node <node>

Node ID or hostname

<iface>

Network interface name

Example:

```
# vinfra node network iface up eth2 --node 4f96acf5-3bc8-4094-bcb6-4d1953be7b55
+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| contained_in          |                                      |
| dhcp4                 | 10.37.130.138                        |
| dhcp4_enabled         | True                                 |
| dhcp6                 | fe80::21c:42ff:fef8:5b90             |
| dhcp6_enabled         | True                                 |
| dns4                  | - 127.0.0.1                          |
| dns6                  | []                                   |
| duplex                |                                      |
| gw4                   | 10.94.0.1                            |
| gw6                   |                                      |
| ignore_auto_dns_v4    | False                                |
| ignore_auto_dns_v6    | False                                |
| ignore_auto_routes_v4 | False                                |
| ignore_auto_routes_v6 | False                                |
| ipv4                  | - 10.37.130.138/24                   |
| ipv6                  | - fe80::21c:42ff:fef8:5b90/64        |
| mac_addr              | 00:1c:42:f8:5b:90                    |
```

```
| mtu                   | 1500                                 |
| multicast             | True                                 |
| name                  | eth2                                 |
| node_id               | 4f96acf5-3bc8-4094-bcb6-4d1953be7b55 |
| plugged               | True                                 |
| roles_set             |                                      |
| rx_bytes              | 97632                                |
| rx_dropped            | 0                                    |
| rx_errors             | 0                                    |
| rx_overruns           | 0                                    |
| rx_packets            | 1258                                 |
| speeds                | current: null                        |
|                       | max: null                            |
| state                 | up                                   |
| tx_bytes              | 1116                                 |
| tx_dropped            | 0                                    |
| tx_errors             | 0                                    |
| tx_overruns           | 0                                    |
| tx_packets            | 8                                    |
| type                  | iface                                |
+-----------------------+--------------------------------------+
```

This commands brings up the network interface `eth2` located on the node with the ID `4f96acf5-3bc8-4094-bcb6-4d1953be7b55`.

```
# vinfra node network iface down eth2 --node 4f96acf5-3bc8-4094-bcb6-4d1953be7b55
+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| contained_in          |                                      |
| dhcp4                 |                                      |
| dhcp4_enabled         | True                                 |
| dhcp6                 |                                      |
| dhcp6_enabled         | True                                 |
| dns4                  | - 127.0.0.1                          |
| dns6                  | []                                   |
| duplex                |                                      |
| gw4                   | 10.94.0.1                            |
| gw6                   |                                      |
| ignore_auto_dns_v4    | False                                |
| ignore_auto_dns_v6    | False                                |
| ignore_auto_routes_v4 | False                                |
| ignore_auto_routes_v6 | False                                |
| ipv4                  | []                                   |
| ipv6                  | []                                   |
| mac_addr              | 00:1c:42:f8:5b:90                    |
| mtu                   | 1500                                 |
| multicast             | True                                 |
| name                  | eth2                                 |
| node_id               | 4f96acf5-3bc8-4094-bcb6-4d1953be7b55 |
| plugged               | False                                |
| roles_set             |                                      |
```

```
| rx_bytes              | 97984                                |
| rx_dropped            | 0                                    |
| rx_errors             | 0                                    |
| rx_overruns           | 0                                    |
| rx_packets            | 1264                                 |
| speeds                | current: null                        |
|                       | max: null                            |
| state                 | down                                 |
| tx_bytes              | 1116                                 |
| tx_dropped            | 0                                    |
| tx_errors             | 0                                    |
| tx_overruns           | 0                                    |
| tx_packets            | 8                                    |
| type                  | iface                                |
+-----------------------+--------------------------------------+
```

This commands brings down the network interface `eth2` located on the node with the ID `4f96acf5-3bc8-4094-bcb6-4d1953be7b55`.

2.4.4 vinfra node network iface set

Modify network interface parameters (overwrites omitted options to interace default values):

```
usage: vinfra node network iface set [--ipv4 <ipv4>] [--ipv6 <ipv6>] [--gw4 <gw4>]
                                     [--gw6 <gw6>] [--mtu <mtu>] [--dhcp4 | --no-dhcp4]
                                     [--dhcp6 | --no-dhcp6] [--network <network>]
                                     [--auto-routes-v4 | --ignore-auto-routes-v4]
                                     [--auto-routes-v6 | --ignore-auto-routes-v6]
                                     [--connected-mode | --datagram-mode] --node <node>
                                     <iface>
```

`--ipv4 <ipv4>`

A comma-separated list of IPv4 addresses

`--ipv6 <ipv6>`

A comma-separated list of IPv6 addresses

`--gw4 <gw4>`

Gateway IPv4 address

`--gw6 <gw6>`

Gateway IPv6 address

`--mtu <mtu>`

MTU interface value

`--dhcp4`
: Enable DHCPv4

`--no-dhcp4`
: Disable DHCPv4

`--dhcp6`
: Enable DHCPv6

`--no-dhcp6`
: Disable DHCPv6

`--auto-routes-v4`
: Enable automatic IPv4 routes

`--ignore-auto-routes-v4`
: Ignore automatic IPv4 routes

`--auto-routes-v6`
: Enable automatic IPv6 routes

`--ignore-auto-routes-v6`
: Ignore automatic IPv6 routes

`--network <network>`
: Network ID or name

`--connected-mode`
: Enable connected mode (InfiniBand interfaces only)

`--datagram-mode`
: Enable datagram mode (InfiniBand interfaces only)

`--node <node>`
: Node ID or hostname

`<iface>`
: Network interface name

Example:

```
# vinfra node network iface set eth2 --network Private \
--node 4f96acf5-3bc8-4094-bcb6-4d1953be7b55
+---------+-------------------------------------+
```

```
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | 8a378098-6760-4fe9-ac20-1f18a8ed9d2e |
+---------+--------------------------------------+
```

This command creates a task to assign the network interface `eth2` located on the node with the ID `4f96acf5-3bc8-4094-bcb6-4d1953be7b55` to the network `Private`.

Task outcome:

```
# vinfra task show 8a378098-6760-4fe9-ac20-1f18a8ed9d2e
+---------+------------------------------------------------------------------+
| Field   | Value                                                            |
+---------+------------------------------------------------------------------+
| args    | - 4f96acf5-3bc8-4094-bcb6-4d1953be7b55                           |
|         | - eth2                                                           |
| kwargs  | roles_set: 6095a997-e5f1-493d-a750-41ddf277153b                  |
| name    | backend.presentation.network.tasks.NetworkInterfaceChangeTask    |
| result  | contained_in: null                                               |
|         | dhcp4: null                                                      |
|         | dhcp4_enabled: false                                             |
|         | dhcp6: null                                                      |
|         | dhcp6_enabled: false                                             |
|         | duplex: null                                                     |
|         | gw4: null                                                        |
|         | gw6: null                                                        |
|         | ignore_auto_routes_v4: true                                      |
|         | ignore_auto_routes_v6: true                                      |
|         | ipv4:                                                            |
|         | - 10.37.130.103/24                                               |
|         | ipv6:                                                            |
|         | - fe80::21c:42ff:fe75:7c4d/64                                    |
|         | mac_addr: 00:1c:42:75:7c:4d                                      |
|         | mtu: 1500                                                        |
|         | multicast: true                                                  |
|         | name: eth2                                                       |
|         | node_id: 4f96acf5-3bc8-4094-bcb6-4d1953be7b55                    |
|         | plugged: true                                                    |
|         | roles_set: 6095a997-e5f1-493d-a750-41ddf277153b                  |
|         | rx_bytes: 38156                                                  |
|         | rx_dropped: 0                                                    |
|         | rx_errors: 0                                                     |
|         | rx_overruns: 0                                                   |
|         | rx_packets: 225                                                  |
|         | speeds:                                                          |
|         |   current: null                                                  |
|         |   max: null                                                      |
|         | state: up                                                        |
|         | tx_bytes: 13087                                                  |
|         | tx_dropped: 0                                                    |
|         | tx_errors: 0                                                     |
|         | tx_overruns: 0                                                   |
```

```
|         | tx_packets: 145                                                  |
|         | type: iface                                                      |
| state   | success                                                          |
| task_id | 8a378098-6760-4fe9-ac20-1f18a8ed9d2e                             |
+---------+------------------------------------------------------------------+
```

2.4.5 vinfra node network bond create

Create a network bonding:

```
usage: vinfra node network bond create [--ipv4 <ipv4>] [--ipv6 <ipv6>] [--gw4 <gw4>]
                                       [--gw6 <gw6>] [--mtu <mtu>] [--dhcp4 | --no-dhcp4]
                                       [--dhcp6 | --no-dhcp6] [--network <network>]
                                       [--auto-routes-v4 | --ignore-auto-routes-v4]
                                       [--auto-routes-v6 | --ignore-auto-routes-v6]
                                       [--bonding-opts <bonding_opts>] --node <node>
                                       --bond-type <bond-type> --ifaces <ifaces>
```

`--ipv4 <ipv4>`

A comma-separated list of IPv4 addresses

`--ipv6 <ipv6>`

A comma-separated list of IPv6 addresses

`--gw4 <gw4>`

Gateway IPv4 address

`--gw6 <gw6>`

Gateway IPv6 address

`--mtu <mtu>`

MTU interface value

`--dhcp4`

Enable DHCPv4

`--no-dhcp4`

Disable DHCPv4

`--dhcp6`

Enable DHCPv6

`--no-dhcp6`

Disable DHCPv6

`--auto-routes-v4`

Enable automatic IPv4 routes

`--ignore-auto-routes-v4`

Ignore automatic IPv4 routes

`--auto-routes-v6`

Enable automatic IPv6 routes

`--ignore-auto-routes-v6`

Ignore automatic IPv6 routes

`--network <network>`

Network ID or name

`--bonding-opts <bonding_opts>`

Additional bonding options

`--bond-type <bond-type>`

Bond type (`balance-rr`, `active-backup`, `balance-xor`, `broadcast`, `802.3ad`, `balance-tlb`, `balance-alb`)

`--node <node>`

Node ID or hostname

`--ifaces <ifaces>`

A comma-separated list of network interface names, e.g., `iface1,iface2,...,iface<N>`

Example:

```
# vinfra node network bond create --ifaces eth2,eth3 --bond-type balance-xor \
--dhcp4 --node fd1e46de-6e17-4571-bf6b-1ac34ec1c225
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | becf96ad-9e39-4bec-b82c-4e1219a196de |
+---------+--------------------------------------+
```

This command creates a task to bond network interfaces `eth2` and `eth3` into `bond0` of the type `balance-xor` on the node with the ID `fd1e46de-6e17-4571-bf6b-1ac34ec1c225`.

Task outcome:

```
# vinfra task show becf96ad-9e39-4bec-b82c-4e1219a196de
+---------+------------------------------------------------------------------------+
| Field   | Value                                                                  |
+---------+------------------------------------------------------------------------+
```

```
| args    | - fd1e46de-6e17-4571-bf6b-1ac34ec1c225                                 |
| kwargs  | bond_type: balance-xor                                                 |
|         | ifaces:                                                                |
|         | - eth2                                                                 |
|         | - eth3                                                                 |
|         | registration_token: 3102ed1a                                           |
| name    | backend.presentation.network.tasks.NetworkInterfaceCreateBondingTask   |
| result  | bond_type: balance-xor                                                 |
|         | dhcp4: 10.37.130.117                                                   |
|         | dhcp4_enabled: true                                                    |
|         | dhcp6: fe80::21c:42ff:fe81:27d0                                        |
|         | dhcp6_enabled: true                                                    |
|         | duplex: null                                                           |
|         | gw4: 10.94.0.1                                                         |
|         | gw6: null                                                              |
|         | ignore_auto_routes_v4: false                                           |
|         | ignore_auto_routes_v6: false                                           |
|         | ipv4:                                                                  |
|         | - 10.37.130.117/24                                                     |
|         | ipv6:                                                                  |
|         | - fe80::21c:42ff:fe81:27d0/64                                          |
|         | mac_addr: 00:1c:42:81:27:d0                                            |
|         | mtu: 1500                                                              |
|         | multicast: true                                                        |
|         | name: bond0                                                            |
|         | node_id: fd1e46de-6e17-4571-bf6b-1ac34ec1c225                          |
|         | plugged: true                                                          |
|         | roles_set: ''                                                          |
|         | rx_bytes: 3048                                                         |
|         | rx_dropped: 0                                                          |
|         | rx_errors: 0                                                           |
|         | rx_overruns: 0                                                         |
|         | rx_packets: 22                                                         |
|         | speeds:                                                                |
|         |   current: null                                                        |
|         |   max: null                                                            |
|         | state: up                                                              |
|         | tx_bytes: 1782                                                         |
|         | tx_dropped: 0                                                          |
|         | tx_errors: 0                                                           |
|         | tx_overruns: 0                                                         |
|         | tx_packets: 13                                                         |
|         | type: bonding                                                          |
| state   | success                                                                |
| task_id | becf96ad-9e39-4bec-b82c-4e1219a196de                                   |
+---------+------------------------------------------------------------------------+
```

2.4.6 vinfra node network bond delete

Delete a network bonding:

```
usage: vinfra node network bond delete --node <node> <iface>
```

`--node <node>`

Node ID or hostname

`<iface>`

Network interface name

Example:

```
# vinfra node network bond delete bond0 --node fd1e46de-6e17-4571-bf6b-1ac34ec1c225
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | 91a0825a-3d33-41a0-8b87-6fc151dbc45f |
+---------+--------------------------------------+
```

This command creates a task to delete the bond `bond0` from the node with the ID `fd1e46de-6e17-4571-bf6b-1ac34ec1c225`.

Task outcome:

```
# vinfra task show 91a0825a-3d33-41a0-8b87-6fc151dbc45f
+---------+-------------------------------------------------------------------+
| Field   | Value                                                             |
+---------+-------------------------------------------------------------------+
| args    | - fd1e46de-6e17-4571-bf6b-1ac34ec1c225                            |
|         | - bond0                                                           |
| kwargs  | {}                                                                |
| name    | backend.presentation.network.tasks.NetworkInterfaceRemoveTask     |
| state   | success                                                           |
| task_id | 91a0825a-3d33-41a0-8b87-6fc151dbc45f                              |
+---------+-------------------------------------------------------------------+
```

2.4.7 vinfra node network vlan create

Create a VLAN:

```
usage: vinfra node network vlan create [--ipv4 <ipv4>] [--ipv6 <ipv6>] [--gw4 <gw4>]
                                       [--gw6 <gw6>] [--mtu <mtu>] [--dhcp4 | --no-dhcp4]
                                       [--dhcp6 | --no-dhcp6] [--network <network>]
                                       [--auto-routes-v4 | --ignore-auto-routes-v4]
                                       [--auto-routes-v6 | --ignore-auto-routes-v6]
```

```
                                    --node <node> --iface <iface> --tag <tag>
```

`--ipv4 <ipv4>`

A comma-separated list of IPv4 addresses

`--ipv6 <ipv6>`

A comma-separated list of IPv6 addresses

`--gw4 <gw4>`

Gateway IPv4 address

`--gw6 <gw6>`

Gateway IPv6 address

`--mtu <mtu>`

MTU interface value

`--dhcp4`

Enable DHCPv4

`--no-dhcp4`

Disable DHCPv4

`--dhcp6`

Enable DHCPv6

`--no-dhcp6`

Disable DHCPv6

`--auto-routes-v4`

Enable automatic IPv4 routes

`--ignore-auto-routes-v4`

Ignore automatic IPv4 routes

`--auto-routes-v6`

Enable automatic IPv6 routes

`--ignore-auto-routes-v6`

Ignore automatic IPv6 routes

`--network <network>`

Network ID or name

`--node <node>`

Node ID or hostname

`--iface <iface>`

Interface name

`--tag <tag>`

VLAN tag number

Example:

```
# vinfra node network vlan create --iface eth2 --tag 100 --dhcp4 \
--node fd1e46de-6e17-4571-bf6b-1ac34ec1c225
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | 0b978acd-367b-47ad-8572-4f4e6ffb8877 |
+---------+--------------------------------------+
```

This command creates a task to create a VLAN with the tag `100` on the network interface `eth2` on the node with the ID `fd1e46de-6e17-4571-bf6b-1ac34ec1c225`.

Task outcome:

```
# vinfra task show 0b978acd-367b-47ad-8572-4f4e6ffb8877
+---------+-------------------------------------------------------------------------+
| Field   | Value                                                                   |
+---------+-------------------------------------------------------------------------+
| args    | - fd1e46de-6e17-4571-bf6b-1ac34ec1c225                                  |
| kwargs  | iface: eth2                                                             |
|         | tag: 100                                                                |
| name    | backend.presentation.network.tasks.NetworkInterfaceCreateVlanTask       |
| result  | built_on: eth2                                                          |
|         | dhcp4: null                                                             |
|         | dhcp4_enabled: false                                                    |
|         | dhcp6: null                                                             |
|         | dhcp6_enabled: false                                                    |
|         | duplex: null                                                            |
|         | gw4: null                                                               |
|         | gw6: null                                                               |
|         | ignore_auto_routes_v4: true                                             |
|         | ignore_auto_routes_v6: true                                             |
|         | ipv4: []                                                                |
|         | ipv6:                                                                   |
|         | - fe80::21c:42ff:fe81:27d0/64                                           |
|         | mac_addr: 00:1c:42:81:27:d0                                             |
|         | mtu: 1500                                                               |
|         | multicast: true                                                         |
|         | name: eth2.100                                                          |
|         | node_id: fd1e46de-6e17-4571-bf6b-1ac34ec1c225                           |
```

```
|         | plugged: true                                                          |
|         | roles_set: ''                                                          |
|         | rx_bytes: 0                                                            |
|         | rx_dropped: 0                                                          |
|         | rx_errors: 0                                                           |
|         | rx_overruns: 0                                                         |
|         | rx_packets: 0                                                          |
|         | speeds:                                                                |
|         |   current: null                                                        |
|         |   max: null                                                            |
|         | state: up                                                              |
|         | tag: 100                                                               |
|         | tx_bytes: 738                                                          |
|         | tx_dropped: 0                                                          |
|         | tx_errors: 0                                                           |
|         | tx_overruns: 0                                                         |
|         | tx_packets: 7                                                          |
|         | type: vlan                                                             |
| state   | success                                                                |
| task_id | 0b978acd-367b-47ad-8572-4f4e6ffb8877                                   |
+---------+------------------------------------------------------------------------+
```

2.4.8 vinfra node network vlan delete

Delete a VLAN:

```
usage: vinfra node network vlan delete --node <node> <iface>
```

`--node <node>`

Node ID or hostname

`<iface>`

Network interface name

Example:

```
# vinfra node network vlan delete eth2.100 --node fd1e46de-6e17-4571-bf6b-1ac34ec1c225
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | b7e94ae1-14aa-4e73-b71a-605cf8b3f458 |
+---------+--------------------------------------+
```

This command creates a task to delete the VLAN interface `eth2.100` from the node with the ID `fd1e46de-6e17-4571-bf6b-1ac34ec1c225`.

Task outcome:

```
# vinfra task show b7e94ae1-14aa-4e73-b71a-605cf8b3f458
+---------+-------------------------------------------------------------+
| Field   | Value                                                       |
+---------+-------------------------------------------------------------+
| args    | - fd1e46de-6e17-4571-bf6b-1ac34ec1c225                      |
|         | - eth2.100                                                  |
| kwargs  | {}                                                          |
| name    | backend.presentation.network.tasks.NetworkInterfaceRemoveTask |
| state   | success                                                     |
| task_id | b7e94ae1-14aa-4e73-b71a-605cf8b3f458                        |
+---------+-------------------------------------------------------------+
```

2.5 Managing Node Disks

2.5.1 vinfra node disk list

List node disks:

```
usage: vinfra node disk list (-a | --node <node>)
```

`-a, --all`

List disks on all nodes

`--node <node>`

Node ID or hostname to list disks on

Example:

```
# vinfra node disk list --node 94d58604-6f30-4339-8578-adb7903b7277 \
-c id -c node_id -c device -c used -c size -c role
+----------------+----------------+--------+--------+-----------+------------+
| id             | node_id        | device | used   | size      | role       |
+----------------+----------------+--------+--------+-----------+------------+
| E0B7CE6F-<...> | 94d58604-<...> | sda    | 5.5GiB | 239.1GiB  | mds-system |
| EAC7DF5D-<...> | 94d58604-<...> | sdb    | 2.1GiB | 1007.8GiB | cs         |
| 49D792CA-<...> | 94d58604-<...> | sdc    | 2.1GiB | 1007.8GiB | cs         |
+----------------+----------------+--------+--------+-----------+------------+
```

This command lists disks on the node with the ID `94d58604-6f30-4339-8578-adb7903b7277`. (The output is abridged to fit on page.)

2.5.2 vinfra node disk show

Show details of a disk:

```
usage: vinfra node disk show --node <node> <disk>
```

`--node <node>`

Node ID or hostname

`<disk>`

Disk ID or device name

Example:

```
# vinfra node disk show EAC7DF5D-9E60-4444-85F7-5CA5738399CC \
--node 94d58604-6f30-4339-8578-adb7903b7277
+--------------------+--------------------------------------+
| Field              | Value                                |
+--------------------+--------------------------------------+
| being_released     | False                                |
| device             | sdb                                  |
| disk_status        | ok                                   |
| encryption         |                                      |
| id                 | EAC7DF5D-9E60-4444-85F7-5CA5738399CC |
| is_blink_available | False                                |
| is_blinking        | False                                |
| latency            |                                      |
| lun_id             |                                      |
| model              | Vz_HARDDISK2                         |
| mountpoint         | /vstorage/33aac2d5                   |
| node_id            | 94d58604-6f30-4339-8578-adb7903b7277 |
| role               | cs                                   |
| rpm                |                                      |
| serial_number      | 45589b5823ce4c188b55                 |
| service_id         | 1026                                 |
| service_params     | journal_type: inner_cache            |
|                    | tier: 0                              |
| service_status     | ok                                   |
| slot               |                                      |
| smart_status       | not_supported                        |
| space              | full_size: 1099511627776             |
|                    | size: 1082101518336                  |
|                    | used: 2246164480                     |
| tasks              |                                      |
| temperature        | 0.0                                  |
| transport          |                                      |
| type               | hdd                                  |
+--------------------+--------------------------------------+
```

This commands shows the details of the disk with the ID `EAC7DF5D-9E60-4444-85F7-5CA5738399CC` attached to

the node with the ID `94d58604-6f30-4339-8578-adb7903b7277`.

2.5.3 vinfra node disk assign

Add multiple disks to the storage cluster:

```
usage: vinfra node disk assign --disk <disk>:<role>[:<key=value,...>]
                               --node <node>
```

`--disk <disk>:<role>[:<key=value,...>]`

Disk configuration in the format:

- `<disk>`: disk device ID or name;
- `<role>`: disk role (`cs`, `mds`, `journal`, `mds-journal`, `mds-system`, `cs-system`, `system`);
- comma-separated `key=value` pairs with keys (optional):
  - `tier`: disk tier (0, 1, 2 or 3);
  - `journal-tier`: journal (cache) disk tier (0, 1, 2 or 3);
  - `journal-type`: journal (cache) disk type (`no_cache`, `inner_cache` or `external_cache`);
  - `journal-disk`: journal (cache) disk ID or device name;
  - `journal-size`: journal (cache) disk size, in bytes;
  - `bind-address`: bind IP address for the metadata service.

E.g., `sda:cs:tier=0,journal-type=inner_cache`. This option can be used multiple times.

`--node <node>`

Node ID or hostname

Example:

```
# vinfra node disk assign --disk sdc:cs --node f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | 080337ba-0508-44a0-9363-eddcd9df9f0d |
+---------+--------------------------------------+
```

This command creates a task to assign the role `cs` to the disk `sdc` on the node with the ID `f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4`

Task outcome:

```
# vinfra task show 080337ba-0508-44a0-9363-eddcd9df9f0d
+---------+-------------------------------------------------------------------------------+
| Field   | Value                                                                         |
+---------+-------------------------------------------------------------------------------+
| args    | []                                                                            |
| kwargs  | cluster_id: 1                                                                 |
|         | disks:                                                                        |
|         | - id: D3BEF4BB-AA3B-4DB6-9376-BC7CDA636700                                    |
|         |   role: cs                                                                    |
|         |   service_params: {}                                                          |
|         | logger:                                                                       |
|         |   __classname: backend.logger.tracer.TracingLogger                            |
|         |   __dict:                                                                     |
|         |     prefix: POST /api/v2/1/nodes/f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4/disks/ |
|         |     token: '3215629651314950'                                                 |
|         | node_id: f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4                                 |
| name    | backend.tasks.disks.BulkAssignDiskTask                                        |
| result  | {}                                                                            |
| state   | success                                                                       |
| task_id | 080337ba-0508-44a0-9363-eddcd9df9f0d                                          |
+---------+-------------------------------------------------------------------------------+
```

2.5.4 vinfra node disk release

Release a disk from the storage cluster. Start data migration from the node as well as cluster replication and rebalancing to meet the configured redundancy level:

```
usage: vinfra node disk release [--force] --node <node> <disk>
```

`--force`

Release without data migration

`--node <node>`

Node ID or hostname

`<disk>`

Disk ID or device name

Example:

```
# vinfra node disk release sdc --node f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | 587a936d-3953-481c-a2cd-b1223b890bec |
+---------+--------------------------------------+
```

This command creates a task to release the role `cs` from the disk `sdc` on the node with the ID

`f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4`.

Task outcome:

```
# vinfra task show 587a936d-3953-481c-a2cd-b1223b890bec
+---------+-----------------------------------------------------------------------------------------+
| Field   | Value                                                                                   |
+---------+-----------------------------------------------------------------------------------------+
| args    | []                                                                                      |
| kwargs  | cluster_id: 1                                                                           |
|         | disk_id: 43EF3400-EA95-43DE-B624-3D7ED0F9DDDD                                           |
|         | force: false                                                                            |
|         | logger:                                                                                 |
|         |   __classname: backend.logger.tracer.TracingLogger                                      |
|         |   __dict:                                                                               |
|         |     prefix: POST /api/v2/1/nodes/f59dabdb-                                              |
|         | bd1c-4944-8af2-26b8fe9ff8d4/disks/43EF3400-EA95-43DE-B624-3D7ED0F9DDDD/release/         |
|         |     token: '3217122839314940'                                                           |
|         | node_id: f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4                                           |
| name    | backend.tasks.disks.ReleaseDiskTask                                                     |
| state   | success                                                                                 |
| task_id | 587a936d-3953-481c-a2cd-b1223b890bec                                                    |
+---------+-----------------------------------------------------------------------------------------+
```

2.5.5 vinfra node disk blink

Start blinking the specified disk bay to identify disk for maintenance purposes:

```
usage: vinfra node disk blink on --node <node> <disk>
```

`--node <node>`

Node ID or hostname

`<disk>`

Disk ID or device name

Stop blinking the specified disk bay:

```
usage: vinfra node disk blink off --node <node> <disk>
```

`--node <node>`

Node ID or hostname

`<disk>`

Disk ID or device name

Example:

```
# vinfra node disk blink on sda --node f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4
```

This command starts blinking the disk `sda` on the node with the ID `f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4`.

```
# vinfra node disk blink off sda --node f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4
```

This command stops blinking the disk `sda` on the node with the ID `f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4`.

2.5.6 vinfra node iscsi target add

Add an iSCSI target as a disk to a node:

```
usage: vinfra node iscsi target add [--auth-username <auth-username>]
                                    [--auth-password <auth-password>]
                                    --portal <portal> --node <node> <target-name>
```

`--auth-username <auth-username>`

User name

`--auth-password <auth-password>`

User password

`--portal <portal>`

Portal IP address in the format `IP:port` (this option can be specified multiple times)

`--node <node>`

Node ID or hostname

`<target-name>`

Target name

Example:

```
# vinfra node iscsi target add iqn.2014-06.com.vstorage:target1 \
--portal 172.16.24.244:3260 --node f1931be7-0a01-4977-bfef-51a392adcd94
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | c42bfbe5-7292-41c2-91cb-446795535ab9 |
+---------+--------------------------------------+
```

This command creates a task to connect a remote ISCSI target `iqn.2014-06.com.vstorage:target1` with the IP address `172.16.24.244` and port `3260` to the node with the ID `f1931be7-0a01-4977-bfef-51a392adcd94`.

Task outcome:

```
# vinfra task show c42bfbe5-7292-41c2-91cb-446795535ab9
+---------+-------------------------------------------------------------+
| Field   | Value                                                       |
+---------+-------------------------------------------------------------+
| args    | - f1931be7-0a01-4977-bfef-51a392adcd94                      |
| kwargs  | portals:                                                    |
|         | - address: 172.16.24.244                                    |
|         |   port: 3260                                                |
|         | target_name: iqn.2014-06.com.vstorage:target1               |
| name    | backend.presentation.nodes.iscsi_initiators.tasks.ConnectTask |
| result  | connected: true                                             |
|         | portals:                                                    |
|         | - address: 172.16.24.244                                    |
|         |   port: 3260                                                |
|         | state: connected                                            |
|         | target_name: iqn.2014-06.com.vstorage:target1               |
| state   | success                                                     |
| task_id | c42bfbe5-7292-41c2-91cb-446795535ab9                        |
+---------+-------------------------------------------------------------+
```

2.5.7 vinfra node iscsi target delete

Delete an iSCSI target from a node:

```
usage: vinfra node iscsi target delete --node <node> <target-name>
```

`--node <node>`

Node ID or hostname

`<target-name>`

Target name

Example:

```
# vinfra node iscsi target delete iqn.2014-06.com.vstorage:target1 \
--node f1931be7-0a01-4977-bfef-51a392adcd94
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | c8dc74ee-86d6-4b89-8b6f-153ff1e78cb7 |
+---------+--------------------------------------+
```

This command creates a task to disconnect a remote ISCSI target `iqn.2014-06.com.vstorage:target1` from the node with the ID `f1931be7-0a01-4977-bfef-51a392adcd94`.

Task outcome:

```
# vinfra task show c8dc74ee-86d6-4b89-8b6f-153ff1e78cb7
+---------+-----------------------------------------------------------------+
| Field   | Value                                                           |
+---------+-----------------------------------------------------------------+
| args    | - f1931be7-0a01-4977-bfef-51a392adcd94                          |
| kwargs  | target_name: iqn.2014-06.com.vstorage:target1                   |
| name    | backend.presentation.nodes.iscsi_initiators.tasks.DisconnectTask |
| state   | success                                                         |
| task_id | c8dc74ee-86d6-4b89-8b6f-153ff1e78cb7                            |
+---------+-----------------------------------------------------------------+
```

2.6 vinfra cluster create

Create a storage cluster:

```
usage: vinfra cluster create [--disk <disk>:<role>[:<key=value,...>]]
                             [--tier-encryption {0,1,2,3}] --node <node> <cluster-name>
```

`--disk <disk>:<role> [:<key=value,...>]`

Disk configuration in the format:

- `<disk>`: disk device ID or name;
- `<role>`: disk role (`cs`, `mds`, `journal`, `mds-journal`, `mds-system`, `cs-system`, `system`);
- comma-separated `key=value` pairs with keys (optional):
  - `tier`: disk tier (0, 1, 2 or 3);
  - `journal-tier`: journal (cache) disk tier (0, 1, 2 or 3);
  - `journal-type`: journal (cache) disk type (`no_cache`, `inner_cache` or `external_cache`);
  - `journal-disk`: journal (cache) disk ID or device name;
  - `journal-size`: journal (cache) disk size, in bytes;
  - `bind-address`: bind IP address for the metadata service.

E.g., `sda:cs:tier=0,journal-type=inner_cache`. This option can be used multiple times.

`--tier-encryption {0,1,2,3}`

Enable encryption for storage cluster tiers. Encryption is disabled by default. This option can be used multiple times.

`--node <node>`

Node ID or hostname

`<cluster-name>`

Storage cluster name

Example:

```
# vinfra cluster create stor1 --node 94d58604-6f30-4339-8578-adb7903b7277
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | d9ca8e1d-8ac8-4459-898b-2d803efd7bc6 |
+---------+--------------------------------------+
```

This command creates a task to create the storage cluster `stor1` on the node with the ID `94d58604-6f30-4339-8578-adb7903b7277`. As disk roles are not explicitly specified, they are assigned automatically: `mds-system` to the system disk, and `cs` to all other disks.

Task outcome:

```
# vinfra task show d9ca8e1d-8ac8-4459-898b-2d803efd7bc6
+---------+----------------------------------------+
| Field   | Value                                  |
+---------+----------------------------------------+
| args    | - stor1                                |
|         | - 94d58604-6f30-4339-8578-adb7903b7277 |
|         | - null                                 |
|         | - null                                 |
| kwargs  | {}                                     |
| name    | backend.tasks.cluster.CreateNewCluster |
| result  | cluster_id: 1                          |
| state   | success                                |
| task_id | d9ca8e1d-8ac8-4459-898b-2d803efd7bc6   |
+---------+----------------------------------------+
```

2.7 Showing Storage Cluster Overview and Details

2.7.1 vinfra cluster overview

Show storage cluster overview:

```
usage: vinfra cluster overview
```

Example:

```
# vinfra cluster overview
+-------------------+--------------------------+
| Field             | Value                    |
+-------------------+--------------------------+
| chunks            | blocked: 0               |
|                   | degraded: 0              |
|                   | deleting: 0              |
|                   | healthy: 2               |
|                   | offline: 0               |
|                   | overcommitted: 0         |
|                   | pending: 0               |
|                   | replicating: 0           |
|                   | standby: 0               |
|                   | total: 2                 |
|                   | unique: 2                |
|                   | urgent: 0                |
|                   | void: 0                  |
| fs_stat           | chunk_maps: 2            |
|                   | chunk_nodes: 2           |
|                   | file_maps: 2             |
|                   | files: 9                 |
|                   | inodes: 9                |
|                   | used_size: 11335680      |
| id                | 1                        |
| license           | capacity: 1099511627776  |
|                   | expiration_ts: null      |
|                   | keynumber: null          |
|                   | status: 0                |
|                   | used_size: 11335680      |
| logic_space       | free: 1099500292096      |
|                   | total: 1099511627776     |
|                   | used: 11335680           |
| name              | stor1                    |
| repl              | eta: null                |
|                   | reads: 0                 |
|                   | writes: 0                |
| resistance        | to_lose: 0               |
|                   | total: 1                 |
| space_per_service | abgw: null               |
|                   | compute: null            |
|                   | iscsi: null              |
|                   | nfs: null                |
|                   | s3: null                 |
| status            | healthy                  |
| tiers             | - id: 0                  |
|                   |   phys_space:            |
|                   |     free: 2164191700992  |
|                   |     total: 2164203036672 |
```

```
|                   |     used: 11335680       |
+-------------------+--------------------------+
```

This command shows an overview of the cluster.

2.7.2 vinfra cluster show

Show cluster details:

```
usage: vinfra cluster show
```

Example:

```
# vinfra cluster show
+--------+-------------------------------------------+
| Field  | Value                                     |
+--------+-------------------------------------------+
| id     | 1                                         |
| name   | stor1                                     |
| nodes  | - host: stor-4.example.com.vstoragedomain  |
|        |   id: 4b83a87d-9adf-472c-91f0-782c47b2d5f1 |
|        |   is_installing: false                    |
|        |   is_releasing: false                     |
|        | - host: stor-3.example.com.vstoragedomain  |
|        |   id: 7d7d37b8-4c06-4f1a-b3a6-4b54257d70ce |
|        |   is_installing: false                    |
|        |   is_releasing: false                     |
|        | - host: stor-5.example.com.vstoragedomain  |
|        |   id: fd1e46de-6e17-4571-bf6b-1ac34ec1c225 |
|        |   is_installing: false                    |
|        |   is_releasing: false                     |
|        | - host: stor-1.example.com.vstoragedomain  |
|        |   id: 94d58604-6f30-4339-8578-adb7903b7277 |
|        |   is_installing: false                    |
|        |   is_releasing: false                     |
|        | - host: stor-2.example.com.vstoragedomain  |
|        |   id: f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4 |
|        |   is_installing: false                    |
|        |   is_releasing: false                     |
+--------+-------------------------------------------+
```

This command shows cluster details.

2.8 vinfra cluster delete

Delete the storage cluster:

```
usage: vinfra cluster delete
```

Example:

```
# vinfra cluster delete
Operation waiting (timeout=600s) [Elapsed Time: 0:01:09] ... |
Operation successful
```

This command releases all nodes from the storage cluster.

CHAPTER 3

Managing Compute Cluster

3.1 vinfra service compute cluster create

Create a compute cluster:

```
vinfra service compute cluster create [--public-network <network>]
                                      [--subnet cidr=CIDR[,key=value,...]]
                                      [--cpu-model <cpu-model>] [--force]
                                      --nodes <nodes>
```

`--public-network <network>`

A physical network to connect the public virtual network to. It must include the 'VM public' traffic type.

`--subnet cidr=CIDR[,key=value,...]`

Subnet for IP address management in the public virtual network (the `--public-network` option is required):

- `cidr`: subnet range in CIDR notation;
- comma-separated `key=value` pairs with keys (optional):
  - `gateway`: gateway IP address;
  - `dhcp`: enable/disable the virtual DHCP server;
  - `allocation-pool`: allocation pool of IP addresses from CIDR in the format `ip1-ip2`, where `ip1` and `ip2` are starting and ending IP addresses correspondingly. Specify the key multiple times to create multiple IP pools;
  - `dns-server`: DNS server IP address, specify multiple times to set multiple DNS servers.

Example: `--subnet cidr=192.168.5.0/24,dhcp=enable`.

`--cpu-model <cpu-model>`

CPU model for virtual machines.

`--force`

Skip checks for minimal hardware requirements.

`--nodes <nodes>`

A comma-separated list of node IDs or hostnames.

Example:

```
# vinfra service compute cluster create --virtual-ip 10.94.50.244 \
--node 7ffa9540-5a20-41d1-b203-e3f349d62565,\
02ff64ae-5800-4090-b958-18b1fe8f5060,\
6e8afc28-7f71-4848-bdbe-7c5de64c5013,\
37c70bfb-c289-4794-8be4-b7a40c2b6d95,\
827a1f4e-56e5-404f-9113-88748c18f0c2 --enable-nested \
--public-network Public --subnet cidr=10.94.0.0/16,dhcp=enable,\
gateway=10.94.0.1,allocation-pool=10.94.129.64-10.94.129.79,\
dns-server=10.30.0.27,dns-server=10.30.0.28
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | be517afa-fae0-457e-819c-f4d6399f3ae2 |
+---------+--------------------------------------+
```

This command creates a task to create the compute cluster from five nodes specified by ID. It also specifies the virtual IP address (must belong to the network with the `Compute API` traffic type), the public network for VMs, the gateway, the allocation pool of IP addresses to assign to VMs, and the DNS servers to use.

Task outcome:

```
# vinfra task show be517afa-fae0-457e-819c-f4d6399f3ae2
+----------+-------------------------------------------------------------------+
| Field    | Value                                                             |
+----------+-------------------------------------------------------------------+
| args     | - admin                                                           |
| kwargs   | enable_nested: true                                               |
|          | external_network:                                                 |
|          |   roles_set_id: dd42723e-1318-4f8f-9a43-b303ab09cbbe              |
|          |   subnet:                                                         |
|          |     allocation_pools:                                             |
|          |     - end_address: 10.94.129.79                                   |
|          |       start_address: 10.94.129.64                                 |
|          |     cidr: 10.94.0.0/16                                            |
|          |     dns_servers:                                                  |
|          |     - 10.30.0.27                                                  |
|          |     - 10.30.0.28                                                  |
|          |     enable_dhcp: true                                             |
|          |     gateway: 10.94.0.1                                            |
```

```
|          | nodes:                                                          |
|          | - 7ffa9540-5a20-41d1-b203-e3f349d62565                          |
|          | - 02ff64ae-5800-4090-b958-18b1fe8f5060                          |
|          | - 6e8afc28-7f71-4848-bdbe-7c5de64c5013                          |
|          | - 37c70bfb-c289-4794-8be4-b7a40c2b6d95                          |
|          | - 827a1f4e-56e5-404f-9113-88748c18f0c2                          |
| name     | backend.presentation.compute.tasks.DeployComputeClusterTask     |
| progress | 100                                                             |
| state    | success                                                         |
| task_id  | be517afa-fae0-457e-819c-f4d6399f3ae2                            |
+----------+-----------------------------------------------------------------+
```

3.2 Showing Compute Cluster Details and Overview

3.2.1 vinfra service compute cluster show

Display compute cluster details:

```
usage: vinfra service compute cluster show
```

Example:

```
# vinfra service compute cluster show
+--------------+-------------------------------------------------+
| Field        | Value                                           |
+--------------+-------------------------------------------------+
| capabilities | cpu_models:                                     |
|              | - SandyBridge                                   |
|              | - IvyBridge                                     |
|              | - Haswell                                       |
|              | - Haswell-noTSX                                 |
|              | - Broadwell                                     |
|              | - Broadwell-noTSX                               |
|              | - Skylake-Client                                |
|              | - Skylake-Server                                |
|              | - HostPassthrough                               |
|              | os_distributions:                               |
|              |   linux:                                        |
|              |     centos6: Centos 6                           |
|              |     centos7: Centos 7                           |
|              |     debian9: Debian 9 (Stretch)                 |
|              |     linux: Generic Linux                        |
|              |     rhel7: Red Hat Enterprise Linux             |
|              |     ubuntu16.04: Ubuntu 16.04 LTS               |
|              |     ubuntu18.04: Ubuntu 18.04 LTS               |
```

```
|              |   windows:                                          |
|              |     win10: Microsoft Windows 10                     |
|              |     win2k12: Microsoft Windows Server 2012           |
|              |     win2k12r2: Microsoft Windows Server 2012 R2      |
|              |     win2k16: Microsoft Windows Server 2016           |
|              |     win2k8r2: Microsoft Windows Server 2008 R2       |
|              |     win7: Microsoft Windows 7                       |
|              |     win8.1: Microsoft Windows 8.1                   |
|              |     windows: Generic Windows                        |
| options      | cpu_model: null                                     |
| status       | active                                              |
+--------------+-----------------------------------------------------+
```

This command shows the status and capabilities of the compute cluster.

3.2.2 vinfra service compute cluster stat

Display compute cluster statistics:

```
usage: vinfra service compute cluster stat
```

Example:

```
# vinfra service compute cluster stat
+----------+------------------------------+
| Field    | Value                        |
+----------+------------------------------+
| compute  | block_capacity: 0            |
|          | block_usage: 0               |
|          | cpu_usage: 0.0               |
|          | mem_total: 0                 |
|          | mem_usage: 0                 |
|          | vcpus: 0                     |
| datetime | 2018-09-11T15:50:18.758258   |
| physical | block_capacity: 1099511627776 |
|          | block_free: 1099498911464    |
|          | cpu_cores: 10                |
|          | mem_total: 41006247936       |
| reserved | cpus: 5                      |
|          | memory: 17721982976          |
| servers  | count: 0                     |
|          | error: 0                     |
|          | in_progress: 0               |
|          | running: 0                   |
|          | stopped: 0                   |
|          | top:                         |
|          |   disk: []                   |
|          |   memory: []                 |
|          |   vcpus: []                  |
```

```
+----------+------------------------------+
```

This command shows the overview of the compute cluster.

3.3 vinfra service compute cluster set

Change compute cluster parameters:

```
vinfra service compute cluster set [--cpu-model <cpu-model>]
```

`--cpu-model <cpu-model>`

Set the default CPU model for virtual machines (`SandyBridge`, `IvyBridge`, `Haswell`, `Haswell-noTSX`, `Broadwell`, `Broadwell-noTSX`, `Skylake-Client`, `Skylake-Server`, `HostPassthrough`)

Example:

```
# vinfra service compute cluster set --cpu-model Haswell
```

This command sets the default CPU model for VMs to `Haswell`.

3.4 Managing Compute Nodes

3.4.1 vinfra service compute node add

Add a node to the compute cluster:

```
usage: vinfra service compute node add [--compute] [--controller] [--force] <node-id>
```

`--compute`

Compute node role

`--controller`

Compute controller node role

`--force`

Skip checks for minimal hardware requirements

`<node-id>`

ID or hostname of the compute node

Example:

```
# vinfra service compute node add --node 827a1f4e-56e5-404f-9113-88748c18f0c2
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | 4c58e63c-31b6-406a-8070-9197445ec794 |
+---------+--------------------------------------+
```

This command creates a task to add the node with the ID `827a1f4e-56e5-404f-9113-88748c18f0c2` to the compute cluster.

Task outcome:

```
# vinfra task show 4c58e63c-31b6-406a-8070-9197445ec794
+----------+-----------------------------------------------------------------+
| Field    | Value                                                           |
+----------+-----------------------------------------------------------------+
| args     | []                                                              |
| kwargs   | nodes:                                                          |
|          | - 827a1f4e-56e5-404f-9113-88748c18f0c2                          |
| name     | backend.presentation.compute.tasks.ComputeClusterAddNodesTask |
| progress | 100                                                             |
| state    | success                                                         |
| task_id  | 4c58e63c-31b6-406a-8070-9197445ec794                            |
+----------+-----------------------------------------------------------------+
```

3.4.2 vinfra service compute node list

List compute nodes:

```
usage: vinfra service compute node list
```

Example:

```
# vinfra service compute node list
+--------------------------------------+-----------------------------------+-------+-----+
| id                                   | hypervisor_hostname               | state | vms |
+--------------------------------------+-----------------------------------+-------+-----+
| 7ffa9540-5a20-41d1-b203-e3f349d62565 | stor-1.example.com.vstoragedomain | up    |   1 |
| 6e8afc28-7f71-4848-bdbe-7c5de64c5013 | stor-3.example.com.vstoragedomain | up    |   1 |
| 02ff64ae-5800-4090-b958-18b1fe8f5060 | stor-2.example.com.vstoragedomain | up    |   1 |
| 827a1f4e-56e5-404f-9113-88748c18f0c2 | stor-5.example.com.vstoragedomain | up    |   0 |
| 37c70bfb-c289-4794-8be4-b7a40c2b6d95 | stor-4.example.com.vstoragedomain | up    |   1 |
+--------------------------------------+-----------------------------------+-------+-----+
```

This command lists nodes in the compute cluster.

3.4.3 Showing Compute Node Details

Display compute node details:

```
usage: vinfra service compute node show <node>
```

<node>

Node ID or hostname

Example:

```
# vinfra service compute node show 7ffa9540-5a20-41d1-b203-e3f349d62565
+--------------------+----------------------------------------------+
| Field              | Value                                        |
+--------------------+----------------------------------------------+
| host_ip            | 10.37.130.101                                |
| hypervisor_hostname | stor-1.example.com.vstoragedomain           |
| hypervisor_id      | 1                                            |
| id                 | 7ffa9540-5a20-41d1-b203-e3f349d62565         |
| state              | up                                           |
| statistics         | compute:                                     |
|                    |   block_capacity: 0                          |
|                    |   block_usage: 0                             |
|                    |   cpu_usage: 0                               |
|                    |   mem_total: 0                               |
|                    |   mem_usage: 0                               |
|                    |   vcpus: 0                                   |
|                    | datetime: '2018-09-11T16:39:15.290999+00:00' |
|                    | physical:                                    |
|                    |   cpu_cores: 2                               |
|                    |   mem_free: 414105600                        |
|                    |   mem_total: 8201244672                      |
|                    | reserved:                                    |
|                    |   cpus: 1                                    |
|                    |   memory: 5773                               |
| vms                | 0                                            |
+--------------------+----------------------------------------------+
```

This command shows the details of the compute node with the ID `7ffa9540-5a20-41d1-b203-e3f349d62565`.

3.4.4 vinfra service compute node fence

Fence a compute node:

```
usage: vinfra service compute node fence <node>
```

<node>

Node ID or hostname

Example:

```
# vinfra service compute node fence e6255aed-d6e7-41b2-ba90-86164c1cd9a6
Operation successful
```

This command fences the node with the ID `e6255aed-d6e7-41b2-ba90-86164c1cd9a6`.

3.4.5 vinfra service compute node unfence

Unfence a compute node:

```
usage: vinfra service compute node unfence <node>
```

`<node>`

Node ID or hostname

Example:

```
# vinfra service compute node unfence e6255aed-d6e7-41b2-ba90-86164c1cd9a6
Operation successful
```

This command unfences the node with the ID `e6255aed-d6e7-41b2-ba90-86164c1cd9a6`.

3.4.6 vinfra service compute node release

Release a node from the compute cluster:

```
usage: vinfra service compute node release [--compute] [--controller] <node-id>
```

`--compute`

Compute node role

`--controller`

Compute controller node role

`<node-id>`

ID or hostname of the compute node

Example:

```
# vinfra service compute node release --node 827a1f4e-56e5-404f-9113-88748c18f0c2
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
```

```
| task_id | 3b39738c-80a6-40a6-a50d-c3c8118ed212 |
+---------+--------------------------------------+
```

This command creates a task to release the node with the ID `827a1f4e-56e5-404f-9113-88748c18f0c2` from the compute cluster.

Task outcome:

```
# vinfra task show 3b39738c-80a6-40a6-a50d-c3c8118ed212
+---------+-------------------------------------------------------------------+
| Field   | Value                                                             |
+---------+-------------------------------------------------------------------+
| args    | []                                                                |
| kwargs  | nodes:                                                            |
|         | - 827a1f4e-56e5-404f-9113-88748c18f0c2                            |
| name    | backend.presentation.compute.tasks.ComputeClusterDeleteNodesTask  |
| state   | success                                                           |
| task_id | 3b39738c-80a6-40a6-a50d-c3c8118ed212                              |
+---------+-------------------------------------------------------------------+
```

3.5 Managing Networks

3.5.1 vinfra service compute network create

Create a compute network:

```
usage: vinfra service compute network create [--dhcp | --no-dhcp] [--type {vxlan|flat}]
                                             [--dns-nameserver <dns-nameserver>]
                                             [--allocation-pool <allocation-pool>]
                                             [--gateway <gateway> | --no-gateway]
                                             [--ip-version <ip-version>]
                                             [--physical-network <physical-network>]
                                             [--cidr <cidr>] <network-name>
```

`--dhcp`

Enable DHCP

`--no-dhcp`

Disable DHCP

`--dns-nameserver <dns-nameserver>`

DNS server IP address. This option can be used multiple times.

`--allocation-pool <allocation-pool>`

Allocation pool to create inside the network in the format: `ip_addr_start-ip_addr_end`. This option can

be used multiple times.

`--gateway <gateway>`

Gateway IP address

`--no-gateway`

Do not configure a gateway for this network

`--ip-version <ip-version>`

Network IP version

`--type {vxlan|flat}`

Virtual network type (`vxlan` is private and `flat` is public)

`--physical-network <physical-network>`

A physical network to link to a flat network

`--cidr <cidr>`

Subnet range in CIDR notation

`<network-name>`

Network name

Example:

```
# vinfra service compute network create myprivnet --type vxlan \
--cidr 192.128.128.0/24 --gateway 192.128.128.1
+------------------+--------------------------------------+
| Field            | Value                                |
+------------------+--------------------------------------+
| id               | 3848fb5d-bc98-4320-acd0-cde2df7c5bdd |
| name             | myprivnet                            |
| physical_network |                                      |
| project_id       | 72a5db3a033c403a86756021e601ef34     |
| subnet           | allocation_pools:                    |
|                  | - end: 192.128.128.254               |
|                  |   start: 192.128.128.2               |
|                  | cidr: 192.128.128.0/24               |
|                  | dns_nameservers: []                  |
|                  | enable_dhcp: true                    |
|                  | gateway_ip: 192.128.128.1            |
|                  | ip_version: 4                        |
| type             | vxlan                                |
+------------------+--------------------------------------+
```

This command creates a private network `myprivnet` with the specific CIDR and gateway.

3.5.2 vinfra service compute network list

List compute networks:

```
usage: vinfra service compute network list
```

Example:

```
# vinfra service compute network list -c id -c name -c cidr -c type -c allocation_pools
+----------------+-----------+-------+------------------+------------------------+
| id             | name      | type  | cidr             | allocation_pools       |
+----------------+-----------+-------+------------------+------------------------+
| 1bf2c9da-<...> | private   | vxlan | 192.168.128.0/24 | - end: 192.168.128.254 |
|                |           |       |                  |   start: 192.168.128.2 |
| 3848fb5d-<...> | myprivnet | vxlan | 192.128.128.0/24 | - end: 192.128.128.254 |
|                |           |       |                  |   start: 192.128.128.2 |
| 417606ac-<...> | public    | flat  | 10.94.0.0/16     | - end: 10.94.129.79    |
|                |           |       |                  |   start: 10.94.129.64  |
+----------------+-----------+-------+------------------+------------------------+
```

This command lists networks used in the compute cluster. (The output is abridged to fit on page.)

3.5.3 vinfra service compute network show

Display compute network details:

```
usage: vinfra service compute network show <network>
```

<network>

Network ID or name

Example:

```
# vinfra service compute network show 417606ac-1dbe-426a-844d-e047831ddce9
+-----------------+--------------------------------------+
| Field           | Value                                |
+-----------------+--------------------------------------+
| allocation_pools |                                     |
| cidr            |                                      |
| dns_nameservers |                                      |
| enable_dhcp     |                                      |
| gateway_ip      |                                      |
| id              | 417606ac-1dbe-426a-844d-e047831ddce9 |
| ip_version      |                                      |
| name            | public                               |
| physical_network | Public                              |
| project_id      | 72a5db3a033c403a86756021e601ef34     |
```

```
| type              | flat                                 |
+-------------------+--------------------------------------+
```

This command shows the details of the network with the ID `417606ac-1dbe-426a-844d-e047831ddce9`.

3.5.4 vinfra service compute network set

Modify compute network parameters:

```
usage: vinfra service compute network set [--dhcp | --no-dhcp]
                                          [--dns-nameserver <dns-nameserver>]
                                          [--allocation-pool <allocation-pool>]
                                          [--gateway <gateway> | --no-gateway]
                                          [--name <name>] <network>
```

`--dhcp`

Enable DHCP

`--no-dhcp`

Disable DHCP

`--dns-nameserver <dns-nameserver>`

DNS server IP address. This option can be used multiple times.

`--allocation-pool <allocation-pool>`

Allocation pool to create inside the network in the format: `ip_addr_start-ip_addr_end`. This option can be used multiple times.

`--gateway <gateway>`

Gateway IP address

`--no-gateway`

Do not configure a gateway for this network

`--name <name>`

A new name for the network

`<network>`

Network ID or name

Example:

```
# vinfra service compute network set myprivnet --no-dhcp
+-------------------+--------------------------------------+
```

```
| Field            | Value                                |
+------------------+--------------------------------------+
| id               | 3848fb5d-bc98-4320-acd0-cde2df7c5bdd |
| name             | myprivnet                            |
| physical_network |                                      |
| project_id       | 72a5db3a033c403a86756021e601ef34     |
| subnet           | allocation_pools:                    |
|                  | - end: 192.128.128.254               |
|                  |   start: 192.128.128.2               |
|                  | cidr: 192.128.128.0/24               |
|                  | dns_nameservers: []                  |
|                  | enable_dhcp: false                   |
|                  | gateway_ip: 192.128.128.1            |
|                  | ip_version: 4                        |
| type             | vxlan                                |
+------------------+--------------------------------------+
```

This command disables DHCP for the private network `myprivnet`.

3.5.5 vinfra service compute network delete

Delete a compute network:

```
usage: vinfra service compute network delete <network>
```

`<network>`

Network ID or name

Example:

```
# vinfra service compute network delete myprivnet
Operation successful
```

This command deletes the private network `myprivnet`.

3.6 Managing Images

3.6.1 vinfra service compute image create

Create a new compute image:

```
usage: vinfra service compute image create [--min-disk <size-gb>] [--min-ram <size-mb>]
                                           [--os-distro <os-distro>] [--protected]
                                           [--disk-format <disk_format>]
```

```
                                        [--container-format <format>]
                                        --file <file> <image-name>
```

`--min-disk <size-gb>`

Minimum disk size required to boot from image, in gigabytes

`--min-ram <size-mb>`

Minimum RAM size required to boot from image, in megabytes

`--os-distro <os-distro>`

OS distribution. To list available distributions, run `service compute cluster show`.

`--protected`

Protect image from deletion

`--disk-format <disk_format>`

Disk format `aki`, `ami`, `ari`, `detect`, `iso`, `ploop`, `qcow2`, `raw`, `vdi`, `vhd`, `vhdx`, `vmdk` (default: `detect`)

`--container-format <format>`

Container format: `aki`, `ami`, `ari`, `bare`, `docker`, `ovf`, `ova` (default: `bare`)

`--file <file>`

Create image from a local file

`<image-name>`

Image name

Example:

```
# vinfra service compute image create mycirrosimg \
--file /distr/cirros-0.4.0-x86_64-disk.img
Uploading image to server [elapsed time: 0:00:04]... |
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | 03874663-d03f-4891-a10b-64837e7faf43 |
+---------+--------------------------------------+
```

This command creates a task to create a Cirros image from the local file and upload it to Acronis Software-Defined Infrastructure.

Task outcome:

```
# vinfra task show 03874663-d03f-4891-a10b-64837e7faf43
+---------+------------------------------------------------------------------+
| Field   | Value                                                            |
+---------+------------------------------------------------------------------+
```

```
| details |                                                                    |
| name    | backend.presentation.compute.images.tasks.ImportComputeImageTask |
| result  | id: 179f45ef-c5d6-4270-b0c0-085b542544c5                           |
| state   | success                                                            |
| task_id | 03874663-d03f-4891-a10b-64837e7faf43                               |
+---------+--------------------------------------------------------------------+
```

3.6.2 vinfra service compute image list

List compute images:

```
usage: vinfra service compute image list
```

Example:

This command lists images available to the compute cluster.

```
# vinfra service compute image list
+--------------------------------------+-------------+----------+--------+-------------+
| id                                   | name        |     size | status | disk_format |
+--------------------------------------+-------------+----------+--------+-------------+
| 179f45ef-c5d6-4270-b0c0-085b542544c5 | mycirrosimg | 12716032 | active | qcow2       |
| 4741274f-5cca-4205-8f66-a2e89fb346cc | cirros      | 12716032 | active | qcow2       |
+--------------------------------------+-------------+----------+--------+-------------+
```

3.6.3 vinfra service compute image show

Display compute image details:

```
usage: vinfra service compute image show <image>
```

<image>

Image ID or name

Example:

```
# vinfra service compute image show 4741274f-5cca-4205-8f66-a2e89fb346cc
+------------------+------------------------------------------------------------------+
| Field            | Value                                                            |
+------------------+------------------------------------------------------------------+
| checksum         | 443b7623e27ecf03dc9e01ee93f67afe                                 |
| container_format | bare                                                             |
| created_at       | 2018-09-11T13:29:10Z                                             |
| disk_format      | qcow2                                                            |
| file             | /api/v2/compute/images/4741274f-5cca-4205-8f66-a2e89fb346cc/file/ |
| id               | 4741274f-5cca-4205-8f66-a2e89fb346cc                             |
```

```
| min_disk         | 1                                                                    |
| min_ram          | 0                                                                    |
| name             | cirros                                                               |
| os_distro        | linux                                                                |
| os_type          | linux                                                                |
| project_id       | 72a5db3a033c403a86756021e601ef34                                     |
| protected        | False                                                                |
| size             | 12716032                                                             |
| status           | active                                                               |
| tags             | []                                                                   |
| updated_at       | 2018-09-11T13:29:13Z                                                 |
| virtual_size     |                                                                      |
| visibility       | public                                                               |
+------------------+----------------------------------------------------------------------+
```

This command shows the details of the default Cirros image.

3.6.4 vinfra service compute image set

Modify compute image parameters:

```
usage: vinfra service compute image set [--min-disk <size-gb>] [--min-ram <size-mb>]
                                        [--os-distro <os-distro>] [--protected]
                                        [--name <name>] <image>
```

`--min-disk <size-gb>`

Minimum disk size required to boot from image, in gigabytes

`--min-ram <size-mb>`

Minimum RAM size required to boot from image, in megabytes

`--os-distro <os-distro>`

OS distribution. To list available distributions, run `service compute cluster show`.

`--protected`

Protect image from deletion

`--name <name>`

Image name

`<image>`

Image ID or name

Example:

```
# vinfra service compute image set 4741274f-5cca-4205-8f66-a2e89fb346cc --protected --min-ram 1
+------------------+------------------------------------------------------------------------+
| Field            | Value                                                                  |
+------------------+------------------------------------------------------------------------+
| checksum         | 443b7623e27ecf03dc9e01ee93f67afe                                       |
| container_format | bare                                                                   |
| created_at       | 2018-09-11T13:29:10Z                                                   |
| disk_format      | qcow2                                                                  |
| file             | /api/v2/compute/images/4741274f-5cca-4205-8f66-a2e89fb346cc/file/      |
| id               | 4741274f-5cca-4205-8f66-a2e89fb346cc                                   |
| min_disk         | 1                                                                      |
| min_ram          | 1                                                                      |
| name             | cirros                                                                 |
| os_distro        | linux                                                                  |
| os_type          | linux                                                                  |
| project_id       | 72a5db3a033c403a86756021e601ef34                                       |
| protected        | True                                                                   |
| size             | 12716032                                                               |
| status           | active                                                                 |
| tags             | []                                                                     |
| updated_at       | 2018-09-12T09:26:29Z                                                   |
| virtual_size     |                                                                        |
| visibility       | public                                                                 |
+------------------+------------------------------------------------------------------------+
```

This command protects the default Cirros image and sets the minimum RAM size for it to 1 GB.

3.6.5 vinfra service compute image save

Download a compute image:

```
usage: vinfra service compute image save [--file <filename>] <image>
```

`--file <filename>`

File to save the image to (default: `stdout`)

`<image>`

Image ID or name

Example:

```
# vinfra service compute image save 4741274f-5cca-4205-8f66-a2e89fb346cc --file cirros.qcow2
Operation successful
```

This command downloads the default Cirros image to the local disk as `cirros.qcow2`.

3.6.6 vinfra service compute image delete

Delete a compute image:

```
usage: vinfra service compute image delete <image>
```

`<image>`

Image ID or name

Example:

```
# vinfra service compute image delete 179f45ef-c5d6-4270-b0c0-085b542544c5
Operation successful
```

This command deletes the image with the ID `179f45ef-c5d6-4270-b0c0-085b542544c5`.

3.7 Managing Flavors

3.7.1 vinfra service compute flavor create

Create a new compute flavor:

```
usage: vinfra service compute flavor create [--swap <size-mb>] --vcpus <vcpus>
                                            --ram <size-mb> <flavor-name>
```

`--swap <size-mb>`

Swap space size, in megabytes

`--vcpus <vcpus>`

Number of virtual CPUs

`--ram <size-mb>`

Memory size, in megabytes

`<flavor-name>`

Flavor name

Example:

```
# vinfra service compute flavor create myflavor --vcpus 1 --ram 3072
+-------+--------------------------------------+
| Field | Value                                |
+-------+--------------------------------------+
```

```
| id    | 561a48ea-0c1c-4152-8b7d-e4b4af276c2d |
| name  | myflavor                             |
| ram   | 3072                                 |
| swap  | 0                                    |
| vcpus | 1                                    |
+-------+--------------------------------------+
```

This command creates a flavor `myflavor` with 1 vCPU and 3 GB RAM.

3.7.2 vinfra service compute flavor list

List compute flavors:

```
usage: vinfra service compute flavor list
```

Example:

```
# vinfra service compute flavor list
+--------------------------------------+----------+-------+------+-------+
| id                                   | name     |   ram | swap | vcpus |
+--------------------------------------+----------+-------+------+-------+
| 100                                  | tiny     |   512 |    0 |     1 |
| 101                                  | small    |  2048 |    0 |     1 |
| 102                                  | medium   |  4096 |    0 |     2 |
| 103                                  | large    |  8192 |    0 |     4 |
| 104                                  | xlarge   | 16384 |    0 |     8 |
| 561a48ea-0c1c-4152-8b7d-e4b4af276c2d | myflavor |  3072 |    0 |     1 |
+--------------------------------------+----------+-------+------+-------+
```

This command lists all flavors.

3.7.3 vinfra service compute flavor show

Display compute flavor details:

```
usage: vinfra service compute flavor show <flavor>
```

`<flavor>`

Flavor ID or name

Example:

```
# vinfra service compute flavor show myflavor
+-------+--------------------------------------+
| Field | Value                                |
+-------+--------------------------------------+
```

```
| id    | 561a48ea-0c1c-4152-8b7d-e4b4af276c2d |
| name  | myflavor                             |
| ram   | 3072                                 |
| swap  | 0                                    |
| vcpus | 1                                    |
+-------+--------------------------------------+
```

This command shows the details of the flavor `myflavor`.

3.7.4 vinfra service compute flavor delete

Delete a compute flavor:

```
usage: vinfra service compute flavor delete <flavor>
```

`<flavor>`

Flavor ID or name

Example:

```
# vinfra service compute flavor delete myflavor
Operation successful
```

This command deletes the flavor `myflavor`.

3.8 Managing Storage Policies

You can manage storage policies only after creating the compute cluster.

3.8.1 vinfra cluster storage-policy create

Create a new storage policy:

```
usage: vinfra cluster storage-policy create --tier {0,1,2,3}
                                            (--replicas <norm>[:<min>] |
                                            --encoding <M>+<N>) --failure-domain
                                            {disk,host,rack,row,room} <name>
```

`--tier {0,1,2,3}`

Storage tier

`--replicas <norm>[:<min>]`

Storage replication mapping in the format:

- `norm`: the number of replicas to maintain;
- `min`: the minimum required number of replicas (optional).

`--encoding <M>+<N>`

Storage erasure encoding mapping in the format:

- `M`: the number of data blocks;
- `N`: the number of parity blocks.

`--failure-domain {disk,host,rack,row,room}`

Storage failure domain

`<name>`

Storage policy name

Example:

```
# vinfra cluster storage-policy create mystorpolicy --tier 3 \
--encoding 3+2 --failure-domain host
+----------------+--------------------------------------+
| Field          | Value                                |
+----------------+--------------------------------------+
| failure_domain | host                                 |
| id             | 2199e71e-ce8a-4ba9-81cd-75502f0344ca |
| name           | mystorpolicy                         |
| redundancy     | encoding=3+2                         |
| tier           | 3                                    |
+----------------+--------------------------------------+
```

This command creates a storage policy `mystorpolicy` with the tier set to 3, redundancy scheme to erasure coding 3+2, and failure domain set to host.

3.8.2 vinfra cluster storage-policy list

List existing storage policies:

```
usage: vinfra cluster storage-policy list
```

Example:

```
# vinfra cluster storage-policy list
```

```
+----------------+-------------+------+--------------+----------------+
| id             | name        | tier | redundancy   | failure_domain |
+----------------+-------------+------+--------------+----------------+
| 2199e71e-<...> | mystorpolicy |    3 | encoding=3+2 | host           |
| 4274d6fd-<...> | default      |    0 | replicas=3   | host           |
+----------------+-------------+------+--------------+----------------+
```

This command lists storage policies available to the compute cluster.

3.8.3 vinfra cluster storage-policy show

Show details of a storage policy:

```
usage: vinfra cluster storage-policy show <storage-policy>
```

`<storage-policy>`

Storage policy ID or name

Example:

```
# vinfra cluster storage-policy show mystorpolicy
+----------------+--------------------------------------+
| Field          | Value                                |
+----------------+--------------------------------------+
| failure_domain | host                                 |
| id             | 2199e71e-ce8a-4ba9-81cd-75502f0344ca |
| name           | mystorpolicy                         |
| redundancy     | encoding=3+2                         |
| tier           | 3                                    |
+----------------+--------------------------------------+
```

This command shows the details of the storage policy `mystorpolicy`.

3.8.4 vinfra cluster storage-policy set

Modify storage policy parameters:

```
usage: vinfra cluster storage-policy set [--name <name>] [--tier {0,1,2,3}]
                                         [--replicas <norm>[:<min>] |
                                         --encoding <M>+<N>] [--failure-domain
                                         {disk,host,rack,row,room}] <storage-policy>
```

`--name <name>`

A new name for the storage policy

`--tier {0,1,2,3}`

Storage tier

`--replicas <norm>[:<min>]`

Storage replication mapping in the format:

- `norm`: the number of replicas to maintain;
- `min`: the minimum required number of replicas (optional).

`--encoding <M>+<N>`

Storage erasure encoding mapping in the format:

- `M`: the number of data blocks;
- `N`: the number of parity blocks.

`--failure-domain {disk,host,rack,row,room}`

Storage failure domain

`<storage-policy>`

Storage policy ID or name

Example:

```
# vinfra cluster storage-policy set mystorpolicy --encoding 5+2
+----------------+--------------------------------------+
| Field          | Value                                |
+----------------+--------------------------------------+
| failure_domain | host                                 |
| id             | 2199e71e-ce8a-4ba9-81cd-75502f0344ca |
| name           | mystorpolicy                         |
| redundancy     | encoding=5+2                         |
| tier           | 3                                    |
+----------------+--------------------------------------+
```

This command changes the redundancy type for the storage policy `mystorpolicy` from erasure coding 3+2 to 5+2.

3.8.5 vinfra cluster storage-policy delete

The default policy cannot be deleted.

Remove an existing storage policy:

```
usage: vinfra cluster storage-policy delete <storage-policy>
```

`<storage-policy>`

Storage policy ID or name

Example:

```
# vinfra cluster storage-policy delete mystorpolicy
Operation successful
```

This command deletes the storage policy `mystorpolicy`.

3.9 Managing Volumes

3.9.1 vinfra service compute volume create

Create a new compute volume:

```
usage: vinfra service compute volume create [--description <description>]
                                            [--network-install <network_install>]
                                            [--image <image>] --storage-policy
                                            <storage_policy> --size <size-gb>
                                            <volume-name>
```

`--description <description>`

Volume description

`--network-install <network_install>`

Perform network install (`true` or `false`)

`--image <image>`

Source compute image ID or name

`--storage-policy <storage_policy>`

Storage policy ID or name

`--size <size-gb>`

Volume size, in gigabytes

`<volume-name>`

Volume name

Example:

```
# vinfra service compute volume create myvolume --storage-policy default --size 8
+--------------------------------+--------------------------------------+
| Field                          | Value                                |
+--------------------------------+--------------------------------------+
| attachments                    | []                                   |
| availability_zone              | nova                                 |
| bootable                       | False                                |
| consistencygroup_id            |                                      |
| created_at                     | 2018-09-12T12:30:12.665916           |
| description                    |                                      |
| encrypted                      | False                                |
| id                             | c9c0e9e7-ce7a-4566-99d5-d7e40f2987ab |
| imageRef                       |                                      |
| migration_status               |                                      |
| multiattach                    | False                                |
| name                           | myvolume                             |
| network_install                | False                                |
| os-vol-host-attr:host          |                                      |
| os-vol-mig-status-attr:migstat |                                      |
| os-vol-mig-status-attr:name_id |                                      |
| project_id                     | 72a5db3a033c403a86756021e601ef34     |
| replication_status             |                                      |
| size                           | 8                                    |
| snapshot_id                    |                                      |
| source_volid                   |                                      |
| status                         | creating                             |
| storage_policy_name            | default                              |
| updated_at                     |                                      |
| user_id                        | 98bf389983c24c07af9677b931783143     |
| volume_image_metadata          |                                      |
+--------------------------------+--------------------------------------+
```

This command creates a volume `myvolume` sized 8 GB and chooses the default storage policy for it.

3.9.2 vinfra service compute volume list

List compute volumes:

```
usage: vinfra service compute volume list
```

Example:

```
# vinfra service compute volume list -c id -c name -c size -c status
+--------------------------------------+----------+------+-----------+
| id                                   | name     | size | status    |
+--------------------------------------+----------+------+-----------+
| c9c0e9e7-ce7a-4566-99d5-d7e40f2987ab | myvolume |    8 | available |
+--------------------------------------+----------+------+-----------+
```

This command lists volumes available to the compute cluster. (The output is abridged to fit on page.)

3.9.3 vinfra service compute volume show

Display compute volume details:

```
usage: vinfra service compute volume show <volume>
```

<volume>

Volume ID or name

Example:

```
# vinfra service compute volume show myvolume
+--------------------------------+------------------------------------------------------+
| Field                          | Value                                                |
+--------------------------------+------------------------------------------------------+
| attachments                    | []                                                   |
| availability_zone              | nova                                                 |
| bootable                       | False                                                |
| consistencygroup_id            |                                                      |
| created_at                     | 2018-09-12T12:30:12.665916                           |
| description                    |                                                      |
| encrypted                      | False                                                |
| id                             | c9c0e9e7-ce7a-4566-99d5-d7e40f2987ab                 |
| imageRef                       |                                                      |
| migration_status               |                                                      |
| multiattach                    | False                                                |
| name                           | myvolume                                             |
| network_install                | False                                                |
| os-vol-host-attr:host          | stor-1.example.com.vstoragedomain@vstorage#vstorage  |
| os-vol-mig-status-attr:migstat |                                                      |
| os-vol-mig-status-attr:name_id |                                                      |
| project_id                     | 72a5db3a033c403a86756021e601ef34                     |
| replication_status             |                                                      |
| size                           | 8                                                    |
| snapshot_id                    |                                                      |
| source_volid                   |                                                      |
| status                         | available                                            |
| storage_policy_name            | default                                              |
| updated_at                     | 2018-09-12T12:30:33.167654                           |
| user_id                        | 98bf389983c24c07af9677b931783143                     |
| volume_image_metadata          |                                                      |
+--------------------------------+------------------------------------------------------+
```

This command shows the details for the volume `myvolume`.

3.9.4 vinfra service compute volume set

Modify volume parameters:

```
usage: vinfra service compute volume set [--description <description>]
                                         [--network-install <network_install>]
                                         [--storage-policy <storage_policy>]
                                         [--bootable <bootable>]
                                         [--name <name>] <volume>
```

`--description <description>`

Volume description

`--network-install <network_install>`

Perform network install (`true` or `false`)

`--storage-policy <storage_policy>`

Storage policy ID or name

`--bootable <bootable>`

Make bootable (`true` or `false`)

`--name <name>`

A new name for the volume

`<volume>`

Volume ID or name

Example:

```
# vinfra service compute volume set myvolume --storage-policy mystorpolicy
+--------------------------------+------------------------------------------------------+
| Field                          | Value                                                |
+--------------------------------+------------------------------------------------------+
| attachments                    | []                                                   |
| availability_zone              | nova                                                 |
| bootable                       | False                                                |
| consistencygroup_id            |                                                      |
| created_at                     | 2018-09-12T12:30:12.665916                           |
| description                    |                                                      |
| encrypted                      | False                                                |
| id                             | c9c0e9e7-ce7a-4566-99d5-d7e40f2987ab                 |
| imageRef                       |                                                      |
| migration_status               |                                                      |
| multiattach                    | False                                                |
| name                           | myvolume                                             |
| network_install                | False                                                |
```

```
| os-vol-host-attr:host          | stor-1.example.com.vstoragedomain@vstorage#vstorage |
| os-vol-mig-status-attr:migstat |                                                     |
| os-vol-mig-status-attr:name_id |                                                     |
| project_id                     | 72a5db3a033c403a86756021e601ef34                    |
| replication_status             |                                                     |
| size                           | 8                                                   |
| snapshot_id                    |                                                     |
| source_volid                   |                                                     |
| status                         | available                                           |
| storage_policy_name            | mystorpolicy                                        |
| updated_at                     | 2018-09-12T12:55:29.298717                          |
| user_id                        | 98bf389983c24c07af9677b931783143                    |
| volume_image_metadata          |                                                     |
+--------------------------------+-----------------------------------------------------+
```

This command changes the storage policy of the volume `myvolume` to `mystorpolicy`.

3.9.5 vinfra service compute volume extend

Extend a compute volume:

```
usage: vinfra service compute volume extend --size <size_gb> <volume>
```

`<volume>`

Volume ID or name

Example:

```
# vinfra service compute volume extend myvolume --size 16
Operation successful
```

This command extends the volume `myvolume` to 16 GB.

3.9.6 vinfra service compute server volume attach

Attach a volume to a compute server:

```
usage: vinfra service compute server volume attach --server <server> <volume>
```

`--server <server>`

Compute server ID or name

`<volume>`

Volume ID or name

Example:

```
# vinfra service compute server volume attach e4cb5363-1fb2-41f5-b24b-18f98a388cba \
--server 871fef54-519b-4111-b18d-d2039e2410a8
+--------+--------------------------------------+
| Field  | Value                                |
+--------+--------------------------------------+
| device | /dev/vdb                             |
| id     | e4cb5363-1fb2-41f5-b24b-18f98a388cba |
+--------+--------------------------------------+
```

This command attaches the available volume with the ID `e4cb5363-1fb2-41f5-b24b-18f98a388cba` to the VM with the ID `871fef54-519b-4111-b18d-d2039e2410a8`.

3.9.7 vinfra service compute server volume detach

Detach a volume from a compute server:

```
usage: vinfra service compute server volume detach --server <server> <volume>
```

`--server <server>`

Compute server ID or name

`<volume>`

Volume ID or name

Example:

```
# vinfra service compute server volume detach e4cb5363-1fb2-41f5-b24b-18f98a388cba \
--server 871fef54-519b-4111-b18d-d2039e2410a8
Operation successful
```

This command detaches the volume with the ID `e4cb5363-1fb2-41f5-b24b-18f98a388cba` from the VM with the ID `871fef54-519b-4111-b18d-d2039e2410a8`.

3.9.8 vinfra service compute volume delete

Delete a compute volume:

```
usage: vinfra service compute volume delete <volume>
```

`<volume>`

Volume ID or name

Example:

```
# vinfra service compute volume delete myvolume2
Operation successful
```

This command deletes the volume `myvolume2`.

3.10 Managing Virtual Machines

3.10.1 vinfra service compute server create

Create a new compute server:

```
usage: vinfra service compute server create [--description <description>]
                                            [--metadata <metadata>]
                                            [--user-data <user-data>]
                                            [--key-name <key-name>]
                                            [--config-drive] [--count <count>]
                                            --network <id=id[,key=value,...]>
                                            --volume <source=source[,key=value,...]>
                                            --flavor <flavor> <server-name>
```

`--description <description>`

Server description

`--metadata <metadata>`

Server metadata

`--user-data <user-data>`

User data file

`--key-name <key-name>`

Key pair to inject

`--config-drive`

Use an ephemeral drive

`--count <count>`

If count is specified and greater than 1, the `name` argument is treated as a naming pattern.

`--network <id=id[,key=value,...]>`

Create a compute server with a specified network. Specify this option multiple times to create multiple networks.

- `id`: attach network interface to a specified network (ID or name);
- comma-separated `key=value` pairs with keys (optional):
  - `mac`: MAC address for network interface;
  - `fixed-ip`: fixed IP address for network interface;
  - `spoofing-protection`: enable or disable spoofing protection for network interface (`on` or `off`).

`--volume <source=source[,key=value,...]>`

Create a compute server with a specified volume. Specify this option multiple times to create multiple volumes.

- `source`: source type (`volume`, `image`, `snapshot`, or `blank`);
- comma-separated `key=value` pairs with keys (optional):
  - `id`: resource ID or name for the specified source type (required for source types `volume`, `image`, and `snapshot`);
  - `size`: block device size, in gigabytes (required for source types `image` and `blank`);
  - `boot-index`: block device boot index (required for multiple volumes with source type `volume`);
  - `bus`: block device controller type (`ide`, `usb`, `virtio`, `scsi`, or `sata`);
  - `type`: block device type (`disk` or `cdrom`);
  - `rm`: remove block device on compute server termination (`yes` or `no`) ;
  - `storage-policy`: block device storage policy.

`--flavor <flavor>`

Flavor ID or name

`<server-name>`

A new name for the compute server

Example:

```
# vinfra service compute server create myvm \
--network id=private,fixed-ip=192.168.0.100 \
--volume source=image,id=cirros,size=1 --flavor tiny
+--------------+--------------------------------------+
| Field        | Value                                |
+--------------+--------------------------------------+
| config_drive |                                      |
```

```
| created      | 2018-09-12T13:25:02Z                 |
| description  |                                      |
| flavor       | 100                                  |
| host         |                                      |
| id           | f6656fb5-e165-4afa-a119-45882acc6af1 |
| key_name     |                                      |
| metadata     | {}                                   |
| name         | myvm                                 |
| networks     | []                                   |
| power_state  | NOSTATE                              |
| project_id   | 72a5db3a033c403a86756021e601ef34     |
| status       | BUILD                                |
| task_state   | scheduling                           |
| updated      | 2018-09-12T13:25:03Z                 |
| user_data    |                                      |
| volumes      | []                                   |
+--------------+--------------------------------------+
```

This command creates a virtual machine `myvm based on the default Cirros image and the flavor `tiny` and connects it to the network `private` with the fixed IP address 192.168.128.100.

3.10.2 vinfra service compute server list

List compute servers:

```
usage: vinfra service compute server list
```

Example:

```
# vinfra service compute server list
+--------------------------------------+------+--------+----------------------------------+
| id                                   | name | status | host                             |
+--------------------------------------+------+--------+----------------------------------+
| 8cd29296-8bee-4efb-828d-0e522d816c6e | myvm | ACTIVE | stor-4.example.com.vstoragedomain |
+--------------------------------------+------+--------+----------------------------------+
```

This command lists all virtual machines in the compute cluster.

3.10.3 vinfra service compute server show

Display compute server details:

```
usage: vinfra service compute server show <server>
```

<server>

Compute server ID or name

Example:

```
# vinfra service compute server show myvm
+--------------+------------------------------------------+
| Field        | Value                                    |
+--------------+------------------------------------------+
| config_drive |                                          |
| created      | 2018-09-12T13:30:02Z                     |
| description  |                                          |
| flavor       | 100                                      |
| host         | stor-4.example.com.vstoragedomain        |
| id           | 8cd29296-8bee-4efb-828d-0e522d816c6e     |
| key_name     |                                          |
| metadata     | {}                                       |
| name         | myvm                                     |
| networks     | - id: 1bf2c9da-e324-49f0-8d41-20410615f905 |
|              |   ipam_enabled: true                     |
|              |   ips:                                   |
|              |   - 192.168.128.100                      |
|              |   mac_addr: fa:16:3e:54:04:f0            |
|              |   name: private                          |
|              |   spoofing_protection: false             |
| power_state  | RUNNING                                  |
| project_id   | 72a5db3a033c403a86756021e601ef34         |
| status       | ACTIVE                                   |
| task_state   |                                          |
| updated      | 2018-09-12T13:30:53Z                     |
| user_data    |                                          |
| volumes      | - 30092d44-3bdd-46d8-a360-a5bc6434cbf8   |
+--------------+------------------------------------------+
```

This command shows the details of the virtual machine `myvm`.

3.10.4 vinfra service compute server stat

Display compute server statistics:

```
usage: vinfra service compute server stat <server>
```

<server>

Compute server ID or name

Example:

```
# vinfra service compute server stat myvm
+----------+---------------------------------+
| Field    | Value                           |
+----------+---------------------------------+
| datetime | 2018-09-12T13:37:46.803999+00:00 |
```

```
| metrics  | block_capacity: 1073741824       |
|          | block_usage: 134549504           |
|          | cpu_usage: 0                     |
|          | mem_total: 536870912             |
|          | mem_usage: 130412544             |
|          | vcpus: 1                         |
+----------+----------------------------------+
```

This command shows the statistics for the virtual machine `myvm`.

3.10.5 vinfra service compute server iface attach

Attach a network to a compute server:

```
usage: vinfra service compute server iface attach [--mac <mac>] [--ip <ip-address>]
                                                  [--spoofing-protection {on,off}]
                                                  --server <server> --network <network>
```

`--mac <mac>`

MAC address

`--ip <ip-address>`

IP address

`--spoofing-protection {on|off}`

Enable spoofing protection for the network interface

`--server <server>`

Compute server ID or name

`--network <network>`

Network ID or name

Example:

```
# vinfra service compute server iface attach --network myprivnet --server myvm
+---------------+--------------------------------------+
| Field         | Value                                |
+---------------+--------------------------------------+
| fixed_ip      | 192.168.129.8                        |
| id            | 690ed3f2-2301-40e2-879a-126db2ecb57b |
| mac_address   | fa:16:3e:54:59:08                    |
| network_id    | 0710372e-2bdf-4dfe-b413-eb763da37e68 |
| spoofing<...> | False                                |
+---------------+--------------------------------------+
```

This command attaches the private network `myprivnet` to the virtual machine `myvm`.

3.10.6 vinfra service compute server iface list

List compute server networks:

```
usage: vinfra service compute server iface list --server <server>
```

`--server <server>`

Compute server ID or name

Example:

```
# vinfra service compute server iface list --server myvm
+----------------+---------------+-------------------+----------------+
| id             | network_id    | mac_address       | fixed_ip       |
+----------------+---------------+-------------------+----------------+
| 690ed3f2-<...> | 0710372e-<...> | fa:16:3e:54:59:08 | 192.168.129.8   |
| a5b13bf3-<...> | 1bf2c9da-<...> | fa:16:3e:b9:33:bb | 192.168.128.100 |
+----------------+---------------+-------------------+----------------+
```

This command lists the virtual networks that the virtual machine `myvm` is attached to. It also shows VM's IP address in each network.

3.10.7 vinfra service compute server iface detach

Detach a network interface from a compute server:

```
usage: vinfra service compute server iface detach --server <server> <interface>
```

`--server <server>`

Compute server ID or name

`<interface>`

Network interface ID

Example:

```
# vinfra service compute server iface detach 471e37fd-13ae-4b8f-b70c-90ac02cc4386 \
--server 6c80b07f-da46-4a8a-89a4-eecb8faceb27
Operation successful
```

This command detaches the network interface with the ID `471e37fd-13ae-4b8f-b70c-90ac02cc4386` from the VM with the ID `6c80b07f-da46-4a8a-89a4-eecb8faceb27`.

3.10.8 vinfra service compute server log

Display compute server log:

```
usage: vinfra service compute server log <server>
```

`<server>`

Compute server ID or name

Example:

```
# vinfra service compute server log myvm > myvm.log
```

This command prints the log of the virtual machine `myvm` to the file `myvm.log`.

3.10.9 vinfra service compute server migrate

Migrate a compute server to another host:

```
usage: vinfra service compute server migrate [--cold] [--node <node>] <server>
```

`--cold`

Perform cold migration. If not set, try to determine migration type automatically.

`--node <node>`

Destination node ID or hostname

`<server>`

Compute server ID or name

Example:

```
# vinfra service compute server migrate 6c80b07f-da46-4a8a-89a4-eecb8faceb27 \
--node e6255aed-d6e7-41b2-ba90-86164c1cd9a6
Operation successful
```

This command starts migration of the VM with the ID `6c80b07f-da46-4a8a-89a4-eecb8faceb27` to the compute node with the ID `e6255aed-d6e7-41b2-ba90-86164c1cd9a6`.

3.10.10 vinfra service compute server resize

Resize a compute server:

```
usage: vinfra service compute server resize --flavor <flavor> <server>
```

`--flavor <flavor>`

Apply flavor with ID or name

`<server>`

Compute server ID or name

Example:

```
# vinfra service compute server resize myvm --flavor small
Operation successful
```

This command changes the flavor of the virtual machine `myvm` to `small`.

3.10.11 vinfra service compute server start

Start a compute server:

```
usage: vinfra service compute server start <server>
```

`<server>`

Compute server ID or name

Example:

```
# vinfra service compute server start myvm
Operation successful
```

This command starts the virtual machine `myvm`.

3.10.12 vinfra service compute server pause

Pause a compute server:

```
usage: vinfra service compute server pause <server>
```

`<server>`

Compute server ID or name

Example:

```
# vinfra service compute server pause myvm
```

This command pauses the running virtual machine `myvm`.

3.10.13 vinfra service compute server unpause

Unpause a compute server:

```
usage: vinfra service compute server unpause <server>
```

`<server>`

Compute server ID or name

Example:

```
# vinfra service compute server unpause myvm
```

This command unpauses the paused virtual machine `myvm`.

3.10.14 vinfra service compute server suspend

Suspend a compute server:

```
usage: vinfra service compute server suspend <server>
```

`<server>`

Compute server ID or name

Example:

```
# vinfra service compute server suspend myvm
Operation successful
```

This command suspends the running virtual machine `myvm`.

3.10.15 vinfra service compute server resume

Resume a compute server:

```
usage: vinfra service compute server resume <server>
```

`<server>`

Compute server ID or name

Example:

```
# vinfra service compute server resume myvm
Operation successful
```

This command resumes the suspended virtual machine `myvm`.

3.10.16 vinfra service compute server reboot

Reboot a compute server:

```
usage: vinfra service compute server reboot [--hard] <server>
```

`--hard`

Perform hard reboot

`<server>`

Compute server ID or name

Example:

```
# vinfra service compute server reboot myvm
Operation successful
```

This command reboots the virtual machine `myvm`.

3.10.17 vinfra service compute server reset-state

Reset compute server state:

```
usage: vinfra service compute server reset-state [--state-error] <server>
```

`--state-error`

Reset server to ‘ERROR’ state

`<server>`

Compute server ID or name

Example:

```
# vinfra service compute server reset-state myvm
Operation successful
```

This command resets the state of the virtual machine `myvm`.

3.10.18 vinfra service compute server stop

Shut down a compute server:

```
usage: vinfra service compute server stop [--hard] <server>
```

`--hard`

Power off a compute server

`<server>`

Compute server ID or name

Example:

```
# vinfra service compute server stop myvm
Operation successful
```

This command stops the virtual machine `myvm`.

3.10.19 vinfra service compute server evacuate

Evacuate a stopped compute server from a failed host:

```
usage: vinfra service compute server evacuate <server>
```

`<server>`

Compute server ID or name

Example:

```
# vinfra service compute server evacuate ``myvm``
Operation successful
```

This command evacuates the stopped VM `myvm` from its node to another, healthy compute node.

3.10.20 vinfra service compute server delete

Delete a compute server:

```
usage: vinfra service compute server delete <server>
```

<server>

Compute server ID or name

Example:

```
# vinfra service compute server delete myvm
Operation successful
```

This command deletes the virtual machine `myvm`.

3.11 vinfra service compute cluster delete

Delete all nodes from the compute cluster:

```
usage: vinfra service compute cluster delete
```

Example:

```
# vinfra service compute cluster delete
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | 063e8a15-fcfe-4629-865f-b5e5fa44b38f |
+---------+--------------------------------------+
```

This command creates a task to release nodes from the compute cluster.

Task outcome:

```
# vinfra task show 063e8a15-fcfe-4629-865f-b5e5fa44b38f
+---------+-----------------------------------------------------------------+
| Field   | Value                                                           |
+---------+-----------------------------------------------------------------+
| args    | []                                                              |
| kwargs  | {}                                                              |
| name    | backend.presentation.compute.tasks.DestroyComputeClusterTask |
| state   | success                                                         |
| task_id | 063e8a15-fcfe-4629-865f-b5e5fa44b38f                            |
+---------+-----------------------------------------------------------------+
```

CHAPTER 4

Managing General Settings

4.1 Managing Licenses

4.1.1 vinfra cluster license load

Load a license from a key.

```
usage: vinfra cluster license load --key <license-key> --type <license-type>
```

`--key <license-key>`

License key to register. Specify this option multiple times to register multiple keys.

`--type <license-type>`

License type (`prolong` or `upgrade`)

Example:

```
# vinfra cluster license load --key A38600ML-3P6W746P-RZSK58BV-Y9ZH05Q5-2X7J48J6-KVRXRYPY-\
Z2FK7ZQ6-Y7FGZNYF --type upgrade
+------------+------------------------+
| Field      | Value                  |
+------------+------------------------+
| capacity   | 10995116277760         |
| expiration | 2021-01-10T12:42:00    |
| free_size  | 10973383165601         |
| spla       | registered: false      |
|            | registration_url: null |
| status     | active                 |
| total_size | 10995116277760         |
| used_size  | 21733112159            |
+------------+------------------------+
```

This command install the license from the key `A38600-3P6W74-RZSK58-Y9ZH05-2X7J48`.

4.1.2 vinfra cluster license show

Show details of the installed license:

```
usage: vinfra cluster license show
```

Example:

```
# vinfra cluster license show
+------------+------------------------+
| Field      | Value                  |
+------------+------------------------+
| capacity   | 10995116277760         |
| expiration | 2021-01-10T12:42:00    |
| free_size  | 10973383165601         |
| spla       | registered: false      |
|            | registration_url: null |
| status     | active                 |
| total_size | 10995116277760         |
| used_size  | 21733112159            |
+------------+------------------------+
```

This command shows the details of the currently installed license.

4.2 Managing Users

4.2.1 vinfra cluster user list-available-roles

List available user roles:

```
usage: vinfra cluster user list-available-roles
```

Example:

```
# vinfra cluster user list-available-roles
+-------------------------------------------------------------------+---------+---------------+
| description                                                       | id      | name          |
+-------------------------------------------------------------------+---------+---------------+
| Can create cluster, join nodes to cluster, and manage (assign and | cluster | Cluster       |
| release) disks.                                                   |         |               |
| Can add and remove SSH keys for cluster nodes access.             | ssh     | SSH           |
| Viewer role (read only)                                           | viewer  | Viewer        |
| Can create and manage compute cluster.                            | compute | Compute       |
| Can modify network settings and roles.                            | network | Network       |
| Can install updates.                                              | updates | Updates       |
| Can perform all management operations.                            | admin   | Administrator |
```

```
| Can create and manage S3 cluster.                                  | s3      | S3           |
| Can create and manage Acronis Backup Gateway.                      | abgw    | ABGW         |
| Can create and manage iSCSI targets LUNs and CHAP users.           | iscsi   | iSCSI        |
| Can create and manage NFS.                                         | nfs     | NFS          |
+--------------------------------------------------------------------+---------+--------------+
```

This command lists user roles available in Acronis Software-Defined Infrastructure.

4.2.2 vinfra cluster user create

Add an admin panel user:

```
usage: vinfra cluster user create [--description <description>] [--enable | --disable]
                                  [--roles <roles>] <name>
```

`--description <description>`

User description

`--enable`

Enable user

`--disable`

Disable user

`--roles <roles>`

A comma-separated list of user roles

`<name>`

User name

Example:

```
# vinfra cluster user create user1 --roles viewer --enable \
--description "A guest user"
Password:
+-------------------+----------------------------------------+
| Field             | Value                                  |
+-------------------+----------------------------------------+
| description       | A guest user                           |
| external_id       |                                        |
| external_provider |                                        |
| id                | 538e54fdeb13498e8f0bbff6773de5a1       |
| is_enabled        | True                                   |
| is_group          | False                                  |
| is_superuser      | False                                  |
| name              | user1                                  |
```

```
| roles             | - description: Viewer role (read only) |
|                   |   id: viewer                           |
|                   |   name: Viewer                         |
+-------------------+----------------------------------------+
```

This command creates and enables the user `user1`, assigns it the role `Viewer`, and sets its password and description.

4.2.3 vinfra cluster user list

List all admin panel users:

```
usage: vinfra cluster user list
```

Example:

```
# vinfra cluster user list
+----------------------------------+----------------------+------------+--------------+-------+
| id                               | name                 | is_enabled | is_superuser | roles |
+----------------------------------+----------------------+------------+--------------+-------+
| 60b67333c545442f98c084a56db7a06d | admin                | True       | True         |       |
+----------------------------------+----------------------+------------+--------------+-------+
```

This command lists users registered in Acronis Software-Defined Infrastructure.

4.2.4 vinfra cluster user show

Show details of an admin panel user:

```
usage: vinfra cluster user show <user>
```

`<user>`

User ID or name

Example:

```
# vinfra cluster user show admin
+-------------------+----------------------------------+
| Field             | Value                            |
+-------------------+----------------------------------+
| description       |                                  |
| external_id       |                                  |
| external_provider |                                  |
| id                | 60b67333c545442f98c084a56db7a06d |
| is_enabled        | True                             |
```

```
| is_group          | False                            |
| is_superuser      | True                             |
| name              | admin                            |
| roles             |                                  |
+-------------------+----------------------------------+
```

This command shows the details of the user `admin`.

4.2.5 vinfra cluster user set

Modify admin panel user parameters:

```
usage: vinfra cluster user set [--description <description>] [--enable | --disable]
                               [--set-roles <roles> | --add-roles <roles> |
                               --del-roles <roles>] [--password] [--name <name>] <user>
```

`--description <description>`

User description

`--enable`

Enable user

`--disable`

Disable user

`--set-roles <roles>`

A comma-separated list of user roles to set (overwrites the current user roles)

`--add-roles <roles>`

A comma-separated list of user roles to add

`--del-roles <roles>`

A comma separated list of user roles to remove

`--password`

User password

`--name <name>`

A new name for the user

`<user>`

User ID or name

Example:

```
# vinfra cluster user set admin --description "The admin"
+-------------------+----------------------------------+
| Field             | Value                            |
+-------------------+----------------------------------+
| description       | The admin                        |
| external_id       |                                  |
| external_provider |                                  |
| id                | 60b67333c545442f98c084a56db7a06d |
| is_enabled        | True                             |
| is_group          | False                            |
| is_superuser      | True                             |
| name              | admin                            |
| roles             |                                  |
+-------------------+----------------------------------+
```

This command adds a description for the user `admin`.

4.2.6 vinfra cluster user change-password

Change password of an admin panel user:

```
usage: vinfra cluster user change-password
```

Example:

```
# vinfra cluster user change-password
Current password:
New password:
Confirm password:
```

This command changes the password of the current user.

4.2.7 vinfra cluster user delete

Remove an admin panel user:

```
usage: vinfra cluster user delete <user>
```

`<user>`

User ID or name

Example:

```
# vinfra cluster user delete user1
Operation successful
```

This command deletes the user `user1`.

4.3 Managing SSH Keys

4.3.1 vinfra cluster sshkey add

Add an SSH public key from a file:

```
usage: vinfra cluster sshkey add <file>
```

`<file>`

SSH public key file

Example:

```
# vinfra cluster sshkey add id_rsa.pub
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | 100a54ce-0bf5-4bc0-8e46-2e8b952343e6 |
+---------+--------------------------------------+
```

This command creates a task to add a public SSH key from the file `mykey.pub` to the list of trusted keys.

Task outcome:

```
# vinfra task show 100a54ce-0bf5-4bc0-8e46-2e8b952343e6
+---------+-------------------------------------------------------------------------------------------+
| Field   | Value                                                                                     |
+---------+-------------------------------------------------------------------------------------------+
| args    | - admin                                                                                   |
|         | - 1                                                                                       |
| kwargs  | key: ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQCueWO956J/u5kjWnia7zePChoTMVBtsh1TDNgOskM |
|         | shfHWUzzfydi3/4sTrJ++6dtIoS1D3VVHvHBvp456PT5e/eVy7u0SipOPPoDY2vS2IEY+zjT6MYABi6oEYom |
|         | Dbi7CsRL02HcTWzAkooZNlimWPggYaMT10BZOKAvNB+Ctpkw8JaT5PRve8UVfjxIQIzL6pQOf0CDeCHgDsvw |
|         | xK7SrqOvBzTlF9mWkGdTGy+R0JrgGk+v9PvDXZwyeK+qS54uaGmpB6ZRkKMroIk3h+nZ4y/1eQ6m1C8Aspa0 |
|         | nnaMaNKOtwOibrd3MDroMcqkJWTTH/cukD3sB+MjL6nmFlrrAfRU6PBkwysIio6/XHS9jG+TI7NeRApkHnwi |
|         | vwIWEKSg6pqaiLUsMi/46KCHzde2OzgO8Hd0R5d7hNN/80mhD7b+bY9wig+VTMoQFQYSWrIy/qLL95ws4amg |
|         | nXOIksNFjfFEE/+lMcZXt3j5kqnjW7OT2/xkqqWoumaM+FEPLNijL18yb29/XJr/cQZX5R9iXSk33DVjhln/ |
|         | HG7xpHqAtrXbvKY8zI8t23otGT/rSvWRWV/wgPBZVWSWtsE99FEMmwmxk/b3KuPhi0jK0IUKcv5UBL+NLHw4 |
|         | rZRiYgw/fWXPO3f6ZSLLJXtW4iW+BQL60qQWUNQ==                                                  |
|         |   user@example.com                                                                        |
| name    | backend.presentation.nodes.ssh.tasks.CreateSshKeyTask                                     |
| result  | id: 6a2fb834-4bc6-4597-ae74-7cacf96b7c75                                                  |
|         | key: ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQCueWO956J/u5kjWnia7zePChoTMVBtsh1TDNgOskM |
|         | shfHWUzzfydi3/4sTrJ++6dtIoS1D3VVHvHBvp456PT5e/eVy7u0SipOPPoDY2vS2IEY+zjT6MYABi6oEYom |
|         | Dbi7CsRL02HcTWzAkooZNlimWPggYaMT10BZOKAvNB+Ctpkw8JaT5PRve8UVfjxIQIzL6pQOf0CDeCHgDsvw |
```

```
|         | xK7SrqOvBzTlF9mWkGdTGy+R0JrgGk+v9PvDXZwyeK+qS54uaGmpB6ZRkKMroIk3h+nZ4y/1eQ6m1C8Aspa0 |
|         | nnaMaNKOtwOibrd3MDroMcqkJWTTH/cukD3sB+MjL6nmFlrrAfRU6PBkwysIio6/XHS9jG+TI7NeRApkHnwi |
|         | vwIWEKSg6pqaiLUsMi/46KCHzde2OzgO8Hd0R5d7hNN/80mhD7b+bY9wig+VTMoQFQYSWrIy/qLL95ws4amg |
|         | nXOIksNFjfFEE/+lMcZXt3j5kqnjW7OT2/xkqqWoumaM+FEPLNijL18yb29/XJr/cQZX5R9iXSk33DVjhln/ |
|         | HG7xpHqAtrXbvKY8zI8t23otGT/rSvWRWV/wgPBZVWSWtsE99FEMmwmxk/b3KuPhi0jK0IUKcv5UBL+NLHw4 |
|         | rZRiYgw/fWXPO3f6ZSLLJXtW4iW+BQL60qQWUNQ== |
|         |   user@example.com |
|         | label: user@example.com |
| state   | success |
| task_id | 100a54ce-0bf5-4bc0-8e46-2e8b952343e6 |
+---------+-----------------------------------------------------------------------------------+
```

4.3.2 vinfra cluster sshkey list

Show the list of added SSH public keys:

```
usage: vinfra cluster sshkey list
```

Example:

```
# vinfra cluster sshkey list
+--------------------------------------+--------------------------------------+------------------+
| id                                   | key                                  | label            |
+--------------------------------------+--------------------------------------+------------------+
| 8ccf7f1b-6a53-4d74-99ce-c410d51a9921 | ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACA | user@example.com |
|                                      | QCueWO956J/u5kjWnia7zePChoTMVBtsh1TDN |                  |
|                                      | gOskMg5shfHWUzzfydi3/4sTrJ++6dtIoS1D3 |                  |
|                                      | VVHvHBvp456PT5e/eVy7u0SipOPPoDY2vS2IE |                  |
|                                      | Y+zjT6MYABi6oEYomIIDbi7CsRL02HcTWzAko |                  |
|                                      | oZNlimWPggYaMT10BZOKAvNB+Ctpkw8JaT5PR |                  |
|                                      | ve8UVfjxIQIzL6pQOf0CDeCHgDsvwcmxK7Srq |                  |
|                                      | OvBzTlF9mWkGdTGy+R0JrgGk+v9PvDXZwyeK+ |                  |
|                                      | qS54uaGmpB6ZRkKMroIk3h+nZ4y/1eQ6m1C8A |                  |
|                                      | spa0f5nnaMaNKOtwOibrd3MDroMcqkJWTTH/c |                  |
|                                      | ukD3sB+MjL6nmFlrrAfRU6PBkwysIio6/XHS9 |                  |
|                                      | jG+TI7NeRApkHnwiiOvwIWEKSg6pqaiLUsMi/ |                  |
|                                      | 46KCHzde2OzgO8Hd0R5d7hNN/80mhD7b+bY9w |                  |
|                                      | ig+VTMoQFQYSWrIy/qLL95ws4amgAQnXOIksN |                  |
|                                      | FjfFEE/+lMcZXt3j5kqnjW7OT2/xkqqWoumaM |                  |
|                                      | +FEPLNijL18yb29/XJr/cQZX5R9iXSk33DVjh |                  |
|                                      | ln/EyHG7xpHqAtrXbvKY8zI8t23otGT/rSvWR |                  |
|                                      | WV/wgPBZVWSWtsE99FEMmwmxk/b3KuPhi0jK0 |                  |
|                                      | IUKcv5UBL+NLHw4gQrZRiYgw/fWXPO3f6ZSLL |                  |
|                                      | JXtW4iW+BQL60qQWUNQ== |                  |
|                                      | user@example.com |                  |
+--------------------------------------+--------------------------------------+------------------+
```

This command lists trusted SSH keys.

4.3.3 vinfra cluster sshkey delete

Remove an SSH public key from storage cluster nodes:

```
usage: vinfra cluster sshkey delete <sshkey>
```

<sshkey>

SSH key value

Example:

```
# vinfra cluster sshkey delete 8ccf7f1b-6a53-4d74-99ce-c410d51a9921
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | 053802b2-b4c3-454d-89e2-6d6d312dd2ed |
+---------+--------------------------------------+
```

This command creates a task to delete the SSH key with the ID `8ccf7f1b-6a53-4d74-99ce-c410d51a9921`.

Task outcome:

```
# vinfra task show 053802b2-b4c3-454d-89e2-6d6d312dd2ed
+---------+------------------------------------------------------------+
| Field   | Value                                                      |
+---------+------------------------------------------------------------+
| args    | - admin                                                    |
|         | - 1                                                        |
|         | - 8ccf7f1b-6a53-4d74-99ce-c410d51a9921                     |
| kwargs  | {}                                                         |
| name    | backend.presentation.nodes.ssh.tasks.RemoveSshKeyTask      |
| state   | success                                                    |
| task_id | 053802b2-b4c3-454d-89e2-6d6d312dd2ed                       |
+---------+------------------------------------------------------------+
```

4.4 Managing External DNS Servers

4.4.1 vinfra cluster settings dns show

Display DNS servers:

```
usage: vinfra cluster settings dns show
```

Example:

```
# vinfra cluster settings dns show
+------------------+-----------------------------------+
| Field            | Value                             |
+------------------+-----------------------------------+
| dhcp_nameservers | 10.10.0.10,10.10.0.11,10.37.130.2 |
| nameservers      | 10.10.0.11,10.10.0.10             |
+------------------+-----------------------------------+
```

This command lists the currently used DNS servers: both internal (obtained via DHCP) and external (static set by the user).

4.4.2 vinfra cluster settings dns set

Set DNS servers:

```
usage: vinfra cluster settings dns set --nameservers <nameservers>
```

`--nameservers <nameservers>`

A comma-separated list of DNS servers

Example:

```
# vinfra cluster settings dns set --nameservers 8.8.8.8
+------------------+---------------+
| Field            | Value         |
+------------------+---------------+
| dhcp_nameservers | - 10.10.0.10  |
|                  | - 10.10.0.11  |
|                  | - 10.37.130.2 |
| nameservers      | - 8.8.8.8     |
+------------------+---------------+
```

This command sets the external DNS server to `8.8.8.8`.

4.5 Configuring Management Node High Availability

4.5.1 vinfra cluster ha create

Create a HA configuration:

```
usage: vinfra cluster ha create --virtual-ip <network:ip> --nodes <nodes> [--force]
```

`--virtual-ip <network:ip>`

HA configuration mapping in the format:

- `network`: network to include in the HA configuration (must include at least one of these traffic types: Internal management, Admin panel, or Compute API);
- `ip`: virtual IP address that will be used in the HA configuration.

Specify this option multiple times to create a HA configuration for multiple networks.

`--nodes <nodes>`

A comma-separated list of node IDs or hostnames

`--force`

Skip checks for minimal hardware requirements

Example:

```
# vinfra cluster ha create --virtual-ip Private:10.37.130.200 \
--virtual-ip Public:10.94.41.244 --nodes 94d58604-6f30-4339-8578-adb7903b7277,\
f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4,7d7d37b8-4c06-4f1a-b3a6-4b54257d70ce
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | 80a00e55-335d-4d41-bac4-5fee4791d423 |
+---------+--------------------------------------+
```

This command creates a task to create a management node HA cluster from nodes with the IDs `94d58604-6f30-4339-8578-adb7903b7277`, `f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4`, and `7d7d37b8-4c06-4f1a-b3a6-4b54257d70ce`.

The command must specify the network with the traffic type `Internal management` as well as one with the traffic type `Admin panel`.

Important: After the HA cluster has been created, the admin panel will only be accessible at the provided public IP address. Log in to said address via SSH to continue managing Acronis Software-Defined Infrastructure with the `vinfra` CLI tool. You may also need to set the `VINFRA_PASSWORD` environment variable again, because you will access different HA cluster nodes on each log in where it may not have been set.

Task outcome:

```
# vinfra task show 80a00e55-335d-4d41-bac4-5fee4791d423
+---------+-------------------------------------------------+
| Field   | Value                                           |
+---------+-------------------------------------------------+
| args    | - - - 6095a997-e5f1-493d-a750-41ddf277153b      |
|         |     - 10.37.130.200                             |
|         |   - - 358bdc39-cd8b-4565-8ebf-e7c12dcd1cf7      |
|         |     - 10.94.41.244                              |
|         | - - 94d58604-6f30-4339-8578-adb7903b7277        |
|         |   - f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4        |
|         |   - 7d7d37b8-4c06-4f1a-b3a6-4b54257d70ce        |
| kwargs  | {}                                              |
| name    | backend.presentation.ha.tasks.CreateHaConfigTask |
| state   | success                                         |
| task_id | 80a00e55-335d-4d41-bac4-5fee4791d423            |
+---------+-------------------------------------------------+
```

4.5.2 vinfra cluster ha join

Join node to the HA configuration:

```
usage: vinfra cluster ha join --nodes <nodes>
```

`--nodes <nodes>`

A comma-separated list of node IDs or hostnames

Example:

```
# vinfra cluster ha join --nodes 4b83a87d-9adf-472c-91f0-782c47b2d5f1
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | 565e9146-254b-4f7a-a2ff-b7119c95baa9 |
+---------+--------------------------------------+
```

This command creates a task to add the node with the ID `4b83a87d-9adf-472c-91f0-782c47b2d5f1` to the management node HA cluster.

Task outcome:

```
# vinfra task show 565e9146-254b-4f7a-a2ff-b7119c95baa9
+---------+-----------------------------------------------+
| Field   | Value                                         |
+---------+-----------------------------------------------+
| args    | - - 4b83a87d-9adf-472c-91f0-782c47b2d5f1      |
| kwargs  | {}                                            |
| name    | backend.presentation.ha.tasks.JoinHaNodesTask |
| state   | success                                       |
```

```
| task_id | 565e9146-254b-4f7a-a2ff-b7119c95baa9          |
+---------+-----------------------------------------------+
```

4.5.3 vinfra cluster ha show

Display the HA configuration:

```
usage: vinfra cluster ha show
```

Example:

```
# vinfra cluster ha show
+-----------------------+-----------------------------------------------------+
| Field                 | Value                                               |
+-----------------------+-----------------------------------------------------+
| ha_cluster_location   | https://10.37.130.200:8888                          |
| nodes                 | - id: 7d7d37b8-4c06-4f1a-b3a6-4b54257d70ce          |
|                       |   interface: xxx                                    |
|                       |   ipaddr: 10.37.130.103                             |
|                       |   is_primary: false                                 |
|                       | - id: 94d58604-6f30-4339-8578-adb7903b7277          |
|                       |   interface: xxx                                    |
|                       |   ipaddr: 10.37.130.101                             |
|                       |   is_primary: true                                  |
|                       | - id: f59dabdb-bd1c-4944-8af2-26b8fe9ff8d4          |
|                       |   interface: xxx                                    |
|                       |   ipaddr: 10.37.130.102                             |
|                       |   is_primary: false                                 |
|                       | - id: 4b83a87d-9adf-472c-91f0-782c47b2d5f1          |
|                       |   interface: xxx                                    |
|                       |   ipaddr: 10.37.130.104                             |
|                       |   is_primary: false                                 |
| primary_node_location | https://10.94.41.23:8888                            |
| virtual_ips           | - ip: 10.37.130.200                                 |
|                       |   roles_set: 6095a997-e5f1-493d-a750-41ddf277153b   |
|                       | - ip: 10.94.41.244                                  |
|                       |   roles_set: 358bdc39-cd8b-4565-8ebf-e7c12dcd1cf7   |
+-----------------------+-----------------------------------------------------+
```

This command shows the management node HA cluster configuration.

4.5.4 vinfra cluster ha release

Relase node from the HA configuration:

```
usage: vinfra cluster ha release --nodes <nodes>
```

`--nodes <nodes>`

A comma-separated list of node IDs or hostnames

Example:

```
# vinfra cluster ha release --nodes 4b83a87d-9adf-472c-91f0-782c47b2d5f1
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | c1f3e9c3-0a7b-455a-96d4-cef3b7e86e62 |
+---------+--------------------------------------+
```

This command creates a task to release the node with the ID `4b83a87d-9adf-472c-91f0-782c47b2d5f1` from the management node HA cluster.

Task outcome:

```
# vinfra task show c1f3e9c3-0a7b-455a-96d4-cef3b7e86e62
+---------+-----------------------------------------------------+
| Field   | Value                                               |
+---------+-----------------------------------------------------+
| args    | - - 4b83a87d-9adf-472c-91f0-782c47b2d5f1            |
| kwargs  | {}                                                  |
| name    | backend.presentation.ha.tasks.DisjoinHaNodesTask    |
| state   | success                                             |
| task_id | c1f3e9c3-0a7b-455a-96d4-cef3b7e86e62                |
+---------+-----------------------------------------------------+
```

4.6 Managing Storage Tier Encryption

4.6.1 vinfra cluster settings encryption show

Display storage tiers encyption:

```
usage: vinfra cluster settings encryption show
```

Example:

```
# vinfra cluster settings encryption show
+-------+-------+
| Field | Value |
+-------+-------+
| tier0 | False |
| tier1 | False |
| tier2 | False |
```

```
| tier3 | False |
+-------+-------+
```

This command shows encryption status of each storage tier.

4.6.2 vinfra cluster settings encryption set

Set storage tiers encyption:

```
usage: vinfra cluster settings encryption set [--tier-enable {0,1,2,3}]
                                              [--tier-disable {0,1,2,3}]
```

`--tier-enable {0,1,2,3}`

Enable encryption for storage tiers. This option can be used multiple times.

`--tier-disable {0,1,2,3}`

Disable encryption for storage tiers. This option can be used multiple times.

Example:

```
# vinfra cluster settings encryption set --tier-enable 2
+-------+-------+
| Field | Value |
+-------+-------+
| tier0 | False |
| tier1 | False |
| tier2 | True  |
| tier3 | False |
+-------+-------+
```

This command enables encryption for the storage tier 2.

4.7 Managing Alerts

4.7.1 vinfra cluster alert list

List alert log entries:

```
usage: vinfra cluster alert list [--all]
```

`--all`

Show both enabled and disabled alerts

Example:

```
# vinfra cluster alert list --all
+----+-----------------+---------------------+----------+---------+
| id | type            | datetime            | severity | enabled |
+----+-----------------+---------------------+----------+---------+
|  1 | Network warning | 2018-08-30T18:02:14 | warning  | True    |
|  2 | Network warning | 2018-08-30T18:02:14 | warning  | True    |
|  3 | Network warning | 2018-08-30T18:02:14 | warning  | True    |
|  4 | Network warning | 2018-08-31T13:02:15 | warning  | True    |
|  5 | Network warning | 2018-08-31T13:02:15 | warning  | True    |
|  6 | Network warning | 2018-08-31T13:02:15 | warning  | True    |
|  7 | Network warning | 2018-08-31T13:02:15 | warning  | True    |
|  8 | Network warning | 2018-08-31T13:02:15 | warning  | True    |
|  9 | Network warning | 2018-08-31T13:02:15 | warning  | True    |
| 10 | Network warning | 2018-08-31T13:02:15 | warning  | True    |
| 11 | Network warning | 2018-08-31T13:02:15 | warning  | True    |
| 12 | Network warning | 2018-08-31T13:02:15 | warning  | True    |
| 13 | Network warning | 2018-08-31T13:02:15 | warning  | True    |
| 14 | Network warning | 2018-08-31T13:02:15 | warning  | True    |
| 15 | Network warning | 2018-08-31T13:02:15 | warning  | True    |
+----+-----------------+---------------------+----------+---------+
```

This command lists all alerts in the log and shows which alerts are enabled and disabled.

4.7.2 vinfra cluster alert show

Show details of the specified alert log entry:

```
usage: vinfra cluster alert show <alert>
```

<alert>

Alert ID

Example:

```
# vinfra cluster alert show 1
+--------------+--------------------------------------------------------------------------------+
| Field        | Value                                                                          |
+--------------+--------------------------------------------------------------------------------+
| _type        | undefined_speed                                                                |
| cluster_id   |                                                                                |
| cluster_name |                                                                                |
| datetime     | 2018-08-30T18:02:14.855302+00:00                                               |
| details      | host: stor-1.example.com.vstoragedomain.                                       |
| enabled      | True                                                                           |
| group        | node                                                                           |
| host         | stor-1.example.com.vstoragedomain.                                             |
| id           | 1                                                                              |
```

```
| message      | Network interface "eth1" on node "stor-1.example.com.vstoragedomain." has |
|              | an undefined speed                                                        |
| node_id      | 4f96acf5-3bc8-4094-bcb6-4d1953be7b55                                      |
| object_id    | eth1                                                                      |
| severity     | warning                                                                   |
| suspended    |                                                                           |
| type         | Network warning                                                           |
+--------------+---------------------------------------------------------------------------+
```

This command shows the details of alert with ID 1.

4.7.3 vinfra cluster alert delete

Remove an entry from the alert log:

```
usage: vinfra cluster alert delete <alert>
```

<alert>

Alert ID

Example:

```
# vinfra cluster alert delete 1
+--------------+-------------------------------------------------------------------------------+
| Field        | Value                                                                         |
+--------------+-------------------------------------------------------------------------------+
| _type        | undefined_speed                                                               |
| cluster_id   |                                                                               |
| cluster_name |                                                                               |
| datetime     | 2018-08-30T18:02:14.855302+00:00                                              |
| details      | host: stor-1.example.com.vstoragedomain.                                      |
| enabled      | True                                                                          |
| group        | node                                                                          |
| host         | stor-1.example.com.vstoragedomain.                                            |
| id           | 1                                                                             |
| message      | Network interface "eth1" on node "stor-1.example.com.vstoragedomain." has an |
|              | undefined speed                                                               |
| node_id      | 4f96acf5-3bc8-4094-bcb6-4d1953be7b55                                          |
| object_id    | eth1                                                                          |
| severity     | warning                                                                       |
| suspended    |                                                                               |
| type         | Network warning                                                               |
+--------------+-------------------------------------------------------------------------------+
```

This command deletes the alert with the ID 1 from the log.

4.8 Managing Audit Log

4.8.1 vinfra cluster auditlog list

List all audit log entries:

```
usage: vinfra cluster auditlog list
```

Example:

```
# vinfra cluster auditlog list
+----+----------+-----------------------+--------------------------+---------------------+
| id | username | type                  | activity                 | timestamp           |
+----+----------+-----------------------+--------------------------+---------------------+
|  1 | admin    | LoginUser             | User login               | 2018-09-07T08:33:44 |
|  2 | admin    | ChangeNetwrokInterface | Configure network        | 2018-09-07T09:53:58 |
|  3 | admin    | UpInterface           | Bring up interface       | 2018-09-07T09:54:44 |
|  4 | admin    | ChangeNetwrokInterface | Configure network        | 2018-09-07T09:54:54 |
|  5 | admin    | CreateBonding         | Create bonding           | 2018-09-07T09:57:24 |
| 17 | admin    | RemoveNode            | Forget node              | 2018-09-07T12:21:59 |
| 14 | admin    | RemoveNetworkIface    | Delete interface         | 2018-09-07T12:17:14 |
| 15 | admin    | RemoveNode            | Forget node              | 2018-09-07T12:17:49 |
|  6 | admin    | UpInterface           | Bring up interface       | 2018-09-07T10:59:28 |
|  7 | admin    | ChangeNetwrokInterface | Configure network        | 2018-09-07T10:59:46 |
|  9 | admin    | UpInterface           | Bring up interface       | 2018-09-07T11:42:29 |
| 10 | admin    | UpInterface           | Bring up interface       | 2018-09-07T11:42:42 |
| 11 | admin    | CreateBonding         | Create bonding           | 2018-09-07T11:43:46 |
| 12 | admin    | ChangeNetwrokInterface | Configure network        | 2018-09-07T11:52:17 |
| 13 | admin    | ChangeNetwrokInterface | Configure network        | 2018-09-07T11:52:44 |
| 16 | admin    | RemoveNode            | Forget node              | 2018-09-07T12:21:51 |
|  8 | admin    | CreateBonding         | Create bonding           | 2018-09-07T11:00:39 |
| 18 | admin    | RemoveNode            | Forget node              | 2018-09-07T12:22:08 |
| 19 | admin    | UpInterface           | Bring up interface       | 2018-09-07T12:33:16 |
| 20 | admin    | CreateVLAN            | Create VLAN              | 2018-09-07T12:34:18 |
| 21 | admin    | RemoveNetworkIface    | Delete interface         | 2018-09-07T13:26:40 |
| 22 | admin    | LoginUser             | User login               | 2018-09-07T14:50:06 |
| 23 | admin    | LoginUser             | User login               | 2018-09-07T14:51:34 |
| 24 | admin    | CreateNetworkRolesSet | Create custom role set   | 2018-09-07T15:06:03 |
| 25 | admin    | ChangeNetworkRolesSet | Configure custom role set | 2018-09-07T15:37:50 |
| 26 | admin    | RemoveNetworkRolesSet | Remove custom role set   | 2018-09-07T15:39:31 |
| 27 | admin    | CreateNetworkRole     | Create custom role       | 2018-09-07T15:58:50 |
| 28 | admin    | RemoveNetworkRole     | Remove custom role       | 2018-09-07T16:20:22 |
+----+----------+-----------------------+--------------------------+---------------------+
```

This command lists the audit log entries.

4.8.2 vinfra cluster auditlog show

Show details of an audit log entry:

```
usage: vinfra cluster auditlog show <auditlog>
```

`<auditlog>`

Audit log ID

Example:

```
# vinfra cluster auditlog show 1
+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| activity     | User login                       |
| cluster_id   |                                  |
| cluster_name |                                  |
| component    | Users                            |
| details      | []                               |
| id           | 1                                |
| message      | User "admin" login               |
| node_id      |                                  |
| result       | success                          |
| session_id   | 817a19beaf244f92604fbf4b40af2c29 |
| task_id      | 5686556295049300                 |
| timestamp    | 2018-09-07T08:33:44.175797+00:00 |
| type         | LoginUser                        |
| username     | admin                            |
+--------------+----------------------------------+
```

This command shows the details of the audit log entry with the ID 1.

4.9 vinfra cluster problem-report

Generate and send a problem report:

```
usage: vinfra cluster problem-report [--email <email>]
                                     [--description <description>] [--send]
```

`--email <email>`

Contact email address

`--description <description>`

Problem description

`--send`

Generate the problem report archive and send it to the technical support team

Example:

```
# vinfra cluster problem-report --email test@example.com --description "Test report" --send
+---------+--------------------------------------+
| Field   | Value                                |
+---------+--------------------------------------+
| task_id | 8bcfb92f-f02b-4de8-8e44-3426047630e3 |
+---------+--------------------------------------+
```

This commands creates a task to send a problem report with the description “Test report” to the technical support team and use test@example.com as a reply-to address. Note the problem report ID in the task details. You will need to mention it in the support ticket.

Task outcome:

```
+---------+-------------------------------------------------------------------------------+
| Field   | Value                                                                         |
+---------+-------------------------------------------------------------------------------+
| details |                                                                               |
| name    | backend.presentation.reports.tasks.ReportProblemTask                          |
| result  | id: '1001923113'                                                              |
|         | path: /var/cache/problem-reports/report-2018-12-10T15:33:23.391329.tar.gz     |
| state   | success                                                                       |
| task_id | 37d5c13a-001c-4789-8242-96825a17deda                                          |
+---------+-------------------------------------------------------------------------------+
```

CHAPTER 5

Monitoring Storage Cluster

Monitoring the 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.

The main command for monitoring is `vstorage -c <cluster_name> top`. It invokes a text user interface that you can control with keys (press **h** for help).

5.1 Monitoring General Storage Cluster Parameters

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

```
Cluster 'stor1': healthy
Space: [OK] allocatable 1.32TB of 1.44TB, free 1.39TB of 1.44TB
MDS nodes: 3 of 3, epoch uptime: 19d 23h
CS nodes:  3 of 3 (3 avail, 0 inactive, 0 offline)
License: ACTIVE (expiration: 01/10/2021, capacity: 10TB, used: 20.3GB)
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   3 avail      0.0%       0/s   1.1%   192m  19d 23h management.655c19da7e854d6f.nodes.svc.vstoragedomain:2510
    1 avail      0.0%       0/s   0.2%   192m  20d  0h management.b2823b72aeff4ddb.nodes.svc.vstoragedomain:2510
    2 avail      0.0%       0/s   0.0%   192m  19d 23h management.bda1f22b3a854b6c.nodes.svc.vstoragedomain:2510

  CSID STATUS      SPACE  AVAIL REPLICAS   UNIQUE IOWAIT IOLAT(ms) QDEPTH HOST
  1027 active     492.0G 451.4G      295       12     0%       0/0    0.0 management.655c19da7e854d6f.nodes.svc.v
  1025 active     492.0G 449.5G      305       22     0%       0/0    0.0 management.b2823b72aeff4ddb.nodes.svc.v
  1026 active     492.0G 453.0G      289        6     0%       0/0    0.0 management.bda1f22b3a854b6c.nodes.svc.v

    CLID   LEASES     READ    WRITE    RD_OPS    WR_OPS    FSYNCS IOLAT(ms) HOST
    2050    1/222       6B/s     6B/s     0ops/s     0ops/s     0ops/s     0.13/1 management.b2823b72aeff4ddb.nodes.
    2226      1/2       0B/s     0B/s     0ops/s     0ops/s     0ops/s       0/0 management.bda1f22b3a854b6c.nodes.
    2142      0/0       0B/s     0B/s     0ops/s     0ops/s     0ops/s       0/0 management.655c19da7e854d6f.nodes.

TIME                SYS SEV MESSAGE
21-12-18 12:06:24  MDS INF Add new MDS#3 at 10.37.130.79:2510 by request from 10.37.130.79:45672
21-12-18 12:06:24  MON INF MDS#3 was started
21-12-18 12:06:35  MON INF MDS#3 was stopped
21-12-18 12:06:35  MON INF CS#1027 was started
21-12-18 12:06:36  MDS INF New CS#1027 at 10.37.130.79:45742 (0.0.0.655c19da7e854d6f), tier=0
21-12-18 12:06:36  MON INF MDS#3 was started
21-12-18 12:06:38  MDS INF CS#1027 is active
21-12-18 12:06:45  MDS INF The cluster physical free space: 1.4Tb (99%), total 1.4Tb
```

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

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* (page 115).

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 *Understanding Disk Space Usage* (page 109).

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 about the number of these chunk servers:

- 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.

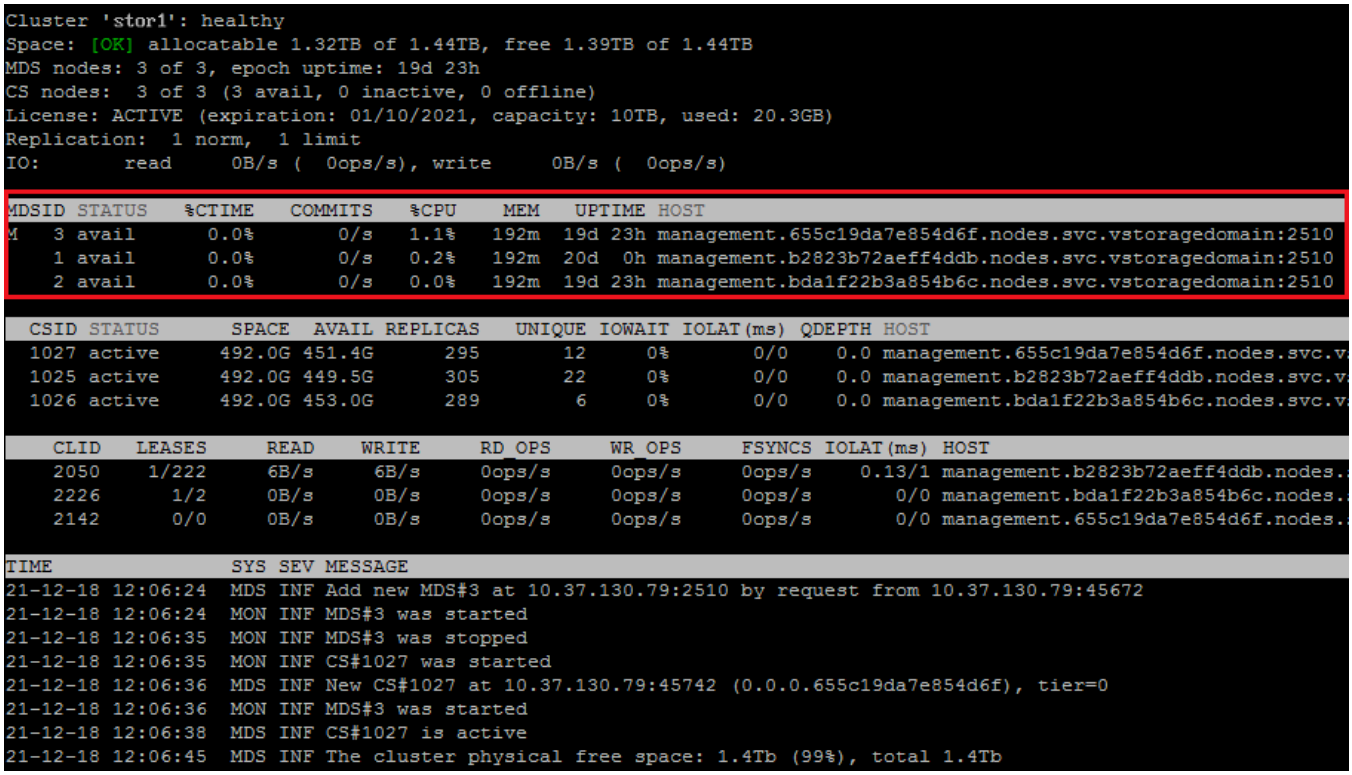
Replication Replication settings. The normal number of chunk replicas and the limit after which a chunk gets blocked until recovered.

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.

5.2 Monitoring Metadata Servers

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



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

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.

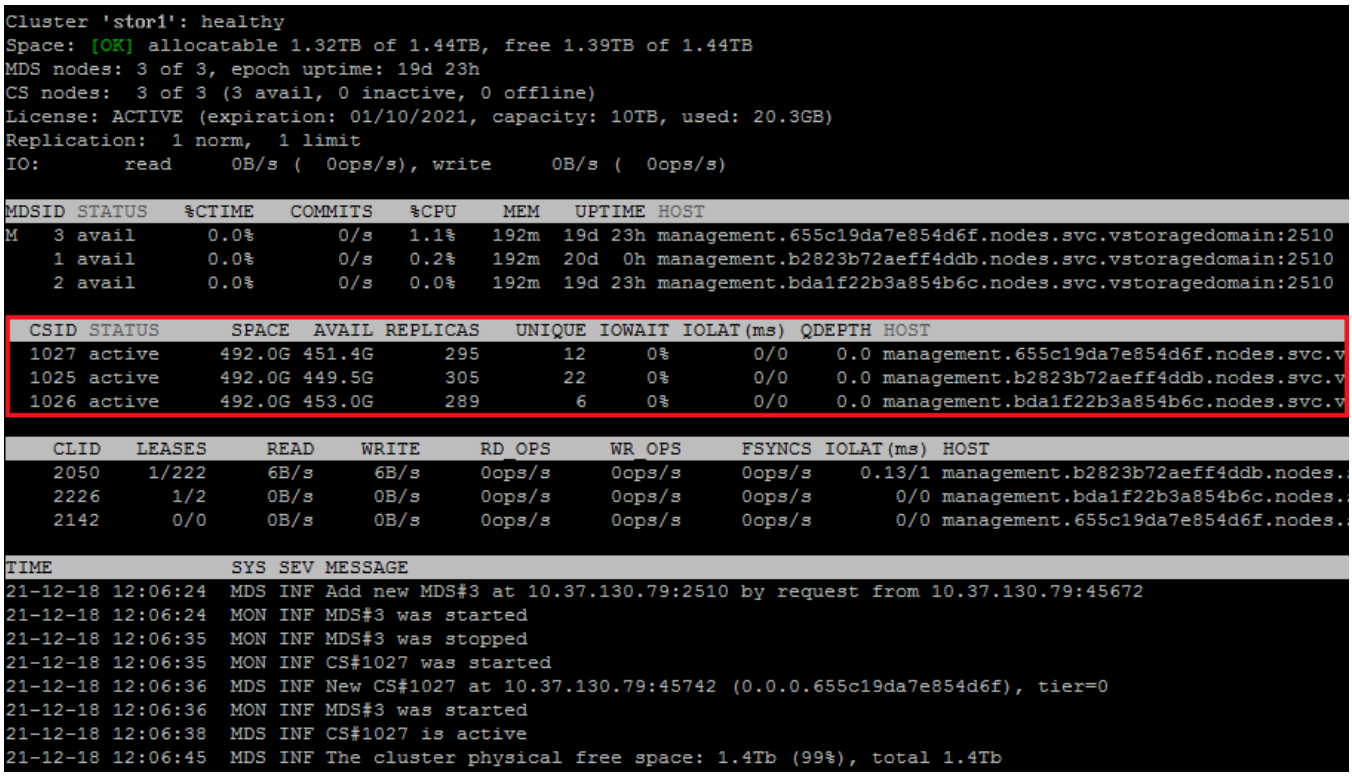
MEM Amount of physical memory the MDS server uses.

UPTIME Time elapsed since the last MDS server start.

HOST MDS server hostname or IP address.

5.3 Monitoring Chunk Servers

By monitoring chunk servers, you can keep track of the disk space available in the storage cluster. To monitor chunk servers, use the `vstorage -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 as follows:

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 server was removed by the administrator.

SPACE Total amount of disk space on the chunk server.

AVAIL Available disk space on the chunk server.

REPLICAS Number of replicas stored on the chunk server.

UNIQUE Number of chunks that do not have replicas.

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.

5.3.1 Understanding Disk Space Usage

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

```
# vstorage -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:

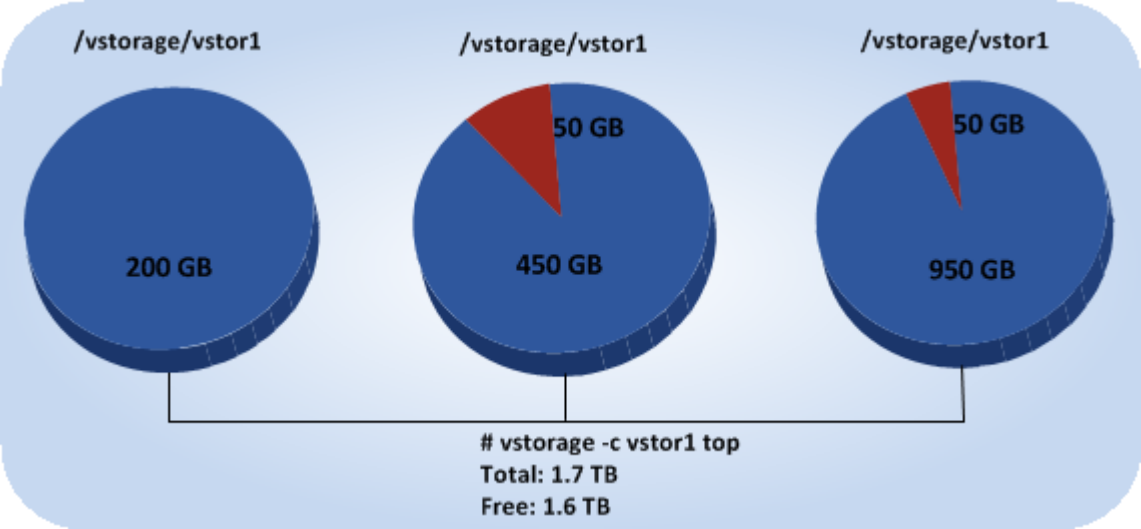
- 1.7TB is the total disk space in the `stor1` cluster. The total disk space is calculated on the basis of used

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.

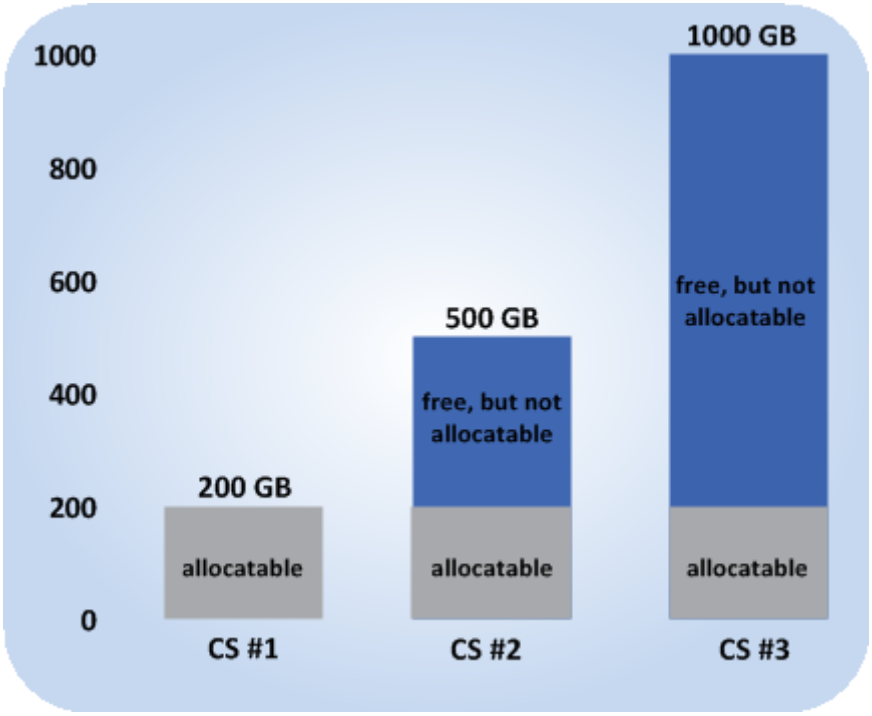
5.3.1.1 Understanding Allocatable Disk Space

When monitoring disk space information in the cluster, you also need to pay attention to the space reported by the `vstorage 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.

Calculation of allocatable disk space is illustrated on 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 is 200 GB, which equals the amount of disk space on the smallest chunk server:

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

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 the replication factor changes to 2, the `vstorage top` command will report the available disk space as 700 GB:

```
# vstorage set-attr -R /mnt/vstorage replicas=2:1
# vstorage -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).

Allocatable disk space may also be limited by license.

5.3.1.2 Viewing Space Occupied by Data Chunks

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

```
# vstorage -c stor1 top
Cluster 'stor1': healthy
Space: [OK] allocatable 1.32TB of 1.44TB, free 1.39TB of 1.44TB
MDS nodes: 3 of 3, epoch uptime: 19d 23h, cluster version: 128
CS nodes:  3 of 3 (3 avail, 0 inactive, 0 offline), storage version: 128
License: ACTIVE (expiration: 01/10/2021, capacity: 10TB, used: 20.3GB)
Replication:  1 norm,  1 limit
Chunks: [OK] 323 (100%) healthy,  0 (0%) standby,  0 (0%) degraded,  0 (0%) urgent,
             0 (0%) blocked,  0 (0%) pending,  0 (0%) offline,  0 (0%) replicating,
             0 (0%) overcommitted,  0 (0%) deleting,  0 (0%) void
FS:  20.3GB in 757 files, 757 inodes,  244 file maps,  323 chunks,  889 chunk replicas
IO:       read     0B/s (  0ops/s), write     0B/s (  0ops/s)
IO total: read   37.1GB ( 473Kops), write  133.7GB ( 4.7Mops)
Repl IO:  read     0B/s, write:     0B/s
Sync rate:   0ops/s, datasync rate:   0ops/s
IO QDEPTH: 0.0 aver, 0.0 max
...
```

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

5.3.2 Exploring Chunk States

The following is a list of all possible states a chunk can be in.

healthy — Number and percentage of chunks that have enough active replicas. The normal state of chunks.

offline — Number and 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.

blocked Number and 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 Number and 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.

replicating Number and percentage of chunks which are being replicated. Write operations on such chunks are frozen until replication ends.

void Number and 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 Number and 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.

urgent Number and 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 Number and 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 Number and 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.

deleting Number and percentage of chunks queued for deletion.

5.4 Monitoring Clients

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

```
Cluster 'stor1': healthy
Space: [OK] allocatable 1.32TB of 1.44TB, free 1.39TB of 1.44TB
MDS nodes: 3 of 3, epoch uptime: 19d 23h
CS nodes:  3 of 3 (3 avail, 0 inactive, 0 offline)
License: ACTIVE (expiration: 01/10/2021, capacity: 10TB, used: 20.3GB)
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   3 avail      0.0%       0/s   1.1%   192m  19d 23h management.655c19da7e854d6f.nodes.svc.vstoragedomain:2510
    1 avail      0.0%       0/s   0.2%   192m  20d  0h management.b2823b72aeff4ddb.nodes.svc.vstoragedomain:2510
    2 avail      0.0%       0/s   0.0%   192m  19d 23h management.bda1f22b3a854b6c.nodes.svc.vstoragedomain:2510

  CSID STATUS      SPACE  AVAIL REPLICAS   UNIQUE IOWAIT IOLAT(ms) QDEPTH HOST
  1027 active     492.0G 451.4G      295       12     0%       0/0    0.0 management.655c19da7e854d6f.nodes.svc.v
  1025 active     492.0G 449.5G      305       22     0%       0/0    0.0 management.b2823b72aeff4ddb.nodes.svc.v
  1026 active     492.0G 453.0G      289        6     0%       0/0    0.0 management.bda1f22b3a854b6c.nodes.svc.v

    CLID   LEASES     READ    WRITE   RD_OPS   WR_OPS   FSYNCS IOLAT(ms) HOST
    2050     1/222     6B/s     6B/s   0ops/s   0ops/s   0ops/s   0.13/1 management.b2823b72aeff4ddb.nodes.
    2226       1/2     0B/s     0B/s   0ops/s   0ops/s   0ops/s       0/0 management.bda1f22b3a854b6c.nodes.
    2142       0/0     0B/s     0B/s   0ops/s   0ops/s   0ops/s       0/0 management.655c19da7e854d6f.nodes.

TIME                SYS SEV MESSAGE
21-12-18 12:06:24  MDS INF Add new MDS#3 at 10.37.130.79:2510 by request from 10.37.130.79:45672
21-12-18 12:06:24  MON INF MDS#3 was started
21-12-18 12:06:35  MON INF MDS#3 was stopped
21-12-18 12:06:35  MON INF CS#1027 was started
21-12-18 12:06:36  MDS INF New CS#1027 at 10.37.130.79:45742 (0.0.0.655c19da7e854d6f), tier=0
21-12-18 12:06:36  MON INF MDS#3 was started
21-12-18 12:06:38  MDS INF CS#1027 is active
21-12-18 12:06:45  MDS INF The cluster physical free space: 1.4Tb (99%), total 1.4Tb
```

The command above shows detailed information about the `stor1` cluster. The monitoring parameters for clients (highlighted in red) are as follows.

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.

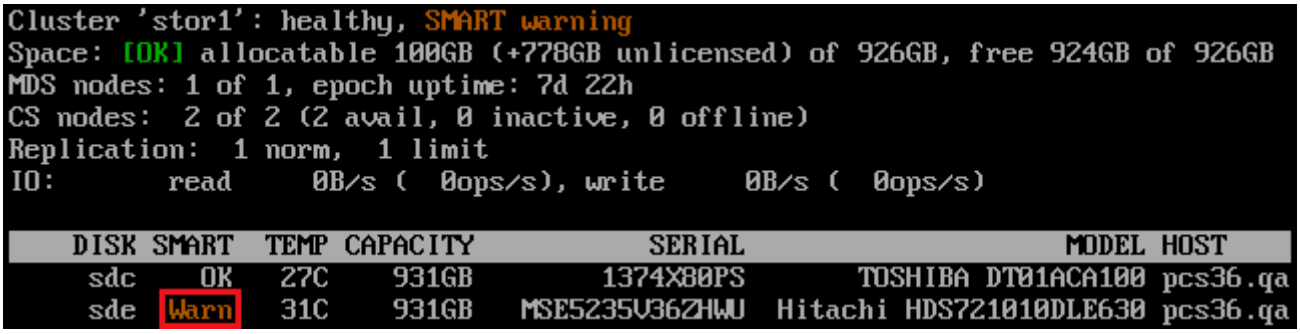
5.5 Monitoring Physical Disks

The S.M.A.R.T. status of physical disks is monitored by the `smartctl` tool installed along with Acronis Software-Defined Infrastructure. For it to work, S.M.A.R.T. functionality must be enabled in node's BIOS. The tool is run every 10 minutes as a `cron` job also added during installation. The `smartctl` tool polls all physical disks attached to nodes in the cluster, including caching and journaling SSDs, and reports the results to the MDS server.

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

```
Cluster 'stor1': healthy, SMART warning
Space: [OK] allocatable 100GB (+778GB unlicensed) of 926GB, free 924GB of 926GB
MDS nodes: 1 of 1, epoch uptime: 7d 22h
CS nodes:  2 of 2 (2 avail, 0 inactive, 0 offline)
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      0.0%       0/s   0.0%    48m   7d 22h pcs36.qa.sw.ru:2510

 CSID STATUS      SPACE  AVAIL REPLICAS   UNIQUE IOWAIT IOLAT(ms) QDEPTH HOST
 1025 active      9.1GB  7.1GB        0        0     0%       0/0    0.0 pcs36.q
 1026 active      916GB  870GB        0        0     0%       0/0    0.0 pcs36.q

 CLID   LEASES     READ    WRITE     RD_OPS     WR_OPS     FSYNCS IOLAT(ms) HOST

TIME                SYS SEV MESSAGE
01-07-14 16:42:19   MON WRN CS#1026 was stopped
01-07-14 16:42:26   JRN INF MDS#1 at 10.29.2.16:2510 became master
01-07-14 16:42:26   MDS WRN License not installed, please add license using comma
01-07-14 16:42:29   MON WRN MDS#1 was stopped
01-07-14 16:42:44   MDS INF CS#1025, CS#1026 are active
01-07-14 16:42:53   MDS INF The cluster is healthy with 2 active CS
01-07-14 16:42:53   MDS INF The cluster physical free space: 925.0Gb (99%), total
```

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:

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.

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

5.6 Monitoring Event Logs

You can use the `vstorage -c <cluster_name> top` utility to monitor significant events happening in the storage cluster. For example:

```
Cluster 'stor1': healthy
Space: [OK] allocatable 1.32TB of 1.44TB, free 1.39TB of 1.44TB
MDS nodes: 3 of 3, epoch uptime: 19d 23h
CS nodes:  3 of 3 (3 avail, 0 inactive, 0 offline)
License: ACTIVE (expiration: 01/10/2021, capacity: 10TB, used: 20.3GB)
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   3 avail      0.0%       0/s   1.1%   192m  19d 23h management.655c19da7e854d6f.nodes.svc.vstoragedomain:2510
    1 avail      0.0%       0/s   0.2%   192m  20d  0h management.b2823b72aeff4ddb.nodes.svc.vstoragedomain:2510
    2 avail      0.0%       0/s   0.0%   192m  19d 23h management.bda1f22b3a854b6c.nodes.svc.vstoragedomain:2510

  CSID STATUS      SPACE  AVAIL REPLICAS   UNIQUE IOWAIT IOLAT(ms) QDEPTH HOST
  1027 active    492.0G 451.4G      295       12     0%       0/0    0.0 management.655c19da7e854d6f.nodes.svc.v
  1025 active    492.0G 449.5G      305       22     0%       0/0    0.0 management.b2823b72aeff4ddb.nodes.svc.v
  1026 active    492.0G 453.0G      289        6     0%       0/0    0.0 management.bda1f22b3a854b6c.nodes.svc.v

    CLID   LEASES     READ    WRITE    RD_OPS    WR_OPS    FSYNCS IOLAT(ms) HOST
    2050    1/222       6B/s     6B/s     0ops/s     0ops/s     0ops/s     0.13/1 management.b2823b72aeff4ddb.nodes.
    2226      1/2       0B/s     0B/s     0ops/s     0ops/s     0ops/s       0/0 management.bda1f22b3a854b6c.nodes.
    2142      0/0       0B/s     0B/s     0ops/s     0ops/s     0ops/s       0/0 management.655c19da7e854d6f.nodes.

TIME                SYS SEV MESSAGE
21-12-18 12:06:24   MDS INF Add new MDS#3 at 10.37.130.79:2510 by request from 10.37.130.79:45672
21-12-18 12:06:24   MON INF MDS#3 was started
21-12-18 12:06:35   MON INF MDS#3 was stopped
21-12-18 12:06:35   MON INF CS#1027 was started
21-12-18 12:06:36   MDS INF New CS#1027 at 10.37.130.79:45742 (0.0.0.655c19da7e854d6f), tier=0
21-12-18 12:06:36   MON INF MDS#3 was started
21-12-18 12:06:38   MDS INF CS#1027 is active
21-12-18 12:06:45   MDS INF The cluster physical free space: 1.4Tb (99%), total 1.4Tb
```

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

TIME Time of event.

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.

The following table lists basic events displayed when you run the `vstorage top` utility.

Table 5.6.1: Basic events

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.

Continued on next page

Table 5.6.1 – continued from previous page

Event	Severity	Description
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.

Continued on next page

Table 5.6.1 – continued from previous page

Event	Severity	Description
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, i.e. do not send any registration messages, or • offline, i.e. have been inactive for longer than `mds.wd.offline_tout`, which is 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`, which is 2 by default. When the cluster fails, the automatic replication is not scheduled any more. So the cluster administrator must take action to either repair failed chunk servers or increase `mds.wd.max_offline_cs`. Setting this value 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 automatic data replication to recover the missing replicas.
Replication completed	MDS info	Generated when the cluster finishes automatic data replication.

Continued on next page

Table 5.6.1 – continued from previous page

Event	Severity	Description
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 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.

5.7 Monitoring 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 `vstorage -c <cluster_name> top` command.
2. Press **V** to display additional information about the cluster. Typical command output may look like this:

```
# vstorage -c stor1 top
Cluster 'stor1': healthy
Space: [OK] allocatable 448.6GB of 492.0GB, free 1.39TB of 1.44TB
```

```
MDS nodes: 3 of 3, epoch uptime: 20d  0h
CS nodes:  3 of 3 (3 avail, 0 inactive, 0 offline)
License: ACTIVE (expiration: 01/10/2021, capacity: 10TB, used: 20.3GB)
Replication:  3 norm,  2 limit
Chunks: [Warning] 187 (57%) healthy,  0 (0%) standby,  0 (0%) degraded,  135 (41%) urgent,
            0 (0%) blocked,  0 (0%) pending,  0 (0%) offline,  1 (0%) replicating,
            0 (0%) overcommitted,  0 (0%) deleting,  0 (0%) void
IO:       read     0B/s (  0ops/s), write  106KB/s (  7ops/s)
...
```

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%.

   For more information on available chunk statuses, see *Exploring Chunk States* (page 112).

CHAPTER 6

Accessing Storage Clusters via iSCSI

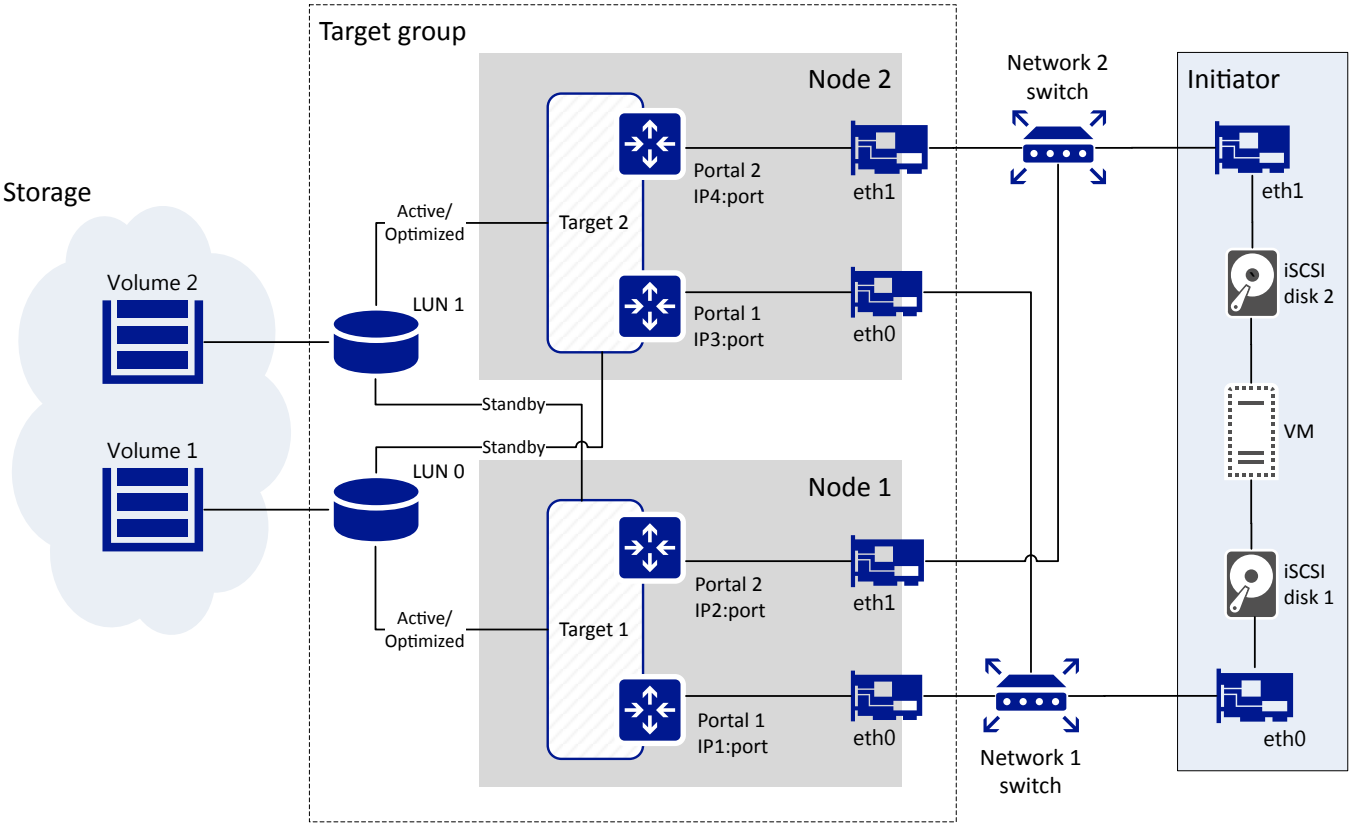
Acronis Software-Defined Infrastructure allows you to export cluster disk space to external operating systems and third-party virtualization solutions in the form of LUN block devices over iSCSI in a SAN-like manner.

In Acronis Software-Defined Infrastructure, you can create groups of redundant targets running on different storage nodes. To each target group you can attach multiple storage volumes with their own redundancy provided by the storage layer. These volumes are exported by targets as LUNs.

Each node in a target group can host a single target for that group if Ethernet is used or one target per FC port if Fibre Channel is used. If one of the nodes in a target group fails along with its target(s), healthy targets from the same group continue to provide access to the LUNs previously serviced by the failed target(s).

You can create multiple target groups on same nodes. A volume, however, may only be attached to one target group at any moment of time.

The figure below shows a typical setup for exporting Acronis Software-Defined Infrastructure disk space via iSCSI.



The figure shows two volumes located on redundant storage provided by Acronis Software-Defined Infrastructure. The volumes are attached as LUNs to a group of two targets running on Acronis Software-Defined Infrastructure nodes. Each target has two portals, one per network interface with the iSCSI traffic type, which makes a total of four discoverable endpoints with different IP addresses. Each target provides access to all LUNs attached to the group. Targets work in the ALUA mode, so one path to the volume is preferred and considered Active/Optimized while the other is Standby. Network interfaces `eth0` and `eth1` on each node are connected to different switches for redundancy. The initiator, e.g., VMware ESXi, is connected to both switches as well and provides volumes as iSCSI disks 1 and 2 to a VM via different network paths. If the Active/Optimized path becomes unavailable for some reason (e.g., the node with the target or network switch fails), the Standby path through the other target will be used instead to connect to the volume. When the Active/Optimized path is restored, it will be used again.

6.1 iSCSI Workflow Overview

The typical workflow of exporting volumes via iSCSI is as follows:

1. Assign the network with the traffic type **iSCSI** to a network interface on each node that you will add to a

target group. See “Managing Traffic Types and Networks” in the *CLI Reference*.

2. Create a target group on chosen nodes, providing details for target WWNs and portals. Targets will be created automatically and added to the group. Target portals will be created on specified network interfaces and ports. See *Creating Target Groups* (page 125).
3. Create volumes and attach them to the target group. See *Managing Volumes* (page 66).
4. Optionally, enable CHAP authorization for the target group, create CHAP accounts, and assign them to the target group. See *Managing CHAP Accounts* (page 134).
5. Optionally, enable ACL authorization for the target group, create a list of initiators that will be allowed to access only specific LUNs. Initiators not on the list will be able to access all LUNs in the target group. See *Managing LUN Views* (page 135).
6. Start the target group. See *Starting and Stopping Target Groups* (page 127).
7. Connect initiators to targets using standard tools of your operating system or product, e.g., `iscsiadm`. Use the `vstorage-target session-list` command to view iSCSI sessions active on a node in a target group.

6.2 Configuring CLI Tool

Before you can use the `vstorage-target` CLI tool to export volumes via iSCSI, set it up as described further. Perform these steps on each node where you plan to run iSCSI targets.

1. Create a configuration file `/etc/vstorage/iscsi/config.json` with at least these mandatory parameters:

```
{
  "ClusterName": "cluster1",
  "VolumesRoot": "/vols/iscsi/vols",
}
```

where `ClusterName` is the name of your storage cluster and `VolumesRoot` is the path to the directory for iSCSI volumes.

You can also set these optional parameters:

- `"PcsLogLevel"`, log level, ranges from 1 (log errors only) to 7 (log all, including debug messages),
- `"LogPath"`, path to log files, default is `"/var/log/vstorage"` (the log will be saved to `vstorage-target.log`),
- `"GetTimeout"`, the timeout for the initiator’s command to read target port group status, default is 3000

ms.

1. Enable the TCM monitor service:

```
# systemctl start vstorage-target-monitor.service
# systemctl enable vstorage-target-monitor.service
```

1. Create the iSCSI volume directory if it does not exist:

```
# mkdir -p /mnt/vstorage/vols/iscsi/
```

If you modify the configuration file later, restart the TCM monitor service to apply changes:

```
# systemctl restart vstorage-target-monitor.service
```

6.3 Managing Target Groups

This section explains how to create and manage groups of iSCSI targets.

6.3.1 Creating Target Groups

Before you create any target groups, assign the network with the iSCSI traffic type to a network interface on each node that you will add to a target group.

To create a target group, you will need a configuration file with a list of nodes to add to the group as well as target WWNs and portals. For example:

```
[
  {
    "NodeId": "01baeabee73e4a0d",
    "WWN": "iqn.2013-10.com.vstorage:test1",
    "Portals": [
      {
        "Addr": "192.168.10.11",
        "Port": 3025
      }
    ]
  },
  {
    "NodeId": "0d90158e9d2444e1",
    "WWN": "iqn.2013-10.com.vstorage:test2",
    "Portals": [
      {
        "Addr": "192.168.10.12",
        "Port": 3025
```

```
      }
    ]
  },
  {
    "NodeId": "a9eca47661a64031",
    "WWN": "iqn.2013-10.com.vstorage:test3",
    "Portals": [
      {
        "Addr": "192.168.10.13",
        "Port": 3025
      }
    ]
  }
]
```

In this configuration file:

- `NodeId` is a node identifier that you can obtain from `/etc/vstorage/host_id` on a node.
- `WWN` is a target world wide name:
  - an IQN if iSCSI protocol is used, e.g., `iqn.2013-10.com.vstorage:test1` (you can only customize the last part after the colon), or
  - a WWPN in NAA format if Fibre Channel protocol is used, e.g., `naa.21000024ff586d3b` (you can obtain the port number from `/sys/class/fc_host/host6/port_name`).
- `Portals` is one or more target portals, IP address and port combinations that the target will be accessible at. The IP address `Addr` is one that belongs to a public network interface on the node that handles the iSCSI traffic type. The port `Port` is optional and defaults to 3260 if omitted.

Once you have the configuration file, e.g., `tg1.json`, you can create the target group with the `vstorage-target tg-create` command. For example, to create an iSCSI target group, run:

```
# vstorage-target tg-create -name tg1 -targets tg1.json -type ISCSI
{
  "Id": "3d8364f5-b830-4211-85af-3a19d30ebac4"
}
```

When you run the command, targets are created on nodes specified in the configuration file and joined to the target group, target portals are created on specified network interfaces and ports.

6.3.2 Starting and Stopping Target Groups

When you create a target group, its targets are initially stopped. You can start them with the `vstorage-target tg-start` command. For example:

```
# vstorage-target tg-start -id 3d8364f5-b830-4211-85af-3a19d30ebac4
```

This command starts all targets in the group with the ID `3d8364f5-b830-4211-85af-3a19d30ebac4`.

All targets in a group can either be running or stopped. So if you add targets to a group of running targets, the new targets will be started automatically.

To stop a target group, use the `vstorage-target tg-stop` command. For example:

```
# vstorage-target tg-stop -id 3d8364f5-b830-4211-85af-3a19d30ebac4
```

6.3.3 Listing Target Groups

You can list target groups with the `vstorage-target tg-list` command that displays basic information about groups. For example:

```
# vstorage-target tg-list
[
  {
    "Id": "3d8364f5-b830-4211-85af-3a19d30ebac4",
    "Name": "tg1",
    "Type": "ISCSI",
    "Running": true,
    "ACL": false,
    "ChapAuth": false,
    "CHAP": {},
    "Mode": 0
  },
  {
    "Id": "78c3b51e-fd9a-485b-91ce-bc0a8171c89d",
    "Name": "tg2",
    "Type": "ISCSI",
    "Running": false,
    "ACL": false,
    "ChapAuth": false,
    "CHAP": {},
    "Mode": 0
  }
]
```

To print complete information about all target groups, use `vstorage-target tg-list -all`.

6.3.4 Printing Details of Target Groups

To print the details of a specific group, use the `vstorage-target tg-status` command. For example:

```
# vstorage-target tg-status -id faeacacd-eba6-416c-9a7f-b5ba9e372e16
```

This command prints the complete details of the target group with the ID `faeacacd-eba6-416c-9a7f-b5ba9e372e16`. One parameter to pay attention to is `NodeState`. It indicates whether a node is in sync with the target group, i.e. aware of its current configuration. The following states can be shown:

- `synced`, node is in sync with the target group,
- `syncing`, node is syncing with the target group,
- `failed`, node failed to sync with the target group (see the `Error` parameter for details),
- `offline`, node is offline,
- `disabled`, node is disabled and its target is offline.

6.3.5 Deleting Target Groups

To delete a target group, use the `vstorage-target tg-delete` command. For example:

```
# vstorage-target tg-delete -id 3d8364f5-b830-4211-85af-3a19d30ebac4
```

This command deletes the target group with the ID `3d8364f5-b830-4211-85af-3a19d30ebac4`.

6.4 Managing iSCSI Volumes

This section describes how to create and manage volumes to be exported via iSCSI.

6.4.1 Creating iSCSI Volumes

To create a volume, use the `vstorage-target vol-create` command. For example:

```
# vstorage-target vol-create -name vol1 -size 1G \
-vstorage-attr "replicas=3:2 failure-domain=host tier=0"
{
```

```
  "Id": "3277153b-5296-49c5-9b66-4c200ddb343d"
}
```

This command creates a 1 GB volume named `vol1` on storage tier 0 with 3:2 replication and host as failure domain.

6.4.2 Listing and Printing Details of iSCSI Volumes

To list volumes, use the `vstorage-target vol-list` command. For example:

```
# vstorage-target vol-list
[
  "3277153b-5296-49c5-9b66-4c200ddb343d",
  "a12110d5-cbbc-498a-acdd-a8567286f927",
  "d5cc3c13-cfb4-4890-a20d-fb80e2a56278"
]
```

Use `vstorage-target vol-stat -all` to print details of all volumes. To print details of a specific volume, run `vstorage-target vol-stat -id <vol_ID>`.

6.4.3 Attaching iSCSI Volumes to Target Groups

To attach a volume to a target group, use the `vstorage-target tg-attach` command. A volume cannot be attached to multiple target groups at the same time. For example:

```
# vstorage-target tg-attach -id 3d8364f5-b830-4211-85af-3a19d30ebac4 \
-volume 3277153b-5296-49c5-9b66-4c200ddb343d -lun 0
```

This command attaches the volume with the ID `3277153b-5296-49c5-9b66-4c200ddb343d` to a target group with the ID `3d8364f5-b830-4211-85af-3a19d30ebac4` as LUN 0. LUN ID numbering must start with 0.

6.4.4 Viewing and Setting iSCSI Volume Parameters

To view and set volume parameters, e.g. redundancy mode, failure domain, or tier, use the commands `vstorage-target vol-attr get` and `vstorage-target vol-attr set`, respectively. For example:

```
# vstorage-target vol-attr get -id d5cc3c13-cfb4-4890-a20d-fb80e2a56278
{
  "chunk-size": "268435456",
  "client-ssd-cache": "1",
  "failure-domain": "host",
```

```
  "replicas": "3:2",
  "tier": "0"
}
# vstorage-target vol-attr set -id d5cc3c13-cfb4-4890-a20d-fb80e2a56278 \
-vstorage-attr "replicas=2:1 tier=1"
```

6.4.5 Increasing iSCSI Volume Size

To increase the size of a volume, use the `vstorage-target vol-grow` command. For example:

```
# vstorage-target vol-grow -id d5cc3c13-cfb4-4890-a20d-fb80e2a56278 -size 2G
```

6.4.6 Setting iSCSI Volume Limits

To set read/write limits for a volume, use the `vstorage-target vol-limits` command. For example:

```
# vstorage-target vol-limits -id d5cc3c13-cfb4-4890-a20d-fb80e2a56278 -read-bps 10485760 \
-write-bps 10485760
```

This command sets read/write speed for the volume with the ID `d5cc3c13-cfb4-4890-a20d-fb80e2a56278` to 10485760 bytes per second.

6.4.7 Detaching iSCSI Volumes from Target Groups

To detach a volume from a target group, use the `vstorage-target tg-detach` command. LUN 0 must be detached last. For example:

```
# vstorage-target tg-detach -id 3d8364f5-b830-4211-85af-3a19d30ebac4 \
-volume d5cc3c13-cfb4-4890-a20d-fb80e2a56278
```

This command detaches the volume with the ID `d5cc3c13-cfb4-4890-a20d-fb80e2a56278` from the target group with the ID `3d8364f5-b830-4211-85af-3a19d30ebac4`.

6.4.8 Deleting iSCSI Volumes

To delete a volume, use the `vstorage-target vol-delete` command. You cannot delete volumes attached to target groups. For example:

```
# vstorage-target vol-delete -id d5cc3c13-cfb4-4890-a20d-fb80e2a56278
```

This command deletes the volume with the ID `d5cc3c13-cfb4-4890-a20d-fb80e2a56278`.

6.5 Managing Nodes

This section describes how to manage nodes in relation to target groups.

6.5.1 Adding Nodes to Target Groups

To add a node to a target group, create a configuration file with details about target WWN and portal. The target will be created automatically on the added node. One node can be added to multiple target groups and the same network interfaces on it can be used simultaneously by multiple targets from different groups.

For example:

```
# vstorage-target node-add -node bbfd0e7a26b1406d -tg 3d8364f5-b830-4211-85af-3a19d30ebac4 \
-targets target.json
```

This command adds the node with the ID `bbfd0e7a26b1406d` to the target group with the ID `3d8364f5-b830-4211-85af-3a19d30ebac4` and creates a target on it according to the `target.json` configuration file that looks as follows:

```
[
  {
    "NodeId": "bbfd0e7a26b1406d",
    "WWN": "iqn.2013-10.com.vstorage:test2",
    "Portals": [
      {
        "Addr": "10.94.104.89",
        "Port": 3260
      }
    ]
  }
]
```

6.5.2 Setting Node Status

To enable or disable a node in a specific target group or all target groups at once, use the `vstorage-target node-set` command. Enabling a node starts its targets while disabling a node stops its targets and moves the `PREFERRED` bit to another node.

For example, to enable a node with the ID `bbfd0e7a26b1406d` in all target groups it belongs to, run

```
# vstorage-target node-set -tg any -node bbfd0e7a26b1406d -enable
```

To disable a node with the ID `bbfd0e7a26b1406d` in the target group with the ID `3d8364f5-b830-4211-85af-3a19d30ebac4`, allowing 60 seconds to move the preferred path to another node (i.e. target), run

```
# vstorage-target node-set -tg 3d8364f5-b830-4211-85af-3a19d30ebac4 -node bbfd0e7a26b1406d \
-disable -release-timeout 60
```

The `release-timeout` parameter sets time in seconds that the initiator has to move the preferred (Active/Optimized) path following the `PREFERRED` bit. If the initiator fails to do so within the given time, the disable operation is cancelled and the node remains enabled. The `PREFERRED` bit, however, is still moved to another node.

The `-force` parameter stops the target(s) on the node at once without moving the `PREFERRED` bit.

6.5.3 Deleting Nodes from Target Groups

To delete a node from a target group, use the `vstorage-target node-del` command. You can delete nodes only after disabling them in specified target groups. Deleting a node also deletes the targets on that node. For example:

```
# vstorage-target node-del -tg 3d8364f5-b830-4211-85af-3a19d30ebac4 -node bbfd0e7a26b1406d
```

This command deletes the node with the ID `bbfd0e7a26b1406d` from the target group with the ID `3d8364f5-b830-4211-85af-3a19d30ebac4`. The node is already disabled in the target group (see *Setting Node Status* (page 132)).

6.6 Managing Targets and Portals

This section describes how to create and manage targets.

The optimal way is to create a single target per node if you use the iSCSI protocol and one target per FC port if you use the FC protocol.

6.6.1 Creating Targets

Typically, targets are created automatically when you create target groups or add nodes to them. However, as you can delete target(s) from a node without removing the node from a target group, you can also create target(s) on such a node again. Use the `vstorage-target target-create` command. For example:

```
# vstorage-target target-create -tg 3d8364f5-b830-4211-85af-3a19d30ebac4 -json target.json
```

This command creates a target based on the `target.json` configuration file in the target group with the ID `3d8364f5-b830-4211-85af-3a19d30ebac4`. The configuration file lists target details like the node to create the target on, WWN, and portal. For example:

```
{
  "NodeId": "bbfd0e7a26b1406d",
  "WWN": "iqn.2013-10.com.vstorage:test22",
  "Portals": [
    {
      "Addr": "10.94.104.90",
      "Port": 3260
    }
  ]
}
```

6.6.2 Adding and Removing Target Portals

To add a portal to a target, use the `vstorage-target target-portal add` command. For example:

```
# vstorage-target target-portal add -wwn iqn.2013-10.com.vstorage:test2 -addr 10.94.104.90 \
-tg 3d8364f5-b830-4211-85af-3a19d30ebac4
```

This command adds a portal with the IP address 10.94.104.90 and default port 3260 to the target with the IQN `iqn.2013-10.com.vstorage:test2` in the target group with the ID `3d8364f5-b830-4211-85af-3a19d30ebac4`.

To delete a portal from a target, use the `vstorage-target target-portal del` command. For example:

```
# vstorage-target target-portal del -wwn iqn.2013-10.com.vstorage:test2 -addr 10.94.104.90 \
-tg 3d8364f5-b830-4211-85af-3a19d30ebac4
```

This command deletes the portal created before.

6.6.3 Deleting Targets

To delete a target from a target group (as well as the node it is on), use the `vstorage-target target-delete` command. For example:

```
# vstorage-target target-delete -tg 3d8364f5-b830-4211-85af-3a19d30ebac4 \
-wwn iqn.2013-10.com.vstorage:test22
```

This command deletes the target with the IQN `iqn.2013-10.com.vstorage:test22` from the target group with the ID `3d8364f5-b830-4211-85af-3a19d30ebac4` and from the node it is located on.

Nodes that have no targets left on them remain in target groups.

6.7 Managing CHAP Accounts

The Challenge-Handshake Authentication Protocol (CHAP) provides a way to restrict access to targets and their LUNs by requiring a user name and a password from the initiator. CHAP accounts apply to entire target groups. Fibre Channel target groups do not use CHAP.

To use CHAP, enable it for the target group:

```
# vstorage-target tg-auth -enable-chap -id <tg_ID>
```

6.7.1 Creating and Listing CHAP Accounts

To create a CHAP account, use the `vstorage-target account-create` command. For example:

```
# vstorage-target account-create -user user1 -desc "User for TG1"
Enter Password:
```

The password must be 12 to 16 characters long.

To list existing CHAP accounts and their details, use the `vstorage-target account-list` command.

6.7.2 Changing CHAP Account Details

To change the password or description of a CHAP account, use the `vstorage-target account-set` command. For example:

```
# vstorage-target account-set description -user user1 -desc "A new description"
# vstorage-target account-set password -user user1
Enter Password:
```

6.7.3 Assigning CHAP Accounts to Target Groups

To assign a CHAP account to a target group, use the `vstorage-target tg-chap` command. For example:

```
# vstorage-target tg-chap set -id faeacacd-eba6-416c-9a7f-b5ba9e372e16 -user user1
```

To remove an assignment, run

```
# vstorage-target tg-chap del -id faeacacd-eba6-416c-9a7f-b5ba9e372e16 -user user1
```

6.7.4 Deleting CHAP Accounts

To delete an unused CHAP account, use the `vstorage-target account-delete` command. For example:

```
# vstorage-target account-delete -user user1
```

6.8 Managing LUN Views

LUN views provide a way to create and manage an access control list (ACL) that limits access to chosen LUNs for specific initiators. Initiators not on the list have access to all LUNs in iSCSI target groups. Volumes exported via Fibre Channel target groups, however, can only be accessed by initiators that are added to group ACL.

To use ACL-based authorization, enable it for the target group:

```
# vstorage-target tg-auth -enable-acl -id <tg_ID>
```

6.8.1 Creating LUN Views

To create a LUN view for an initiator, use the commands `vstorage-target tg-initiator add` or `vstorage-target view-add`. The former command adds an initiator to target group's ACL and creates a view for it. The latter command is used to add views to initiators that are already on the ACL.

For example:

```
# vstorage-target tg-initiator add -alias initiator2 -luns 0,1 \
-tg ee764519-80e3-406e-b637-8d63712badf1 -wwn iqn.1994-05.com.redhat:1535946874d
```

This command adds the initiator with the IQN `iqn.1994-05.com.redhat:1535946874d` to the ACL of the target group with the ID `ee764519-80e3-406e-b637-8d63712badf1` and creates a view allowing it to access the LUNs with the IDs `0` and `1`.

Another example:

```
# vstorage-target view-add -tg faeacacd-eba6-416c-9a7f-b5ba9e372e16 -lun 2 -map 2 \
-wwn iqn.1994-05.com.redhat:1535946874d
```

This command adds a view for the same initiator allowing it to access LUN 2 as well.

6.8.2 Listing LUN Views

To list LUN views for an initiator, use the `vstorage-target view-list` command. For example:

```
# vstorage-target view-list -tg ee764519-80e3-406e-b637-8d63712badf1 \
-wwn iqn.1994-05.com.redhat:1535946874d
```

This command lists views for the initiator with the IQN `iqn.1994-05.com.redhat:1535946874d`.

6.8.3 Changing LUN View Details

To change LUN views for an initiator, use the `vstorage-target view-set` command. For example:

```
# vstorage-target view-set -luns 1 -tg ee764519-80e3-406e-b637-8d63712badf1 \
-wwn iqn.1994-05.com.redhat:1535946874d
```

This command allows the initiator with the IQN `iqn.1994-05.com.redhat:1535946874d` to access only LUN 1. Essentially, it deletes all LUN views for it excluding specified.

6.8.4 Deleting LUN Views

To delete a LUN view for an initiator, use the `vstorage-target view-del` command.

```
# vstorage-target view-del -lun 1 -tg ee764519-80e3-406e-b637-8d63712badf1 \
-wwn iqn.1994-05.com.redhat:1535946874d
```

This command deletes the view for LUN 1 for the initiator with the IQN `iqn.1994-05.com.redhat:1535946874d`.

CHAPTER 7

Advanced Tasks

This chapter describes miscellaneous configuration and management tasks that you may need to perform.

7.1 Updating Kernel with ReadyKernel

ReadyKernel is a kpatch-based service shipped with Acronis Software-Defined Infrastructure and available out-of-the-box on physical servers with active licenses. ReadyKernel offers a more convenient, rebootless alternative to updating the kernel the usual way and allows you not to wait for scheduled server downtime to apply critical security updates. ReadyKernel enables you to receive cumulative kernel patches that fix critical security issues and apply these patches without having to reboot the server. ReadyKernel updates are released for kernels younger than 18 months. When a kernel becomes older that 18 months, you need to switch to a newer kernel to keep receiving ReadyKernel updates.

Upon installation, the patches are loaded into server RAM and immediately applied to the kernel. If the server reboots, these patches are reapplied to the kernel on boot. You can check the details of the applied ReadyKernel patch at any time with `readykernel info`.

If later you install a new kernel or a major kernel update that requires a reboot, the downloaded patches will remain on the server but will not be applied.

In Acronis Software-Defined Infrastructure, ReadyKernel is set to automatically download and apply updates. Checks for new patches are added to each `yum` transaction that takes place on any node in the infrastructure.

Even though ReadyKernel requires no user interaction by default, you can read the following subsections to understand how this tool works and manage it if needed.

7.1.1 Installing ReadyKernel Patches Automatically

ReadyKernel is enabled by default and checks for new patches daily at 12:00 server time by means of a `cron.d` script. If a patch is available, ReadyKernel will download, install, and load it for the current kernel.

To disable automatic updating, run

```
# readykernel autoupdate disable
```

You can re-enable automatic updating later with the following command:

```
# readykernel autoupdate enable <hour>
```

The service will check for patches daily at the specified `<hour>` (set in 24-hour format, server time).

7.1.2 Managing ReadyKernel Patches Manually

7.1.2.1 Dowloading, Installing, and Loading ReadyKernel Patches

To download, install, and instantly load the latest ReadyKernel patch for the current kernel, do the following:

1. Check for new ReadyKernel patches:

   ```
   # readykernel check-update
   ```

2. If a new patch is available, download, install, and instantly load it for the current kernel by running:

   ```
   # readykernel update
   ```

   Note: You can also do this with `yum update`.

ReadyKernel patches are cumulative, i.e. the latest patch includes all the previous ones. To keep the kernel secure, you only need to install and load the latest patch.

7.1.2.2 Loading and Unloading ReadyKernel Patches

To manually load the latest installed ReadyKernel patch to the kernel, do one of the following:

- If an older patch is already loaded, unload it first, then load the latest patch by running:

```
# readykernel load-replace
```

- If no older patches are loaded, load the latest patch by running:

```
# readykernel load
```

To unload the patch from the current kernel, run

```
# readykernel unload
```

7.1.2.3 Installing and Removing ReadyKernel Patches for Specific Kernels

If multiple kernels are installed on the server, you can install a ReadyKernel patch for a specific kernel:

```
# yum install readykernel-patch-<kernel_version>
```

To remove a specific ReadyKernel patch from the server, run

```
# yum remove readykernel-patch-<kernel_version>
```

7.1.2.4 Downgrading ReadyKernel Patches

If you experience problems with the latest ReadyKernel patch, you can downgrade it to an older version if one is available.

To downgrade a patch for the current kernel to the previous version, run

```
# yum downgrade readykernel-patch-$(uname -r)
```

To downgrade a patch for a specific kernel to the previous version, run

```
# yum downgrade readykernel-patch-<kernel_version>
```

You can run these commands multiple times to downgrade to the patch version you need. Alternatively, you can downgrade a patch to a specific version by specifying the desired patch version. For example:

```
# yum downgrade readykernel-patch-12.7-0.4-17.vl7
```

7.1.2.5 Disabling Loading of ReadyKernel Patches on Boot

If for some reason you do not want ReadyKernel patches to be applied at boot time, run the following command:

```
# readykernel autoload disable
```

To re-enable automatic loading of ReadyKernel patches on boot, run

```
# readykernel autoload enable
```

7.1.2.6 Managing ReadyKernel Logs

ReadyKernel logs event information in `/var/log/messages` and `/var/log/kpatch.log`. You can specify logging parameters for the latter in the configuration file `/etc/logrotate.d/kpatch`. For more information on parameters you can use, see the `logrotate` man page.

7.2 Managing Guest Tools

This section explains how to install and uninstall the guest tools. This functionality is required for *Running Commands in Virtual Machines without Network Connectivity* (page 146).

Note: To be able to use the `vinfra` CLI tool mentioned in this section, you may need to perform steps listed in “Providing Credentials” in the *CLI Reference*.

7.2.1 Installing Guest Tools

To be able to install the guest tools in virtual machines, you first need to create and upload compute images from the supplied guest tools ISO files located in `/usr/share/vz-guest-tools/`. Execute the following commands on the controller node of your compute cluster:

- For Linux guest tools:

```
# vinfra service compute image create vz-guest-tools-lin \
--file /usr/share/vz-guest-tools/vz-guest-tools-lin.iso --os-distro linux
Uploading image to server [Elapsed Time: 0:00:05] ...
```

- For Windows guest tools:

```
# vinfra service compute image create vz-guest-tools-win \
--file /usr/share/vz-guest-tools/vz-guest-tools-win.iso --os-distro windows
Uploading image to server [Elapsed Time: 0:00:09] ...
```

Next, you need to attach the created image to a VM and run the guest tools installer. The steps differ for new and already existing VMs and are described in the following subsections.

7.2.1.1 Installing Guest Tools in New VMs

When you create a new VM, you can attach the guest tools image to it and install the guest tools after the operating system. To do this, perform the following steps on the controller node of your compute cluster:

1. Create a new VM with the guest tools image. For example, to create a Linux VM `centos`, run:

```
# vinfra service compute server create centos --network id=private --flavor medium \
--volume source=blank,size=64,boot-index=0,type=disk \
--volume source=image,id=centos7,size=3,boot-index=1,type=cdrom \
--volume source=image,id=vz-guest-tools-lin,size=1,boot-index=2,type=cdrom
```

   Note: Round up the size of volumes to be created from images. E.g., if the OS distribution image is 2.6 GB, use `size=3`.

   In this example, the first volume is a blank virtual HDD, the second volume is the OS distribution image `centos7`, and the third volume is the guest tools image `vz-guest-tools-lin`. Make sure to specify the correct boot order by means of the `boot-index` parameter.

2. Log in to the virtual machine and install an operating system in it.

3. Run guest tools installer inside the VM:

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

```
# mkdir /mnt/cdrom
# mount /dev/sr1 /mnt/cdrom
# bash /mnt/cdrom/install
```

   - Inside a Windows VM, launch the installer in the AutoPlay window if autorun is enabled. Otherwise open the optical drive in Explorer and run `setup.exe`.

     After installing guest tools, restart the VM.

Note: Guest tools rely on QEMU guest agent which is installed alongside the tools. The agent daemon/service `qemu-ga` must be running for the tools to work.

7.2.1.2 Installing Guest Tools in Existing VMs

The steps you need to perform to install the guest tools in existing VMs depend on the guest OS type. They are described in the following subsections.

7.2.1.2.1 Installing Guest Tools in Existing Linux VMs

To install the guest tools in an existing Linux virtual machine, do the following on the controller node of your compute cluster:

1. Create a volume from the uploaded guest tools image. For example:

```
# vinfra service compute volume create vz-guest-tools-lin-vol --storage-policy default \
--size 1 --image vz-guest-tools-lin
```

2. Attach the guest tools volume to the virtual machine. For example:

```
# vinfra service compute server volume attach \
--server centos vz-guest-tools-lin-vol
+--------+--------------------------------------+
| Field  | Value                                |
+--------+--------------------------------------+
| device | /dev/sdb                             |
| id     | 1a40012a-7976-47a1-81f1-ff498cba90af |
+--------+--------------------------------------+
```

3. Log in to the virtual machine, create a mount point for the optical drive with the guest tools image and run the installer:

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

Note: Guest tools rely on QEMU guest agent which is installed alongside the tools. The agent daemon/service `qemu-ga` must be running for the tools to work.

7.2.1.2.2 Installing Guest Tools in Existing Windows VMs

To install the guest tools in an existing Windows virtual machine, do the following on the controller node of your compute cluster:

1. Power off the Windows VM. For example, to stop the `win10` VM, run:

```
# vinfra service compute server stop win10
```

2. Convert its system volume to a template image. You will need the volume ID that you can obtain with `vinfra service compute volume list`. For example, to use the `win10` VM boot volume, run:

```
# vinfra service compute volume list | grep win10
| 7116d747-a1e1-4200-bd4a-25cc51ef006c | win10/windows_10_pro_x64.iso/Boot volume   | <...> |
| ef2f1979-7811-4df6-9955-07e2fc942858 | win10/windows_10_pro_x64.iso/CD/DVD volume | <...> |
# vinfra service compute volume upload-to-image 7116d747-a1e1-4200-bd4a-25cc51ef006c | grep id
| id               | 79da5239-b2bb-4779-ada2-46cb8da8ba0e
```

3. Create a new Windows VM from the template, attaching the guest tools image to it during creation. For example:

```
# vinfra service compute server create newvm --network id=private --flavor medium \
--volume source=image,id=79da5239-b2bb-4779-ada2-46cb8da8ba0e,size=64,boot-index=0,type=disk \
--volume source=image,id=vz-guest-tools-win,size=1,boot-index=1,type=cdrom
```

Note: The size of volume to be created from a template image must be equal to or greater than the minimum volume size specified in the image metadata. You can learn the minimum volume size by using `vinfra service compute image show <image_id> | grep min_disk`.

In this example, the first volume is the template of the original VM's system disk and the second volume is the guest tools image. Make sure to specify the correct boot order by means of the `boot-index` parameter.

4. Once the image is mounted inside the Windows VM, launch the installer in the AutoPlay window if autorun is enabled. Otherwise open the optical drive in Explorer and run `setup.exe`.

   After installing guest tools, restart the VM.

Note: Guest tools rely on QEMU guest agent which is installed alongside the tools. The agent daemon/service `qemu-ga` must be running for the tools to work.

7.2.2 Uninstalling Guest Tools

The steps you need to perform to remove guest tools depend on the guest OS and are described in the following sections.

7.2.2.1 Uninstalling Guest Tools from Linux VMs

To uninstall the guest tools from a Linux guest, log in to the virtual machine and do as follows:

1. Remove the packages:
    1. On RPM-based systems (CentOS and other):

        ```
        # yum remove dkms-vzvirtio_balloon prl_nettool qemu-guest-agent-vz vz-guest-udev
        ```

    2. On DEB-based systems (Debian and Ubuntu):

        ```
        # apt-get remove vzvirtio-balloon-dkms prl-nettool qemu-guest-agent-vz vz-guest-udev
        ```

        If any of the packages listed above are not installed on your system, the command will fail. In this case, exclude these packages from the command and run it again.
2. Remove the files:

    ```
    # rm -f /usr/bin/prl_backup /usr/share/qemu-ga/VERSION /usr/bin/install-tools \
    /etc/udev/rules.d/90-guest_iso.rules /usr/local/bin/fstrim-static /etc/cron.weekly/fstrim
    ```

3. Reload the udev rules:

    ```
    # udevadm control --reload
    ```

After removing guest tools, restart the virtual machine.

7.2.2.2 Uninstalling Guest Tools from Windows VMs

To uninstall the guest tools for Windows, log in to the virtual machine and do as follows:

1. Remove QEMU device drivers from the device manager.

    Important: Do not remove the VirtIO/SCSI hard disk driver and NetKVM network driver. Without the former, the VM will not boot; without the latter, the VM will lose network connectivity.

2. Uninstall QEMU guest agent and guest tools from the list of installed applications.
3. Stop and delete Guest Tools Monitor:

```
> sc stop VzGuestToolsMonitor
> sc delete VzGuestToolsMonitor
```

4. Unregister Guest Tools Monitor from Event Log:

```
> reg delete HKLM\SYSTEM\CurrentControlSet\services\eventlog\Application\VzGuestToolsMonitor
```

5. Delete the autorun registry key for RebootNotifier:

```
> reg delete HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run /v VzRebootNotifier
```

6. Delete the `C:\Program Files\Qemu-ga\` directory.

   If `VzGuestToolsMonitor.exe` is locked, close all the Event Viewer windows. If it remains locked, restart the `eventlog` service:

```
> sc stop eventlog
> sc start eventlog
```

After removing the guest tools, restart the virtual machine.

7.3 Running Commands in Virtual Machines without Network Connectivity

If a VM cannot access a network for some reason, you can still run commands in it from the node the VM resides on. The VM in question must have the guest tools installed in it (see *Managing Guest Tools* (page 141)).

You will need the VM ID that you can obtain with `vinfra service compute server list`. You can also use a virsh domain name that you can get using `virsh list`.

7.3.1 Running Commands in Linux Virtual Machines

To run an arbitrary command inside a Linux VM and receive the output to your console, use the `virsh x-exec` command. For example:

```
# virsh x-exec 1d45a54b-0e20-4d5e-8f11-12c8b4f300db /usr/bin/bash -c 'lsblk'
NAME        MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
loop0         7:0    0 945.9M  1 loop
```

```
loop1         7:1    0     5G  1 loop
  live-rw   253:0    0     5G  0 dm   /
  live-base 253:1    0     5G  1 dm
loop2         7:2    0    32G  0 loop
  live-rw   253:0    0     5G  0 dm   /
sda           8:0    0    64G  0 disk
sdc           8:32   0     1G  1 disk
sr0          11:0    1     2G  0 rom  /run/initramfs/live
```

To copy a file to a Linux VM, use the `virsh x-exec` and `cat` commands. For example:

```
# virsh x-exec 1d45a54b-0e20-4d5e-8f11-12c8b4f300db \
--shell 'cat > test.file' < /home/test.file
```

To get a file from a Linux VM, use the `virsh x-exec` and `cat` commands as well. For example:

```
# virsh x-exec 1d45a54b-0e20-4d5e-8f11-12c8b4f300db \
--shell 'cat /home/test.file' > test.file
```

7.3.2 Running Commands in Windows Virtual Machines

To run an arbitrary command inside a Windows VM and receive the output to your console, use the `virsh x-exec` command. For example:

```
# virsh x-exec bbf4a6ec-865f-4e2c-ac21-8639d1bfb85c --shell dir c:\\
 Volume in drive C has no label.
 Volume Serial Number is D0BE-A8D1

 Directory of c:\

06/10/2009  01:42 PM                24 autoexec.bat
06/10/2009  01:42 PM                10 config.sys
07/13/2009  06:37 PM    <DIR>          PerfLogs
11/12/2018  07:45 AM    <DIR>          Program Files
11/12/2018  07:55 AM    <DIR>          test
11/12/2018  06:23 AM    <DIR>          Users
11/12/2018  07:53 AM    <DIR>          Windows
               2 File(s)             34 bytes
               5 Dir(s)  59,329,495,040 bytes free
```

To copy a file to a Windows VM, use the `virsh x-exec` and `prl_cat` commands. For example:

```
# virsh x-exec bbf4a6ec-865f-4e2c-ac21-8639d1bfb85c \
--shell '%programfiles%\\qemu-ga\\prl_cat c:\\test\\test.file' < /home/test.file
```

To get a file from a Windows VM, use the `virsh x-exec` and `type` commands. For example:

```
# virsh x-exec bbf4a6ec-865f-4e2c-ac21-8639d1bfb85c \
--shell 'type c:\\test\\test.file' > test.file
```

7.4 Setting Virtual Machines CPU Model

Virtual machines are created with the host CPU model by default. If nodes in the compute cluster have different CPUs, live migration of VMs between compute nodes may not work or applications inside VMs that depend on particular CPUs may not function properly. To avoid this, you can find out which CPU model offers compatibility across all nodes in the compute cluster and manually set it as the compute cluster default.

Do the following:

1. Run `virsh capabilities` on each node to print an XML document with information on node's CPU. Join the <cpu> sections from all XML outputs to a single XML file, e.g., `cpu-compare.xml`.
2. Compare the CPU features using `virsh cpu-baseline`. For example:

   ```
   # virsh cpu-baseline cpu-compare.xml | grep model
     <model fallback='allow'>IvyBridge</model>
   ```

   The command will print the most compatible CPU model across all nodes.
3. Set this CPU model for the compute cluster. For example:

   ```
   # vinfra service compute cluster set --cpu-model IvyBridge
   ```

Take note of the following:

- For the list of supported CPU models, run `vinfra service compute cluster set --help`.
- Changing CPU model affects only new VMs (i.e. those created after the change).

See the *CLI Reference* for more details on the command and information on how to use the `vinfra` tool.

7.5 Creating Linux Templates

If you do not have a ready Linux template, you can build one with the `diskimage-builder` tool. The disk image is created with only the root user that has neither password nor SSH keys. You can use the `user data` and `cloud-init` methods to perform initial configuration tasks on VMs that will be deployed from the disk image, for example, create custom user accounts. For more options to customize a VM during boot, refer to the cloud-init documentation.

To create a template and deploy a VM from it, do as follows:

1. Install the `diskimage-builder` package:

```
# yum install diskimage-builder
```

2. For the RHEL 7 guest OS, download the cloud image from the Red Hat Customer Portal (login required) and execute:

```
# export DIB_LOCAL_IMAGE=<path_to_rhel7_image>
```

3. Execute the following command to build a disk image with installed `cloud-init` for the desired Linux guest. For example:

```
# disk-image-create vm centos7 -t qcow2 -o centos7
```

where

- `centos7` is the name of a guest OS. Can be one of the following: `centos6`, `centos7`, `debian`, `rhel7`, or `ubuntu`.

  By default, using the `ubuntu` element will create a disk image for Ubuntu 16.04. To build the Ubuntu 18.04 disk image, add the `DIB_RELEASE=bionic` to the command as follows: `DIB_RELEASE=bionic disk-image-create vm ubuntu -t qcow2 -o ubuntu18`.

- `-o` sets the name for the resulting disk image file.

4. Upload the created disk image using the `vinfra` tool to the compute cluster:

```
# vinfra service compute image create centos7-image --os-distro centos7 \
--disk-format qcow2 --file centos7.qcow2
```

where

- `centos7-image` is the name of a new image.
- `centos7` is the corresponding OS version. Can be one of the following: `centos6`, `centos7`, `debian9`, `rhel7`, `ubuntu16.04`, and `ubuntu18.04`.
- `centos7.qcow2` is the QCOW2-image created on step 3.

5. Create the `user-data` configuration file with a custom user account:

```
# cat <<EOF > user-data
#cloud-config

users:
  - name: user
```

```
    lock-passwd: false
    sudo: ALL=(ALL) NOPASSWD:ALL
    passwd: $6$CjI58V.q7gFvBlMU$9lvue5t8l/VD/tR9L8VTgaRnIksr7ZkSvjtxGxAelJaa.\
WWAmJwRKLjbJwjikOSO2P5DCEYSX/Uf.MuvvhrCE/
EOF
```

where `user` is the name of a custom user and `$6$CjI58V<...>` is a hashed password for the account.

6. Launch the deployment of a VM from the disk image using the configuration file as user data:

```
# vinfra service compute server create centos7-vm --flavor medium --network public \
--user-data user-data --volume source=image,id=centos7-image,size=10
```

where

- `centos7-vm` is the name of a new VM,
- `user-data` is the configuration file created in step 5,
- `centos7-image` is the image added to the compute cluster in step 4.

For more information on using the `vinfra` tool, see *Command Line Reference*.

7.6 Securing OpenStack API Traffic with SSL

By means of the **Compute API** traffic type, Acronis Software-Defined Infrastructure exposes a public endpoint that listens to OpenStack API requests. By default, it points to the IP address of the management node (or to its virtual IP address if high availability is enabled).

Traffic to and from the endpoint can be secured with an SSL certificate. However, as domain names are not used by default, the certificate will need a `subjectAltName` field containing the aforementioned management node IP address. If it does not have such a field, you will need to modify the public endpoint to use a domain name that you have a certificate for.

To secure public OpenStack API traffic with SSL, do the following:

1. Upload the certificate and then private key in the admin panel, on the **SETTINGS** > **Management node** > **SSL ACCESS** screen.
2. Place the CA certificate file to operating system's trusted bundle:

```
# cp ca.pem /etc/pki/ca-trust/source/anchors/
# update-ca-trust extract
```

Alternatively, you can append the `--os-cacert ca.pem` option to each OpenStack client call.

3. If your certificate does not have the `subjectAltName` field, modify all public endpoints to use the domain name for which you have the certificate for. This domain name must resolve to the management node IP address (or to its virtual IP address if high availability is enabled). For example:

```
# openstack --insecure endpoint list | grep public
| 44aa0f53a40e4e52b1c7eeeb20c7811e | <...> | https://10.94.16.12:8774/v2.1/%(tenant_id)s |
| 5a845b4b813047c292db73c42dad5efd | <...> | https://10.94.16.12:8780                    |
| 0b906e518b1041c8b94af7f410403369 | <...> | https://10.94.16.12:9696                    |
| d80af756adf1449f9237c3aeebc9206a | <...> | https://10.94.16.12:8004/v1/%(tenant_id)s   |
| d0e8c7da7d174e1f9aa4efbc6dff2113 | <...> | https://10.94.16.12:5000/v3                 |
| 0e6d3a39d6c44aa883984a35dde434bb | <...> | https://10.94.16.12:9292                    |
| 7d901686bca549f9b294e572f046f634 | <...> | https://10.94.16.12:8776/v2/%(tenant_id)s   |
| 1b68ac7c3f7949fbaeef4a815fe6f3b1 | <...> | https://10.94.16.12:8776/v3/%(tenant_id)s   |

# openstack --insecure endpoint set \
--url https://<DNS_name>:8774/v2.1/%(tenant_id)s 44aa0f53a40e4e52b1c7eeeb20c7811e
# openstack --insecure endpoint set \
--url https://<DNS_name>:8780 5a845b4b813047c292db73c42dad5efd
# openstack --insecure endpoint set \
--url https://<DNS_name>:9696 0b906e518b1041c8b94af7f410403369
# openstack --insecure endpoint set \
--url https://<DNS_name>:8004/v1/%(tenant_id)s d80af756adf1449f9237c3aeebc9206a
# openstack --insecure endpoint set \
--url https://<DNS_name>:5000/v3 d0e8c7da7d174e1f9aa4efbc6dff2113
# openstack --insecure endpoint set \
--url https://<DNS_name>:9292 0e6d3a39d6c44aa883984a35dde434bb
# openstack --insecure endpoint set \
--url https://<DNS_name>:8776/v2/%(tenant_id)s 7d901686bca549f9b294e572f046f634
# openstack --insecure endpoint set \
--url https://<DNS_name>:8776/v3/%(tenant_id)s 1b68ac7c3f7949fbaeef4a815fe6f3b1
```

4. In your OpenRC script, change `OS_AUTH_URL` to the same domain name and remove all parameters related to insecure access. For example:

```
export OS_PROJECT_DOMAIN_NAME=Default
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=admin
export OS_USERNAME=admin
export OS_PASSWORD=<ADMIN_PASSWORD>
export OS_AUTH_URL=https://<DOMAIN_NAME>:5000/v3
export OS_IDENTITY_API_VERSION=3
```

Now you can run OpenStack commands without the `--insecure` option.

7.7 Enabling Backup Gateway Geo-Replication

Make sure the following prerequisites are met:

- Two or more storage clusters with ABGW are deployed.
- All storage clusters are updated to the latest version.
- All storage clusters can ping each other via IP addresses or domain names on port 44445.
- All storage clusters are registered in Acronis Backup Cloud.

To set up geo-replication between two storage clusters, a master and a slave, do the following:

1. Find out the `dc_uid` values from `/mnt/vstorage/vols/acronis-backup/conf.d/dc_uid` on both the master and the slave.
2. Copy the `/mnt/vstorage/vols/acronis-backup/certs/abgw.pem` files from both the master and the slave to a machine from which you will configure replication (i.e. run `vstorage-abgw-ctl`). For example, to `master_abgw.pem` and `slave_abgw.pem`, respectively.
3. Configure replication:

```
# vstorage-abgw-ctl replication \
--master-addr <master_DNS_name> --master-cert master_abgw.pem --master-uid <master_dc_uid> \
--slave-addr <slave_DNS_name> --slave-cert slave_abgw.pem --slave-uid <slave_dc_uid> \
--enable /<account-name>
```

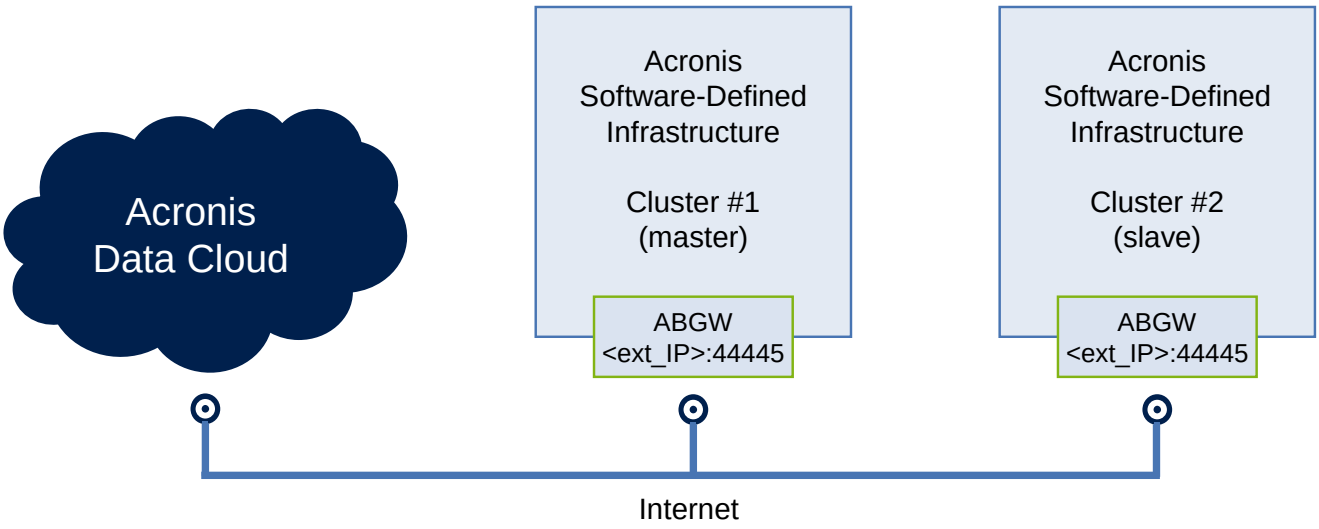
This command will enable replication of existing files. If new files are added to an account, the command will need to be re-run. For more convenience, you can set up a cron job that will run the following script automatically on one of the storage nodes:

```
#!/bin/bash

cd /mnt/vstorage/vols/acronis-backup/storage
for dir in *; do
  vstorage-abgw-ctl replication \
    --master-addr <master_DNS_name> --master-cert master_abgw.pem --master-uid <master_dc_uid> \
    --slave-addr <slave_DNS_name> --slave-cert slave_abgw.pem --slave-uid <slave_dc_uid> \
    --enable /$dir
done
```

If PEM certificates expire, update them in the web interface and exchange them between clusters once again.

If a failover to the slave replica is needed, contact the technical support team. A tool for performing unassisted failovers will be added in a future update.

With geo-replication set up, clusters exchange data as pictured on the diagram.



Where `<ext_IP>` are external IP addresses of each cluster. The DNS configuration will be as follows:

- primary-storage.mysite.domain will resolve to 1.2.3.4, 1.2.3.5, etc. (external IP addresses of cluster #1)
- secondary-storage.mysite.domain will resolve to 5.6.7.8, 5.6.7.9, etc. (public IP addresses of cluster #2)

In your Acronis Data Cloud admin panel, on the **SETTINGS** > **Locations** screen, you can see configuration like this: